IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 29, 2021, accepted August 6, 2021, date of publication August 11, 2021, date of current version August 20, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3104099

AI Meets CRNs: A Prospective Review on the Application of Deep Architectures in Spectrum Management

MDUDUZI C. HLOPHE, (Member, IEEE), AND
BODHASWAR T. MAHARAJ, (Senior Member, IEEE)
Department of Electrical, Electronic and Computer Engineering, University of Pretoria, Pretoria 0028, South Africa

Corresponding author: Mduduzi C. Hlophe (u16250444@tuks.co.za)

This work was supported by the Sentech Chair in Broadband Wireless Multimedia Communications (BWMC) at the University of Pretoria.

ABSTRACT The spectrum low utilization and high demand conundrum created a bottleneck towards fulfilling the requirements of next-generation networks. The cognitive radio (CR) technology was advocated as a *de facto* technology to alleviate the scarcity and under-utilization of spectrum resources by exploiting temporarily vacant spectrum holes of the licensed spectrum bands. As a result, the CR technology became the first step towards the intelligentization of mobile and wireless networks, and in order to strengthen its intelligent operation, the cognitive engine needs to be enhanced through the exploitation of artificial intelligence (AI) strategies. Since comprehensive literature reviews covering the integration and application of deep architectures in cognitive radio networks (CRNs) are still lacking, this article aims at filling the gap by presenting a detailed review that addresses the integration of deep architectures into the intricacies of spectrum management. This is a prospective review whose primary objective is to provide an in-depth exploration of the recent trends in AI strategies employed in mobile and wireless communication networks. The existing reviews in this area have not considered the relevance of incorporating the mathematical fundamentals of each AI strategy and how to tailor them to specific mobile and wireless networking problems. Therefore, this review addresses that problem by detailing how deep architectures can be integrated into spectrum management problems. Beyond reviewing different ways in which deep architectures can be integrated into spectrum management, model selection strategies and how different deep architectures can be tailored into the CR space to achieve better performance in complex environments are then reported in the context of future research directions.

INDEX TERMS Beyond 5G, cognitive radio networks, deep architectures, deep learning, deep Q-learning networks, deep reinforcement learning, energy efficiency, intelligent spectrum management, the Internet of things, machine learning, reinforcement learning.

I. INTRODUCTION AND BACKGROUND

With the migration to beyond 5G networks gaining momentum, paralleled with the phenomenal growth of mobile and wireless devices, the demand for more effective and efficient wireless communications has put enormous pressure on the limited spectrum resources [1]. The impact caused by the demand for more spectrum bands to support the emerging technologies with disparate technological use cases has led to the advocation of the cognitive radio (CR) technology [2]. The CR technology was advocated as the most viable solution for the spectrum shortage and under-utilization quagmire, and through spectrum sharing the licensed spectrum will be reused to support next generation network demands. As a result, the spectrum shortage and spectrum under-utilization duality can be solved through either temporal or spatial reuse of the already licensed spectrum bands. In order to address the spectrum shortage and under-utilization duality, dynamic spectrum access (DSA), driven by the CR technology in spectrum sharing was advocated as the *de facto* technique for spectrum management. Thus, CR-driven DSA consequently became topical among researchers in enabling efficient migration into 5G networks and beyond [3]. However, the vexing dual problem of spectrum scarcity and

The associate editor coordinating the review of this manuscript and approving it for publication was Miguel López-Benítez.

spectrum under-utilization was not the only existing bottleneck in the advancement of wireless technology towards beyond 5G (B5G) networks. The spectrum shortage and under-utilization, coupled with the growing diversity of services of future mobile and wireless networks require that all CR devices be equipped with a technology that will enable more autonomous operation. Also, due to the distributed nature with increased heterogeneity of services in future mobile and wireless networks, the integration of artificial intelligence (AI) strategies into CRs is a requisite. This intuition necessitated that distributed and intelligent spectrum management mechanisms, which will improve the ability to utilize spectrum resources to meet the unprecedented user demands be devised [4].

The CR technology can be defined as a radio system that is capable of obtaining knowledge of its surrounding environment through the spectrum sensing technique, establishes operational policies that forces its internal state to dynamically and autonomously adjust its operational parameters such as the transmission power in accordance with the knowledge obtained in order to achieve the DSA objectives [5]. In behavioral psychology terms, a CR is an intelligent radio device that obtains awareness of its operational environment, learns from it, then adapts its parameters to the statistical variations of the network in real-time [6]. There have been different architectures that have been brought forward for optimal operation of the cognitive radio network (CRN), such as centralized, decentralized, and distributed deployment scenarios. But, due to their fear of being disrupted, as well as interference constraints, centralized control where a central entity manages their operation, has a strong advantage that matches their expected operation. However, because of the distributed nature of future mobile and wireless networks, the distributed architecture holds strong advantages over its counterparts. The failure of one network device (node) does not affect the operation of other nodes because the network has no weak points. In this case, each network node handles its own backup and control operations. Handling its operations locally allows it to act selfishly and independently endure various network failures [7]. However, the distributed approach for network management requires that some additional hardware resources be implemented at node level, which can be very costly.

Apart from the different CRN architectures, CRs environments can be defined depending on the way in which the opportunistic spectrum access (OSA) process is carried out. The first strategy for secondary spectrum usage is the interweave approach, which is based on the original idea of utilizing the spectrum vacancies left by primary users (PUs) in order to establish a data exchange route between secondary users (SUs). The underlay strategy is the next opportunistic spectrum usage approach whereby SUs should always keep their transmission powers below the interference temperature so as to protect the integrity of PU operations. The overlay strategy is the third spectrum reuse approach in which SUs cooperate with PU transmissions while simultaneously performing their own transmissions. However, an interchange between the spectrum usage strategies can be realized through the concept referred to as spectrum handover. For example, if the PU resurfaces on its licensed frequency band to recommence transmission, the CR can move to another frequency band and continue it transmission in overlay mode. Alternatively, it can continue to transmit in the same frequency and operate in underlay mode, but reduce its transmission power level or modulation scheme to avoid interference with the PU [8].

A. MOBILE AND WIRELESS NETWORK EVOLUTION

As the CR technology continues to mature, it has become feasible to consider its application in commercial systems such as cognitive wireless backbones and cognitive machine-to-machine systems. In this way, next-generation CR systems could enable commercial broadband services to co-exist with government services and by doing so, regulators would be able to shoehorn more services into each band in order to alleviate spectrum shortage [9]. The emphasis on current CR research is more on the challenges faced by mobile users, which is to find the temporary spectrum holes in the licensed spectrum that they can opportunistically exploit. Also, the idea that base station (BS) infrastructure could greatly benefit from CR techniques in order to save network operational energy has not been considered yet. It is imperative that when the future of CR systems is being discussed, the issue of BSs as well as other infrastructure such as the mobile edge computing (MEC) server should not be treated as an afterthought. However, a cost-benefit analysis into the operation of several infrastructures does indicate that there are many immediate benefits that outweigh the costs which can drive the increased use of software defined radio (SDR) techniques. Due to their high computational power, which is paralleled with high energy consumption, the BSs present a totally different set of challenges for the CR technology because prior to communication and computation, there is already high energy consumption during spectrum sensing. BSs are supposed to have superior performance to mobile devices since they need to be sensitive to weak signals. However, due to the fact that they have limited power and computational resources, the performance requirements of mobile devices must now match the BSs in spectral bandwidth [10]. Matching the requirements is a very costly process due to equipment being sourced from different vendors and the market time. For example, the idea of improved market time in the midst of the ever-evolving telecommunication standards for a single platform for multiple standards is far-fetched. This is because all vendors and entities are seeking new and effective ways of differentiation in order to maximize their share of customer spend.

Mobile devices and other client-server devices with minimal implementation costs using BSs that are SDR compliant can be able to support multiple devices within a single wireless access network. This would allow service providers to offer a wide range of diversified services, which would further drive the deployment of CR technologies into the network infrastructure. This means that, using the SDR approach, cognitive users can reasonably afford more processing power at less operational costs. However, the pervasive issues related to linearity and sensitivity still remain, i.e., a decrease in either linearity or sensitivity would result in unacceptable impacts on the overall network performance, more especially the power consumption. From the perspective of a wireless network, moving towards cell densification in order to offer increased broadband services to existing mobile clients requires that small cells have outstanding capabilities. Detection, configuring and reconfiguring as well as self-managing their operations in an independent and distributed fashion requires self-organizing capabilities. This aspect was noted by Huawei technologies when highlighting the importance of self-organizing networks (SONs) to streamlined radio deployments as well as spectral allocation [11]. This means that new breakthroughs are required in integrated access nodes and back-haul design in order to cope with the dense networking associated with B5G and the IoT. Due to the plug-and-play, which will become an essential strategy of deployment, integrated access nodes will need to self-organize for the available spectrum blocks for both access and back-hauling tasks. This will be a key capability for enabling high frequency spectrum access, which make SONs the essential key to the imminent connected future [12]. Automating wireless infrastructure configuration, the optimization and the network healing processes actually frees up some of the operational resources which could then be deployed elsewhere. With the positive explosion of connected ''things'', managing, and keeping up with their operations requires an automated approach. However, as a consequence of this, a new set of network assurance challenges that operators will have to deal with emerged [13]. Simultaneously dealing with so many challenges seem to be very daunting such that this challenge can be classified under model and algorithmic deficit problems.

B. THE CR TECHNOLOGY AND SPECTRUM MANAGEMENT

The process of CR operation in terms of optimizing spectrum resource usage began with the reconfigurable radio, which was developed in the early years of television (TV) white space research. Here, the spectrum sensing technique was used to ascertain the availability and/or absence of incumbent activities in order to protect the integrity of their transmissions. However, due to the uncertainties as restrictions of spectrum sensing, telecommunications regulators then realized that the use of the spectrum sensing technique could not provide sufficient protection to incumbent TV users as a stand-alone method [14]. As a result, spectrum databases became the favorable option in spectrum management for many coexistence scenarios in TV bands, and are currently the most dominant dynamic spectrum sharing strategy. The information contained in spectrum databases include the relevant regulations and policies, device locations, their spectrum usage and activities, coverage, and interference levels, as well as services. Spectrum databases are a way of controlling dynamic DSA whereby predictable quality of service (QoS) are provided to mobile devices in order to help to avoid unstable situations such as unnecessary frequent channel switching. This makes the implementation of practical DSA systems much easier [15]. However, as it currently stands, there are some challenges that need to be addressed before spectrum databases can be incorporated into the operation of CR systems. This is because the characteristics of the primary systems as well as their signals tend to differ significantly, i.e., the received signal power levels as well as the coverage areas. A traditional CRN and the decision-making of the CR system is illustrated in FIGURE 1 below.

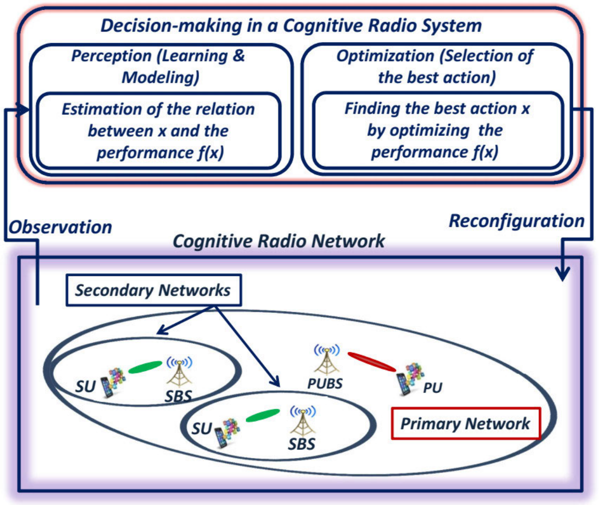
Decision-making in a Cognitive Radio System
Perception (Learning & Modeling)
Estimation of the relation between x and the performance f(x)
Optimization (Selection of the best action)
Finding the best action x by optimizing the performance f(x)
Observation
Reconfiguration
Cognitive Radio Network
Secondary Networks
SU
SBS
PUBS
PU
SU
SBS
Primary Network

FIGURE 1. Illustration of a CRN and decision-making of the CR system.

As shown in FIGURE 1 above, the operation of the CR system can be summarized into two steps, i.e., observation and reconfiguration.

1) PERCEPTION-LEARNING AND MODELING

Learning and modeling in CR systems entails the learning and reasoning concept that fosters optimal resource usage through the estimation of the relationship between environmental state and the performance of the network. The optimal resource usage and management seek to enable a plethora of secondary spectrum access applications [16]. This concept is actually not new to CRs as it stems from the field of AI after being mastered in game theory, dynamic games and stochastic games in particular. Game theoretic approaches offer the basic mathematical optimization tools to model the way in which the interactions take place among autonomous players. In the CR case, a situation in which CRs seek to maximize some specific objective function could be modeled using stochastic games. Modeling CRNs using dynamic and stochastic games aids in studying how the actions taken by CRs are affected by their past experiences and also how they affect their future rewards. In this way, modeling how CRs learn from their past as well as from their neighbors becomes manageable. For example, dynamic games have been used in modeling the interactions among SUs for spectrum access, as well as in problems modeling the interactions between primary users (PUs) and SUs. These models actually range from repeated games to stochastic games up to evolutionary games, and an important aspect of such learning mechanisms is how the learning is performed. Whether the learning has to take the form of a supervised, unsupervised or a reinforced approach is the biggest challenge of the process.

A better choice of the learning approach avoids making the wrong choices in decision making, especially when CR autonomous learning process is considered. Another challenge can be defined in terms of algorithmic design and implementation for CR functionalities. These CR functionalities are related to how mobile devices could be enabled to learn from their past decisions, and reinforce those actions and decisions that led to higher rewards in order to improve their future performance. The design of such learning algorithmic tasks is too complex, and has already presented itself as a huge challenge. This is because the measurements that need to be conducted by learning from the environment also introduce other new challenges related to which measurements should be taken and how the learning should be performed.

2) OPTIMIZATION-SELECTION OF THE BEST ACTION

The optimization process of CR systems entails finding the best action through inference and optimization of the performance to facilitate decision-making. This is done using the information obtained through learning and modeling of the environment. This is the best aspect of CR systems that is related to their intelligence. The primary task of inference in CR systems is to enable the choice of actions leading to efficient decision-making. Inference can be viewed as a process that utilizes both historical and the current knowledge of the environmental context to make better decisions [16]. This means that the learning and reasoning processes discussed above must be enriched with good knowledge to foster increased efficiency of subsequent decision-making [17]. Thus, in the cognitive sense, a tight coupling exists between reasoning and inference. In cognitive reasoning, the power of cognition, which is the complex interaction between previous and present information, is required. This leads to improved reasoning results and culminates in more meaningful and intelligent solutions. As a result, the reasoning-inference duality is the characteristic of the CR that drives it through the process of decision-making. The traditional decision-making in CR systems is the decision on spectrum availability, which involves spectrum sensing, channel selection strategy, and the optimization of radio performance [18]. The decision-making on spectrum availability depends on the CR architecture; it can either be implemented in a centralized, decentralized or distributed fashion. Depending on the architecture employed, decisions are more likely to be influenced by collaborations between CR systems.

3) THE RESIDENT AI OF THE CR TECHNOLOGY

Due to its capability to simultaneously handle spectrum sensing, spectrum access, interference management, and initiating spectrum hand-off, the CR technology presents itself as an effective multi-armed bandit (MAB) problem. Thus, in its own right, the CR technology can effectively address the exploration-exploitation technique without the use of AI strategies. Even without the application of AI strategies, the CR technology is able to effectively address the exploration-exploitation MAB technique within the traditional wireless network space. The MAB technique is firmly rooted in the field of behavioral psychology and is extensively utilized in reinforcement learning (RL). Since the RL strategy is a reward maximization method within the AI space, MAB intuitively describes the maximization of average rewards after pulling its arms several times. As a result, the MAB technique has been very instrumental in coming up with effective cross-layer strategies to enhance the operation of CR systems. For example, solving energy-efficient resource allocation (RA) problems has to be done in parallel with transmission delay, which requires a cross-layer treatment, which, if not an elusive task, is quite a computationally complex challenge. This challenge is exacerbated by the lack of low-complexity joint-layer strategies and coherent research contributions in this aspect are scattered across all layers of the IPv6 protocol stack.

C. NOVELTY AND CONTRIBUTIONS

The novelty and contributions of this prospective review are summarized as follows:

1) The first part discusses the challenges that are faced by CRNs in the IoT and the era beyond 5G networks. Here, the issue of massive IoT deployments and the enormous amount of measurements that need to be taken are discussed. A discussion on the effects that come with addressing requirements with conflicting objectives on the energy consumption closes this section.
2) After the curse of model and algorithmic deficit in current solution approaches has been identified, an introduction to machine learning (ML) and reinforcement learning (RL) is given to enable a smooth understanding of deep architectures. The procedure of building an ML model is discussed before the introduction to RL strategies such as model-free, model-based RL as well as the cost criteria follows. The application of RL strategies in mobile and wireless networks and the problem with RL strategies in spectrum management are discussed in detail.
3) The concrete and rigorous review of deep learning (DL) techniques and their training procedure is followed by their application in mobile and wireless networking problems. In order to enhance understanding, the application of DL techniques is separated into the different DL structures, and the many advantages of DL techniques in spectrum management are discussed in

a prospective perspective with the general idea that it might be helpful to other researchers.

4) In order to effectively handle the dynamics of deep architectures, the symbiotic relationship between DL and RL is explored to usher in the in-depth review of deep architectures. Here, the DRL strategy is discussed as the first deep architecture and its applications in spectrum management are equally explored. Also, the variants of DRL such as deep Q-learning networks (DQNs) are discussed together with their advantages as well as the few applications they have had in spectrum management.
5) In the pursuit of future research endeavors, the last section pinpoints a few open and promising research directions that are worth investigating in future. In that discussion is the issue of enabling predictive analytics through Big Data exploration and exploitation, then the issue of improving data privacy through the use of federated intelligence are pinpointed with the primary objective of providing detailed insights on how deep architectures can be applied to influence future wireless networks specifically to CRN spectrum management.

The remainder of this review is organized as follows: Section II discusses the challenges faced by CRNs in the IoT and the era B5G networks. Section III reviews the current intelligent solutions to the current challenges facing CRNs. Section IV gives a gentle introduction to machine learning and RL, as well as the application of RL in wireless networking problems. In Section V, deep architectures are introduced with a rigorous discussion of DL strategies, as well as the outstanding advantages of DL techniques and ways in which they can be instrumental in solving problems in future mobile and wireless networking. In Section VI, the symbiotic combination of DL and RL is analyzed in order to usher in DRL, which is then discussed in detail together with its various deep architectures. Finally, in Section VII, concluding remarks are given, followed by a discussion of open research problems that require further investigation.

II. CHALLENGES FACED BY CRs IN THE IoT/BEYOND 5G ERA

The ultimate design objective for future mobile and wireless network centers around meeting the diverse QoS requirements of all end-users. Thus, if every future technological invention is equipped with the cognitive technology, and the AI vision gets broadened towards the optimization of the usage radio spectrum, accessing the vacant spectrum bands with less licensed channel interference can be possible. However, this requires that all the network entities of the network layer, the control layer, the management and orchestration layer such as the wireless devices, base stations (BSs), as well as the software defined networking (SDN) controllers to be CR capable. The state-of-the-art OSA algorithms have shown tremendous success in solving spectrum management problems since the proposal of CRs using optimization theory [19]. Algorithms for channel selection and optimum spectrum allocation and spectrum utilization have shown great improvement using noteworthy signal processing algorithms and search heuristics. However, as the wireless technology enters the era of all "things" connected to the internet, spectrum management may no longer be the only problem. Ultra-low latency, superior energy efficiency, exceptional device computational capabilities, privacy, and connectivity all become of paramount importance [20]. With the projected increase in IoT-connected devices globally, huge amounts of communication data will be generated by these connected devices.

The IoT, as a concept, has been viewed as a future internet whereby things possess unique identities, physical interfaces and virtual personalities and linked to exchange environmental observations among themselves as well as with humans [21]. This refers to everyday objects with advanced electronic abilities or high level of technological progress that can be put online and become networked. However, due to the low computational capabilities of mobile and wireless devices, this requirement creates a heavy burden of authentication computing [167]. Thus, the entry of the IoT defines the point where AI strategies have to step in to lend their learning capabilities to the connectivity of future mobile and wireless networks. The IoT, both as a concept as well as a paradigm has rapidly spread over the community of academia and industry and received close attention as among the most pertinent research topics. As a result, it is widely expected that it will be implemented in all kinds of industries in the not-so-distant future.

A. MASSIVE INTERNET OF THINGS DEPLOYMENTS

Because of the distributed nature of the next generation of mobile and wireless networks, the AI strategies need to be implemented in a distributed fashion. Usually, massive IoT deployments are constrained in terms of computation, storage and energy saving capabilities, which consequently pose some serious challenges that require urgent improvement. The increased requirement for even bigger data storage facilities as well as better computational algorithms that will process and make sense out of the data, which marks the entry point of AI strategies into the IoTs. In this case, the IoT era does not only suggest and motivate for the implementation of AI strategies, but also how they should be implemented. This implies that the design efforts for integrating AI into the IoTs should be directed towards enhancing storage and computation, as well as towards the optimization of energy consumption. This will actually not only solve the problems related to performance optimization, but also the huge operational costs. As the IoT continues to steer the operations of the 21st century, new spectrum use cases with staggering resource requirements such as device-to-device, human-to-human, machine-to-machine, vehicle-to-vehicle, and vehicle-to-infrastructure communications, etc., the above requirements need to be achieved with highest precision. Thus, new algorithms need to drive these requirements themselves with the required computational complexity in a distributed manner. However, the current form of the IoT still utilizes the centralized server-client model in order to provide connectivity to various servers and systems [23] and based on current reports, this is quite efficient for now, given that it is still in its infancy. The unfortunate thing about the centralized approach is that it will not be compatible with future distributed mobile and wireless network environments.

B. ENORMOUS AMOUNT OF MEASUREMENTS

As the wireless network becomes more distributed, enormous amounts of measurements will need to be conducted by the learning module of the CR device, and the decisions that need to be made have already presented themselves as major challenges [24]. Subsequently, the decision module of CR devices will be faced with the vexing problem of creating sufficient capacity and energy-efficient operations, combined with low latency for heterogeneous spectrum use cases. On top of these challenges, comes the issues of network reliability and sustainability, which are still very pervasive problems in CRNs. Spectrum management schemes have been carefully designed to succeed in complex environments where future mobile and wireless devices will be operating. As it currently stands, they have shown signs of great success in dealing with the inherent heterogeneity of future generations of mobile and wireless communications. However, be it as it may, there is still more effort that is required in order to break the bottleneck of spectrum sensing errors and huge energy consumption by wireless devices [25]. There has been a lot of research work conducted in order to reduce spectrum sensing errors and also reduce the feedback overhead of spectrum information. This was done by utilizing the spatial and temporal correlations of channel state information (CSI) using compressive sensing (CS) techniques [26]. However, there are still challenges faced by CS algorithms that need to be addressed. CS algorithms require the signals or the signal data to be sparse. But, this is not always the case with real-world signals and data. The problem here is that real-world data has never been sparse and the convergence speed of the existing signal recovery algorithms is relatively slow. In terms of RA techniques, the objective of intelligent resource management is to allocate the available resources in a way that one or more performance metrics can be maximized. This means that the transmission powers, the frequency blocks, the computing power, as well as the memory space can be scheduled among the network terminals based on traffic demands, propagation channel conditions and terminal requirements.

C. CRN REQUIREMENTS WITH CONFLICTING OBJECTIVES

The issue of conflicting objectives requires that a trade-off be quantified between them using balanced optimization. In this way, the optimization of the network throughput, the transmission latency, as well as the energy consumption can be simultaneously guaranteed to ensure better QoS and user experience. The issue of energy consumption is the most important here since it appears as the common denominator in all the parameters to be optimized. Here, the packet dropping rate and flow throughput are the main considerations since we desire to deliver as many packets as possible within their deadlines at the lowest energy possible, which is still elusive. At the moment the problems faced by traditional optimization techniques in efficiently solving the energy consumption problem can be categorized under model or algorithmic deficit. This means that the versatility of current solution algorithms needs to be improved since, in their current state, they have a few limitations going into the 5G era and beyond. Such limitations include, but not limited to: (i) depending on the level of complexity in the scenario. If the scenario is too complex, then accurate mathematical models might not be available to describe the problem. Even if mathematical models can be available, every model is inherently an approximation of the scenario. As a result, a trade-off has to be made between the accuracy of the model and its complexity. Accurate mathematical models may be too complex to handle, whereas simplified models may be not accurate enough to describe the scenario; (ii) another limitation is with the use of static infrastructure. The deployment of static infrastructure might not be flexible enough to adapt to heterogeneous service requirements and randomly evolving environments with unpredictable on-demand connectivity requests.

III. CURRENT INTELLIGENT SOLUTIONS TO CHALLENGES

Solutions to these challenges lie in how well the staggering resource requirements of future use cases and energy efficiencies are handled in the realm of spectrum shortage issues. Since in spectrum management cognitive users take autonomous actions for opportunistic spectrum access, the most convenient model for such network dynamics takes the form of a stochastic/strategic game. Game theoretic techniques have proven to be very efficient optimization tools in the field of mathematical programming. However, with game theoretic algorithms requiring to make computations every time a new problem emerges, even if it is a recurring problem. Another consequence of game theoretic approaches is greedy prioritization, whereby the same experience is reused in every problem. The immediate consequence of this is the wastage of both time and computational resources, resulting in high energy consumption. Therefore, the need to find ways to reuse previous solution experiences as this will ease the energy drain from the computing devices. One way of doing this is to store the actions that previously led to better system performance and just reinforce them when the need arises. It is obvious that this cannot be achieved using greedy prioritization approaches because it will always lead to reusing the same experiences. When it comes to this realization, the natural questions to ask are: (i) how to integrate the required intelligence into the architecture of mobile and wireless networks for improved spectrum resource management? (ii) Where should the experiences required by the algorithms be stored for future reuse? and (iii) Where should the required computations be executed? An ideal response to the above questions would have been to use *cloud-based* approaches.

In *cloud-based* approaches, an artificial brain is placed at a single point of the network that oversees all resource management-related tasks across the whole network. However, a workable solution to these problems is to view the spectrum management problems from a network intelligence perspective and hence motivates for the need to integrate deep architectures into the intricacies of the CR technology. Therefore, the challenge of intermittent connections needs to be addressed with accelerated urgency, and the immediate integration of deep architectures is necessary.

A. CONCERNS WITH CURRENT SOLUTION APPROACHES

Current solutions involve modeling network nodes as random variables with a given spatial distribution in the context of stochastic geometry [27]. The primary concern with employing this tool is that it focuses on the analysis of traditional and non-energy-efficient performance measures. The lack of scientific ways to incorporate the trade-off between the impact of densification in energy efficiency and gain saturation as the density of infrastructure nodes increases puts the problem of energy efficiency under model deficit. ML-based communications systems require no rigidly defined models in order to represent and transform information. This is because they could easily be optimized in an end-to-end manner for a real system with harsh realistic effects. For example, at first glance, the Poisson Kriging at first glance [28] represent attractive alternatives to that of the increasingly popular Bayesian spatial models. Here are the reasons: (i) they are easy to implement with little computational complexity, and (ii) they can account for the shape and size of geographical nodes, thus avoiding the limitations of the conditional auto-regressive models that are commonly used in Bayesian algorithms while allowing for the creation of isopleth risk maps. On the other hand, the recent technological advancements in computer processing units such as the CPUs and graphics card and the distributed data storage make the use of DL more practical than ever in achieving low-complexity joint-layer strategies. However, research in this area is still in its infancy and there are many open problems that must be addressed on methods to integrate DL into the current way wireless communication networks can be operated. Therefore, to address these problems, the applicability of ML techniques has been explored in the wireless communications domain. ML techniques owe their popularity to providing a general framework for solving the complex problems where the models of the phenomenon being learned are too complex to derive and also too dynamic to be summarized in simple mathematical terms [29].

B. THE CURSE OF MODEL AND ALGORITHMIC DEFICIT

Algorithmic deficit problems are very common in mobile and wireless communications. For example, the application of optimal decoders over several well-established channel models tend to be computationally complex. The complexity exponents describing the minimum known complexity that can provably achieve a gap to maximum likelihood performance actually vanishes when high signal-to-noise ratio (SNR) limits are set [30]. In this case, the computational requirements of traditional optimization techniques may prove unbearable for the properties and demands of future wireless network devices. This is the usual case in channels with strong non-linearities such as in optical communications and also for modulation schemes such as continuous phase modulation, as well as in multi-user networks. Given these, the current network-centric philosophy may not be acceptable in simple data gathering and querying applications [31]. This design methodology is not feasible for future large-scale mobile and wireless networks. This is primarily because the large number of protocols executing concurrently make it very difficult to optimize the design, while simultaneously ensuring correct operation. However, on the brighter side, it is well known that neural networks (NNs) are known to be universal function approximators and have shown remarkable capacity for algorithmic learning with recurrent NNs (RNNs) - a construct which has been shown to be Turing-complete [32]. NNs can be executed in a highly parallelized manner by using data and computationally distributed concurrent architectures, and has been demonstrated to work well with small data types conducive to efficient wide single-instruction multiple data operations. Therefore, there is some hope that learned algorithms can be executed significantly faster while simultaneously resulting in lower energy costs than their manually programmed counterparts. Due to the ever-increasing demand for mobile devices and network infrastructure to perform intelligent and low latency tasks, intensive computation and large storage size requirements have continued to be a challenge. As the non-availability of intensive computation and large storage size continue to impede the application of intelligent computational techniques, deep architectures offer promising solutions to these requirements through the integration of computational power and storage of edge-processing nodes. In mobile and wireless networking, the combination of traditional optimization with deep architectures can actually solve these problems which have previously been categorized either as model deficit or as algorithmic deficit.

Model deficit problems usually occur when devices are operating over channels whereby well-established mathematical models do not exist [33]. One example of such cases is molecular communications where inadequate systems are unable to give accurate models for controlled propagation of the carrier molecules. These might include the encoding and decoding of real-world information onto information molecules and vice versa, and also on the transmission and reception systems for carrier or information molecules, which are not readily available in practical scenarios [34]. What exacerbates the issue of mathematically convenient models is the fact that the current generation of communication systems exhibit two main features: (i) high energy cost relative to computation, and (ii) packet collision between concurrent communication due to signal interference [35]. These two features make the design of energy-aware algorithms with carefully managed communication strategy very crucial. This is the most challenging component of future communication systems since most contemporary approaches in design and optimization of wireless network applications are centered on manual customization of the network protocol stack [36]. The application programming interfaces are consistent with manual customization across nearly all platforms that provide an IPv6 stack. Manual customization of the network protocol stack, despite having so many remarkable and applaudable benefits, this comes with rigidly defined models, which cannot deal with the biggest challenges of the IoT that need to be taken very seriously. Surprisingly enough, the concerns about model deficit problems have emerged from within the telecommunications space, but have not yet been addressed. For example, one critical challenge in model deficit emerges when dealing with dense heterogeneous networks, where modeling the positions of nodes in the network is typically difficult to predict deterministically.

In order to solve both model and algorithm deficit problems, ML techniques have been widely recognized for solving the classification and regression problems for which there is no well-defined mathematical model that exists. However, with recent advances in computing power and the ability to collect and store massive amounts of data, ML techniques have found their way into the wireless networking domain in an attempt to put both aforementioned to good use. In addition, the problems that are frequently encountered in CRNs are formulated as classification, detection, estimation, and optimization problems. ML techniques can provide very elegant and practical solutions to these problems. For this reason, the application of ML techniques to CRNs seems almost natural and presents a clear motivation. Therefore, our intention is to elicit more research on the integration of DL techniques into CRNs to solve some of the key challenges associated with the modern IoT systems.

IV. A GENTLE INTRODUCTION TO MACHINE LEARNING AND REINFORCEMENT LEARNING

The use of machine learning (ML) techniques actually depend on the type of available data to be analyzed. As a result, ML techniques can be categorized into three subcategories, i.e., supervised learning, unsupervised learning, and RL.

A. MACHINE LEARNING

Machine learning (ML) is the most prevalent and commonly used of all the AI techniques that are used in the processing Big Data. ML techniques use self-adaptive algorithms that yield increasingly better analysis of patterns, either with experience or with completely new data [37]. They have progressed outstandingly since the proposition of the Turing Machine by Alan Turing, and to date, they are the most common AI techniques that use self-adaptive algorithms. In literature, the ML field is actually split into two subcategories, i.e., supervised and unsupervised. However, some real-life problems are better solved using approaches such as RL and DL, which are also under the universal set of AI strategies. On the one hand, RL strategies, in their fundamental form, can be viewed as a third and separate subcategory of ML since it is a borrowed technique that is firmly rooted in behavioral psychology. In its actual form, it is a combination of behavioral psychology and game theory [38]. On the other hand, DL is an ML method whose techniques are not necessarily viewed as a separate subcategory of ML. However, they are viewed as a means to achieve the ends associated with each one of the three ML sub-categories [39]. This difference is expertly elaborated later in Section V-A. While ML algorithms build their analysis with data in a linear way, DL takes the non-linear approach using a hierarchy of functions. As a result, DL techniques have become more influential owing to its use of a hierarchical level of artificial neural networks (ANNs) in carrying out its processes, (see FIGURE. 6) in Section V-A. The hierarchical functions of DL systems enable algorithms to process input data comprised of features with a non-linear approach to deliver a result in the form of predictions. Thus, depending on the dictates of the DL problem, a DL technique can take either a supervised learning, unsupervised learning, or RL, i.e., DRL perspective.

1) SUPERVISED LEARNING

The supervised learning technique is actually the most fundamental type of ML methods. As an example of the operation of this technique, one can consider a student and a supervisor situation. The student learns from the supervisor through questioning and answering. Then, bringing this analogy to the ML context, the student corresponds to a computer while the supervisor corresponds to the person using the computer. In this way, as it is the case with student and supervisor, the computer learns to map questions to answers by referring from paired samples of questions and answers. The objective of supervised learning is to obtain a generalization. Generalization is defined as the capability that an appropriate answer can be guessed for questions that have not been learned before. Supervised learning techniques have been successfully applied to a variety of real-world problems that require solutions such as *regression*. These are, image processing, speech synthesis and recognition, image processing and recognition, natural language processing, hand-written character recognition, information retrieval, spam filtering, online advertisement, recommendation systems, stock price prediction, weather forecasting, brain signal analysis, gene analysis, etc. In each of these different cases, if the expected answer is a real value, the supervised learning problem is formulated using *regression*; if the expected result is a categorical value, the problem is formulated as a *classification* one; and if an ordinal value is expected, then it is a *ranking* problem.

2) UNSUPERVISED LEARNING

Using the same student-supervisor analogy, unsupervised learning techniques consider a situation whereby the student learns by themselves without a supervisor. Unsupervised learning is actually an automatic form of supervised learning in that the student does not learn from the supervisor, but learns on their own. However, it is sometimes utilized as a pre-processing step in supervised learning tasks. In this case, the computer autonomously collects data through the input and tries to extract useful knowledge without any guidance from the user. Contrary to supervised learning, the learning objective is not necessarily specified. Unsupervised learning techniques have been successfully applied and achieved great success in real-world tasks such as outlier detection, event detection, system diagnosis, data clustering, security, and social network analysis.

B. BUILDING A MACHINE LEARNING MODEL

In ML, model assumptions are very essential prior to building a model to be used to solve problems such as classification, clustering, as well as prediction. Each model has different assumptions that must be met, so checking assumptions is very paramount in both choosing a model to use and in verifying if it is the appropriate one to use. With acceptable assumptions, the ML model is guaranteed to accurately reflect the data and will most likely give accurate results. Secondary to choosing the appropriate model to use, is the issue of gradient flow, and one must have a good understanding of activation and loss functions. One common problem faced in most learning techniques is the issue of gradient flow within a network. This is because some gradients tend to be sharp in certain directions, slow in others, or even zero in other directions, which creates a problem for an optimal selection technique of learning parameters. In order to remedy this issue, the understanding of activation functions are paramount.

1) ACTIVATION FUNCTIONS

Activation functions are one of the parameters used in NN computation, alongside other generalization hyperparameters such as the *learning rate* and *regularization*. These are variables that are set prior to the optimization of the network and make ML algorithms powerful in learning complex and complicated tasks. These functions are used in NNs for the computation of weighted sums of inputs as well as biases, all of which are used to decide whether a neuron node can be fired or not. It does so by manipulating the presented data through some gradient processing such as gradient descent to produce an output. Thus, an activation function of a neural node is the one defining the output of that particular neuron node given an input or a set of inputs [40]. A list of standard activation functions used by ML methods is given in TABLE **1** below [41].

Depending on the representative function, the activation functions listed in TABLE 1 can either be linear or non-linear. Whether the activation function used to control the output of the NN is linear or not is dictated by the application domain. For linear models, the mapping of inputs to outputs using activation functions is performed in the hidden layers before the outputs stage, where the transformation of the vector **x** is given as follows:

TABLE 1. **Activation functions used in machine learning.**

Name	Activation function	Range
Linear function	$f(x_i) = x_i$	$(-\infty, \infty)$
Rectified Linear Unit (ReLU)	$f(x_i) = \max(0, x_i)$	$(0, \infty)$
Sigmoid function	$f(x_i) = \frac{1}{1+e^{-x_i}}$	$(0, 1)$
Hyperbolic tangent	$f(x_i) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	$(-1, 1)$
Gibbs Softmax function	$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$	$(0, 1)$

$$f(x) = w^T x + b, \tag{1}$$

where **x** represents the input vector, **w** denotes the weight matrix, superscript T is the matrix transpose, and b is the bias value of the neuron node [42]. Non-linear activation functions are transfer functions required to convert linear inputs to non-linear outputs and are applied to the outputs of the linear models to produce a transformed non-linear output that is ready for further processing. Here, the neural node or neuron, which takes the sum of the weighted inputs is represented by the non-linear activation function $\phi(\cdot)$. The activation function of a single computational unit is the dot product of the edge weight vector **w** and the input vector **x** plus the scalar bias b, as shown in (2). Thus, the corresponding output of each neural node can be mathematically expressed as follows:

$$f(x) = y(x) = \phi\left(\sum_{i=1}^{n} w_i x_i + b\right) = w \cdot x + b, \tag{2}$$

where the term x_i represents the input of the i^{th} edge, w_i represents the weight of the i^{th} edge. A detailed analysis and derivations of the different activation functions can be found in [41].

2) LOSS OR COST FUNCTIONS

A loss or cost function, also known as an error function, is used to evaluate the performance of an ML model in terms of its ability to estimate the relationship between the input x and the output y. This evaluation is performed using the difference between a predicted value and the actual value, which is estimated by iteratively running the ML model. In each iteration, the estimated prediction (i.e., unknown value of y) is compared with the ground truth (i.e., the known value of y), with the objective of finding parameters and structures or weights that minimize the cost function. In this case, the model is said to be learning to minimize a cost function using gradient-based methods, such as gradient descent, stochastic gradient descent (SGD) that are efficient optimization algorithms that attempt to find either a local or global minima of a given function. Several examples of such functions are listed in TABLE **2** below.

Therefore, a loss function is a way of evaluating how well a specific algorithm models or fits the given data in terms of actual and prediction results [43]. Thus, if the prediction

TABLE 2. Loss functions used in machine learning.

Name	Loss function
Mean squared error (MSE)	$\ell(x,y) = \|\|x-y\|\|_2^2$
Root mean squared error (RMSE)	$\ell(x,y) = \max(0, x_i)$
Categorical cross-entropy	$\ell(x,y) = -\sum_j x_j \log(y_j)$
Hyperbolic tangent	$\ell(x,y) = \tanh(x_i) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
Gibbs Softmax	$\ell(x,y) = \frac{e^{x_i}}{\sum_j e^{x_j}}$

results deviate too much from actual results, the loss function would give a large number (i.e., large error) and vice versa. But with the assistance of optimization functions such as gradient methods, the loss function learns better, and the error is reduced. In this section, selected publications of ML applications in mobile and wireless networks are reviewed. When applying ML techniques into mobile and wireless networking problems, the most important thing to consider is the category of the problem being addressed. Traditionally, ML problems are basically categorized as regression problems, i.e., classification, and clustering problems [45]. Here, the task of the ML technique is the prediction of continuous values for the current inputs so as to identify the class to which the problem belongs to. For example, in content caching problems, one might need to obtain content request probabilities of each user. As such, content caching is thus seen as a regression problem that takes the user profile as input and the content request probabilities as output. However, BS grouping problems can naturally be handled using K-means clustering algorithms, hence belonging to clustering.

C. APPLICATION OF MACHINE LEARNING IN SPECTRUM MANAGEMENT

1) COGNITIVE RADIO NETWORKS

Most applications for tackling algorithm deficit problems on the transmitter side are related to the complexity of the non-convex programs that typically underlie power control optimization algorithms. Mobile and wireless networking tasks that can potentially benefit from ML strategies at the receiver include modulation classification. Such classification problems are justified by the complexity of optimal solutions using traditional optimization techniques, as reported in [46]. Here, a modulation classification based on independent component analysis in conjunction with multiple input multiple output orthogonal frequency-division multiplexing (MIMO-OFDM) signals over frequency-selective, time-varying channels was proposed. Due to the intractability of the problem, the whole processing was divided into two subproblems, i.e., separation and modulation classification. The classification part was implemented using a maximum likelihood and support vector machine (SVM)-based method to achieve performance improvement over time-varying channels, as the invariance was exploited across both the bandwidth and time coherence. A learning-based approach for resource management was proposed in [47], where the authors used a supervised DNN to approximate the unknown non-linear input-output mapping. A class of learn-able algorithms was first characterized and a DNN appropriate for wireless communications was designed. Then, a training set was obtained by running a non-convex solver to produce an optimized output vector for given input channels. Extensive simulations were conducted, and the superiority of DNNs in approximating two considerably complex algorithms was demonstrated.

In another contribution in [48], the authors proposed a self-interference cancellation technique in order to overcome the lack of well-established transceiver chain model of nonlinearities. For this purpose, a supervised NN was used, and full-duplex test bed measurements demonstrated exceptional results from a simple feed-forward NN canceler over the polynomial non-linear canceler with significantly lower computational complexity. The authors in [49] also proposed a mechanism for predicting the outcome of a channel decoding process ahead of the end of the transmission process. For this task, different input features and classification algorithms were discussed and evaluated on their compliance with ultra-reliable low-latency communications (URLLC) requirements. This is because the success of such a prediction scheme would mean that the predictor could be used to request early re-transmissions to reduce transmission latency using automatic re-transmission request (ARQ). In order to demonstrate the feasibility of the proposed scheme, realistic performance estimates incorporating scheduling effects were conducted, and the enhanced hybrid automatic repeat request proved to be feasible over a wide range of scenarios. It is believed that popular content caching reduces network latency and congestion at the core of the network. As a result, the authors in [50] investigated a proactive caching method in the context of cloud radio access networks. In their study, the baseband units were used to predict content request distribution and the mobility pattern of users. The optimization problem was then formulated using joint backhaul-fronthaul loads and content caching and an ML-based echo state network with sub-linear algorithm was proposed as a solution. Using the echo state network, the baseband units demonstrated the ability to predict the content request distribution and mobility pattern for each user. The sub-linear algorithm was then used in determining the appropriate content to cache using limited content request distribution samples. In the performance evaluation step, the results indicated that the proposed algorithm yields significant gains in terms of sum effective capacity compared to other two baseline algorithms.

Since the operation of CR systems involves two processes, i.e., spectrum sensing and spectrum access, energy efficiency is paramount. Thus, for energy efficient operations, a low-cost low-power consumption implementation of spectrum sensing, based on realistic signals was proposed in [51]. The signals were generated using smart embedded devices with different modulation scheme types and the reception interface was an RTL-SDR dongle connected to a MATLAB software. Four signal detection techniques or classifiers, i.e., ANN, SVM, decision tree, and k-nearest neighbors, were used. A comparative analysis of performance of the classifiers was conducted in order to identify the best method using the probability of detection and probability of false alarm. With the results showing no susceptibility to signal to noise ratio values, the ANN and SVM proved to be more accurate than the other two detectors. After spectrum sensing, the reconfigurability of the CR system is a very essential feature for ML techniques in order to dynamically and efficiently allocate the limited resources [44]. Since this part requires a high level of decision-making, this is beyond the traditional ML techniques, as such will be discussed extensively in the RL and DRL sections below.

2) RADIO NETWORK SLICING

Network slicing is meant to support a variety of emerging applications along with a massive number of mobile phones producing large amounts of data, bringing tremendous challenges for network slicing performance. From another perspective, this huge amount of data also offers a new opportunity for the management of network slicing resources. In light of this, the authors in [52] proposed a framework for the optimum performance of device applications with optimized network slice resources. Here, a ML-based network sub-slicing framework in a sustainable 5G environment was proposed in order to optimize network load balancing problems. Each logical slice was divided into virtualized sub-slices of resources and provides the application system with different prioritized resources as necessary. In this way, each sub-slice would focus on spectral efficiency, while the other focus on reduced latency with minimal power consumption. An SVM algorithm was then used to identify different connected device application requirements through feature selection. Then, a K-means algorithm was used to create clusters of sub-slices for similar grouping of types of application services such as application-based, and infrastructure-based services. The four key considerations for the proposed framework include: latency, load balancing, heterogeneity, and power efficiency. A comparative analysis between the proposed framework and existing contributions based on experimental evaluation proves its proficiency for network slicing problems.

The authors in [53] designed an efficient network slicing scheme using a hybrid learning algorithm involving three phases, i.e., data collection, optimal weight feature extraction, and slice classification. Firstly, 5G network slicing data set consisting of attributes associated with various network devices was collected. Then, optimal weight feature extraction was performed whereby a weight is assigned and multiplied to each attribute value for scale variation. The weight function was then optimized by the hybridization of two meta-heuristic algorithms - glowworm swarm and deer hunting optimization algorithms. Using the mentioned attributes, exact network slices were classified into enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and URLLC for each device by a hybrid classifier incorporating deep belief networks (DBNs) and NNs. The experimental results indicated that the proposed hybridization greatly influences the provision of accurate network slicing. A framework for resource orchestration to implement four QoS pillars in a network slicing implementation was proposed in [54]. Traffic classification is used to mark the resource requests in order to serve them better using dedicated logical network slices. A dynamic slicing approach is then implemented using the regression trees algorithm in order to optimally serve multiple network slices. To avoid instability as the resource demands increase, the admission control and slice scheduler are then reformulated as knapsack problems. The performance of the proposed technique was evaluated based on realistic data sets on an experimental prototype - the OpenAirInterface. The simulation results show that it outperforms other baseline algorithms in terms of classification, prediction accuracy, and throughput. A brief summary of ML-based algorithms in spectrum management is tabulated in TABLE 3 below.

D. REINFORCEMENT LEARNING

Reinforcement learning (RL), as opposed to the other ML techniques, takes an opposite task, that of beginning with a complete, interactive, goal-seeking agent. All RL agents have explicit goals, and can sense different aspects of their environments, and can choose actions to influence their environments [55]. As previously stated, the RL strategy is firmly rooted in behavioral psychology and its mathematical fundamentals from game theory, with different variations and add-ons such as Markov decision processes (MDPs) and Q-learning being introduced. Thus, RL is a special kind of supervised learning with some kind of supervision existing in the form of "supervision in reinforcement" [56]. However, unlike in supervised learning, *supervision in reinforcement* refers to a type of supervision that comes in the form of feedback. In this way, the objective of the RL strategy is to acquire the generalization ability in a similar fashion as in supervised learning. The only difference is that the supervisor does not directly give answers to the student's questions, but instead evaluates the student's behavior and gives feedback about it. Therefore, the RL strategy is a sequential decision-maker because of the state-action pairs that occur one after another and judges actions based on the results they produce. The RL algorithm, which is also known as the agent, receives its feedback from the environment. This feedback is only available after an output to a given input or observation has been selected. This kind of feedback is the one that indicates the degree to which the output, known as action, which serves to fulfill the goals of the RL agent. The task is then to train the agent which interacts with the environment and encounters different scenarios, i.e., the states.

The RL strategy operates in this way: in performing certain actions by following a certain policy or strategy in those states, a reward that eventually leads into a new state is received. In this case, the objective of the agent is maximizing the total average reward, while also *reinforcing* those actions

TABLE 3. Application of machine learning techniques in spectrum management.

Application	Objective(s)	Technique(s)	Algorithm(s)	Key consideration(s)	Refs
Cognitive radio	Minimizing transmit power	Channel & traffic statistics	Power & rate allocation	Transmission costs, energy efficiency	[44]
	Modulation classification	Maximum likelihood estimation	Support vector machines	Bandwidth and time coherence	[46]
	Power allocation optimization	Supervised deep neural networks	Weighted minimum mean squared error	DNN size computational time	[47]
	Interference cancellation	Supervised neural networks	Feed-forward NN canceller	NN size, computational complexity	[48]
	Prediction - channel decoder	Supervised auto-encoder	Binary predictor	Effective block error rates	[49]
	Proactive caching	Echo state network	Sub-linear algorithm	Periodic mobility pattern, memory capacity	[50]
	Primary user signal detection	ANN, SVM, decision tree, KNN	RTL-SDR and MATLAB software	Implementation cost and power consumption	[51]
Network slicing	Improving spectral efficiency	Feature selection	Support vector machine, K-means	Load balancing, Latency, Power efficiency	[52]
	Network slice provisioning	Learning hybridization	Deep belief networks, Neural networks	Weight assignment, scale variation	[53]
	Network slice orchestration	Regression trees, Knapsack algorithm	Root mean squared error	Number of tenants, resource demand increase	[54]

that previously led to better rewards. As a result, the number of times the agent performs a certain action increases in subsequent actions. In summary, the RL algorithm works by applying sequential decision-making whereby through the agents' interaction with the environment, takes actions based on its environmental observations, and subsequently receives a reward that takes the system state to the next level, while also receiving feedback on each selected action [57]. For example, assuming that while the agent is in state s_t at time t, it uses a policy/strategy π to select an action (i.e., behavior) a_t from a set of possible behaviors, i.e., $a_t \in \mathcal{A}$, where $\mathcal{A}$ denotes the set of possible behaviors; upon selecting and executing a certain behavior, it obtains a reward r_t, which leads it to reaching a new state s_{t+1}. This is such that this can be expressed as the tuple (s_t, a_t, r_t, s_{t+1}). Therefore, the best way to understand the RL strategy is by first defining its basic concepts such as environments, agents, states, actions, rewards, discount factor, policy, and value functions as follows [60]:

- **Environment**: The environment can be defined as the space through which the agent makes its moves by taking an action a_t as input in the current state s_t, returns the reward r_t as the output which takes the agent to its next state s_{t+1}. In this case, the environment can be thought of as a transfer function that transforms an action that is taken in the current state s_t into the next state s_{t+1} after receiving a reward r_t. However, the function of the environment is not always known, and it is just a *black box* where only the inputs and outputs can be seen.
- **Agent**: The agent can be defined as a function that transforms the new state s_{t+1} and reward r_t into the next action a_{t+1}. In short, an agent takes actions. However, as opposed to the environment, the function of the agent can be known.
- **State and Action**: The state and action can be jointly defined as the concrete and immediate situations that the agent can be in at any given time. The start-state s_0 is defined in (6) below.
- **Reward**: The reward is the feedback that evaluates either the success or the failure of the agents' actions. The rewards that effectively evaluates the agent's actions can either be immediate or be delayed.
- **Discount factor**: The discount factor was designed to control future rewards and make them have less weight that immediate ones by enforcing some kind of short-term satisfaction in the agent. Usually denoted by $\gamma^t \in [0, 1]$, the discount factor is multiplied by the future rewards as discovered by the agent in order to dampen the effect of immediate rewards on the actions chosen by the agent.
- **Policy**: The policy or strategy, denoted as π, is employed by the agent in determining the next action a_{t+1} based on the current environmental state s_t. It operates by mapping states to the actions that promise to lead to the highest reward.
- **Value function**: The value, denoted by V, is the expected long-term return with discount γ, such that $V^{\pi}(s)$ is the expected long-term return of the current state while following the policy π.
- **The Q-value**: The Q-value, also referred to as the action-value function maps the state-action pairs to the rewards, such that $Q^{\pi}(s_t, a_t)$ refers to the long-term return of the current state s_t, after taking action a_t under policy π. Lastly,
- **Trajectory**: The trajectory is defined as the sequence of states and actions that influence future states and actions.

RL strategies can be divided into three sub-categories: (i) model-free and the (ii) model-based methods [58], with the (iii) cost criteria existing as a sub-category that is firmly embedded within model-based RL methods as one of its randomization techniques. These sub-categories are defined based on the way in which each one of them obtains its rewards depending on the policy and system dynamics (i.e., the model).

E. RL ALGORITHMIC ASPECTS AND LEARNING POLICIES

In the application of RL strategies, setting up the decision-making procedure is the initial step, where the available information is specified to the decision maker. In the case of deterministic RL models, the transition probabilities can either be zero or one. Thus, if the initial state is known, i.e., the start-state s_0, the controller is able to fully predict the system state evolution as a result of applying a sequence of actions. This kind of control model constitutes *open loop* policies, which are defined as policies that require no a priori information regarding the initial state [62]. However, there exist some situations whereby the state evolution cannot be fully predicted. In these situations, it is advisable to use policies that require more information on the system, because when the transition probabilities are not only zero or one, it will turn out to be constrained MDPs (CMDPs). Then, the performance of the system under study can be improved by choosing actions in a random way, using randomization mechanisms. However, it is paramount to define the way in which an RL agents deals with the perception of the environment. In this case, it is fitting to define MDPs for a given environment. This involves the definitions of Bellman optimality equations and how the value functions and policy functions are obtained for any given state. For instance, the Bellman optimality equation and the Bellman expectation equation are similar, which means that they are not the same. The difference is the way in which action is taken.

In the Bellman optimality equation, the value of a state can be decomposed into the immediate reward and the discounted value of the successor state as follows [59]:

$$V(s) = \mathbb{E}\left[R_{t+1} + \gamma^t V(s_{t+1})|s_t = s\right], \tag{3}$$

where R_{t+1} and s_{t+1} respectively represent the immediate reward and the successor state. The same analysis applies also when defining the fundamental property of the optimal action-value function as follows:

$$Q(s,a) = \mathbb{E}\left[R_{t+1} + \gamma^t \max_{a' \in \mathcal{A}} Q(s',a')\right]. \tag{4}$$

Equations (3) and (4) can still stand for the Bellman expectation equation, by, instead of taking the average of the actions, the agent takes the action with the maximum value. This distinction is very important when the dictates of the problem to be solved requires such clarity in terms of being of function approximation, i.e., Q-learning, or of policy-based search, i.e., policy gradient, methods.

1) MODEL-FREE REINFORCEMENT LEARNING

The Q-learning algorithm is a model-free RL algorithm to learn the value of an action in a particular state without requiring any model of the environment. In model-free RL, the value function is estimated first, then the policy can be determined based on the estimated value function [61]. As a result, it is able to handle problems with stochastic transitions and rewards without requiring any adaptations, and has shown tremendous success in many real-world problems where problems with discrete states and actions are considered. It uses incremental trial-by-trial or trial-and-error learning of a cached habit strength, such that habitual responding persists unchanged after revaluation as an automatic movement procedure. The computation of the Q-learning algorithm predominantly uses temporal differences (TD) prediction error mechanism to estimate the value of $Q(s, a)$. The TD thus becomes the agent that learns from the environment through computation episodes without any prior knowledge. For instance, if in the problem being addressed the agent has to start computation from a starting point, say s_0, follow a computation trajectory towards reaching the target, $Q^*(s, a)$ is the expected value. This is known as the cumulative discounted reward of taking an action, a, in state, s, then following the optimal policy. The Q-learning algorithm uses the TD to estimate the value of $Q^*(s, a)$. It maintains a Q-table $Q(\mathcal{S}, \mathcal{A})$, where $\mathcal{S}$ and $\mathcal{A}$ respectively represent the sets of states and actions. Thus, $Q(s, a)$ becomes the current estimate of $Q^*(s, a)$. Then, depending on the dictates of the problem, the Q-function can use either of the Bellman equations and take the two inputs, s and a. This process is repeated, updating the Q-table and maximizing the Q-value, until the learning is stopped by reaching the termination condition. Then, the $Q(s, a)$ returns the expected future reward, as follows:

$$\begin{aligned} Q(s,a) &= Q(s,a) \\ &\quad + \alpha_t\left[R(s,a) + \gamma^t \max Q(s',a') - Q(s,a)\right], \end{aligned} \tag{5}$$

where α_t represents the learning rate, $\max Q(s', a')$ is the estimate of the future value, $Q(s, a)$ represents the old value, then $\max Q(s', a') - Q(s, a)$ is the TD prediction error.

In typical problems usually encountered in decision theory, the policy iteration considers a function $J(\pi)$ as the performance measure such that the objective function is given as follows:

$$J(\pi) = \mathbb{E}\left\{\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)|s_0, \pi\right\}, \tag{6}$$

where the discounted reward r is being summed from the start-state, s_0 over t time steps. Since the time duration of observing the system is infinite, i.e., $t = \{0, 1, 2, \cdots, \infty\}$, the system observation is said to be performed over an infinite horizon. However, the finite horizon, $t = \{0, 1, 2, \cdots, T-1\}$ is actually an intuitive and fundamental formalism for decision-theoretic planning when considering discrete-time problems, typical of wireless networking problems. Equation (6) can alternatively be defined as follows:

$$Q^\pi(s,a) = \mathbb{E}\{\sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k}|s_t = s, a_t = a, \pi\}, \tag{7}$$

where the term k represents the number of steps taken after the start state s_0. In both definitions, a discounted weighting of states is defined as $d^\pi(s)$, where the discount $d^\pi(s)$ is encountered when starting at s_0 while following the policy π [60]:

$$\pi : d^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t Pr\{s_t = s|s_0, \pi\}, \tag{8}$$

which is the average reward of a policy that the agent receives on every time step it acts on the environment. So, using (6) above, the reward computation considers the environmental dynamics being characterized by the following state transition probabilities, $\mathcal{P}^a_{s,s'} = Pr\{s_{t+1} = s'|s_t = s, a_t = a\}$ [60].

2) MODEL-BASED RL AND POLICY-BASED SEARCH ALGORITHMS

In model-based RL, an action is chosen based on declarative memory of previous hedonic values embedded in modeled world relationships. Contrary to model-free RL, here an action is adjusted after the devaluation of the outcome or when the contingency degradation needs to be restated in order to upgrade the goal value or to reduce uncertainty. Model-based RL has its roots in control theory, as a result in most literature the system state is denoted as x instead of s and action as u instead of a. The term normally used for *action* is *control* such that the controller is the one that influences both the incurred costs as well as the evolution of the system by choosing actions at each time step. The algorithms used here are actually policy-based search algorithms that use tree search mechanisms in their computation. The striking difference it has from model-free RL is that the rewards obtained depend on the policy and the system dynamics, that is the model. Here, the definition of a cost function enables for the computation of optimal actions directly from the model. The learning policy is a value that is adopted by the learning agent to guide it in learning the best solution. An ϵ-greedy exploration policy is used to learn the best action among the actions available in the action space [63]. The three most dominant policies used in RL strategies are discussed, the on-policy is illustrated in FIGURE 2 below:

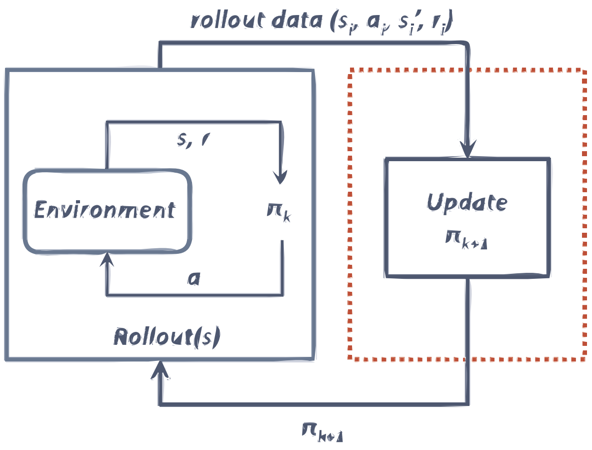

FIGURE 2. On-policy RL algorithm.

FIGURE 2 above, shows the on-policy RL algorithm, where experiences are collected using the latest learned policy. Then, the policy is improved using the collected experience, and the policy π_k is updated using the data collected by π_k itself. This is a sort of an online interaction, where the learning agent interacts with the environment to collect samples. This is done in an attempt to improve the decision-making policy. The current policy, π_k, is then optimized and used to determine what spaces and actions need to be explored. An off-policy RL algorithm is illustrated in FIGURE 3 below.

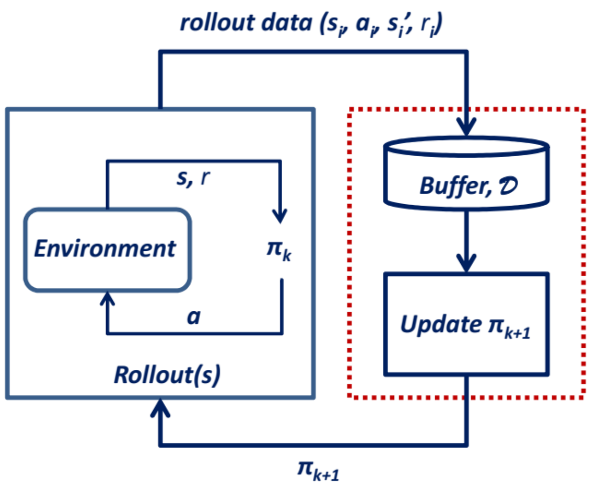

FIGURE 3. Off-policy RL algorithm.

FIGURE 3 above, shows an off-policy RL algorithm, where the learning agent interacts with the environment to collect the samples. This is the classic off-policy setting, where the agent's experiences are appended to the data buffer, i.e., the replay buffer $\mathcal{D}$. Here, each new policy π_k collects additional data such that $\mathcal{D}$ is composed of samples π_0, π_1, π_2, $\cdots$, π_K. All this data is used to train an updated new policy, π_{k+1}. In contrast to the on-policy algorithm, off-policy algorithms seek to improve a policy different from that used to generate the data. An off-line RL algorithm is illustrated in FIGURE 4 below.

An off-line RL algorithm such as the one shown in FIGURE 4 uses previously collected data, without any additional online data collection. The learning agent no longer has the ability to interact with the environment and collect additional transitions using the behavior policy. As a result, the learning algorithm is provided with a static data set of fixed interactions, $\mathcal{D}$, and must learn the best policy it can using this data set. This kind of offline behavioral policy is most used in deep Q-learning networks (DQNs) for in a process known as experience replay, discussed in **Section VI-B1**. Here at each time step, the agent's decision-making procedure is characterized by a policy, θ, as follows:

$$\pi(s, a; \theta) = Pr\{s_t = s|a_t = a, \theta\}, \quad \forall s \in \mathcal{S},\ a \in \mathcal{A}, \tag{9}$$

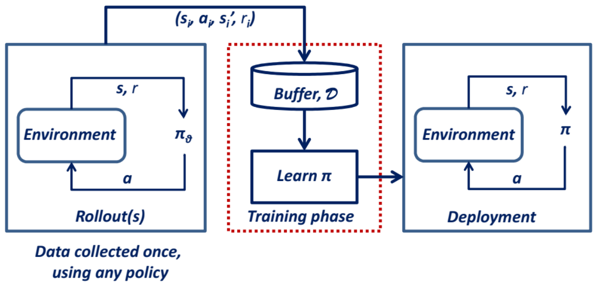

FIGURE 4. **Off-line RL algorithm.**

where the term $\theta \in \mathcal{R}$ is the parameter vector with π assumed to be differentiable with respect to θ, the differential $\frac{\partial \pi(s,a)}{\partial \theta}$ exists [60]. The expression $Pr\{\cdot|\cdot, \theta\}$ is the transition probability defining the evolution of the system from the current state to the next. The system evolves sequentially between the different states in a random way, with the current state and control action fully determining the probability of proceeding from one state to the next. The problems usually studied using model-based RL are special in such a way that the problem has more than one objective such that the controller has to minimize one objective subject to constraints on the others. These are discussed in the cost criteria section **IV-F5** below. The fundamental difference between model-free and model-based RL algorithms comes with the randomization process employed. Thus, knowing the outcome of the randomization process becomes useful to the controller, but there are some cases whereby some system parameters such as the transition probabilities are completely unknown. In such cases, the controller can only estimate these parameters and improve system control by accumulating more information on the evolution of the system.

F. RANDOMIZATION AND FEEDBACK CLASSES

The efficient extensions of the foundational algorithms differ mainly in the way in which feedback from the environment is utilized to speed up the learning process. Mostly, the way in which they concentrate on relevant parts of the problem, and for both model-based and model-free settings these efficient extensions have shown useful in scaling up to larger problems. Depending on the dictates of the problem, the decision-making procedure can follow any of the randomization processes, i.e., the variants of MDPs.

1) MARKOV DECISION PROCESSES

Markov decision processes (MDPs) are controlled Markov chains constituting the basic framework that dynamically controls systems that evolve stochastically. MDPs can be successfully used in modeling RL problems such as stochastic planning problems, game playing, as well as robotic control. Hence, MDPs are the *de facto* standard formalism in the learning of sequential decision-making [64]. The formal framework of MDPs can be defined synonymous with the definitions of the value functions V and the policy π. Thus, MDPs can be defined as a generalization of non-controlled Markov chains since many useful properties of Markov chains carry over to controlled Markov chains. The key property of MDPs is that: conditioned on the state and action at any given time t, the previous states and the next one are independent of each other. Therefore, there are two ways of formulating the agent's objectives using MDPs, i.e., (i) the average reward function in which policies are ranked is according to their long-term expected reward per time step, and (ii) the start-state formulation where the objective is on the long-term reward obtained from the designated state.

2) PARTIALLY OBSERVABLE MARKOV DECISION PROCESSES

These are a combination of MDPs and hidden Markov models. Partially observable MDPs (POMDPs) are used in the modeling of system dynamics connecting unobservable system states to observations. In this case, an agent can perform actions that affect the system and cause its state to change, with the objective of maximizing the reward that depends on the sequence of system states and the agent's actions. In this case, the reward is usually a discounted reward as shown in (6) above. However, an agent is not able to observe the system state directly, but at any discrete time point it can only make observations depending on the state in order to form a *belief state*. A *belief state* can be defined as the situation that the system believes it is currently in. The *belief state* can thus be expressed as the probability distribution over the system states and the solution of the POMDP is a policy prescribing which action is optimal for each belief state.

3) MULTI-AGENT MARKOV DECISION PROCESSES

Multi-agent MDPs (MMDPs) are used in multi-agent planning, where it is typically assumed that there are heterogeneous agents, each with its own set of actions for a given task to be solved. While, generally, each agent in the system might have its own goals, it is assumed that the problem is fully cooperative [65]. Thus, the system utility for each particular state is assumed to be the same for all agents. However, in uncertain and general utility models such a problem can be modeled as an MMDP in which a chosen action at any state consists of individual action components performed by all the agents. After the MMDP has been fully defined, a useful framework is proposed in which the coordination mechanism can be studied. In this case, MMDPs can be considered to be a general case of stochastic games. Stochastic games are generally defined as repeated interactions between several participants in which the underlying environmental state changes stochastically, and depends on the decisions of each participant [66].

4) DECENTRALIZED PARTIALLY OBSERVABLE MARKOV DECISION PROCESSES

Decentralized partially observable MDPs (Dec-POMDPs) are an extension of the POMDP framework and a specific case of partially observable stochastic games. Here, the agents are usually built for complex models, such as those considering system uncertainty. Dec-POMDPs have strong advantages over system's uncertainties which are common in different domains such as in autonomous navigation, inventory management, wireless sensor networks (WSNs), as well as in e-commerce [69]. Problems are modeled using Dec-POMDPs when the choices have to be made in a decentralized way as dictated by the decision makers. While the Dec-POMDP model offers rich frameworks for cooperative sequential decision-making under uncertainty, the real challenge then becomes the computational complexity of the model.

5) THE COST CRITERIA AND THE CONSTRAINED PROBLEM

The cost criteria, also known as constrained MDPs (CMDPs), is a class of MDPs that usually arise in situations whereby the controller contains more than a single objective [70]. The cost criteria approach uses value functions and discounts in order to evaluate the desirability of choosing different transmission parameters. This enables efficient allocation of spectrum and transmission powers that maximize the long-term reward in a cost-effective manner. A more precise concept of the cost criteria is defined using the tuple

$$\{\mathcal{S}, \mathcal{A}, \mathcal{P}, c, d\}, \quad \text{where} \quad c : \mathcal{K} \rightarrow \mathcal{R} \tag{10}$$

is the immediate cost related to the cost function that has to be minimized, and $d : \mathcal{K} \rightarrow \mathcal{R}^K$ represents a K-dimensional vector of immediate costs related to a set of K constraints. This kind of MDP is frequently used in applications of control in Markov decision chains with finite state spaces. In their application, the MDP is used to formulate stochastic optimization problems that need to be solved online using a RL algorithm. Here, the general idea is that: for any strategy π and initial distributions $\eth$, the finite horizon cost can be defined as follows:

$$C^T(\eth, \pi) = \sum_{t=1}^{T} \mathbb{E}_{\eth}^{\pi} c(s_t, a_t), \tag{11}$$

where T represents the finite horizon, and the term $\mathbb{E}$ is the mathematical expectation. An alternative cost that gives less importance to the long-term rewards is called the discounted cost, such that for a fixed discount factor γ^t, it is defined as follows:

$$C_{\gamma}^T(\eth, \pi) = (1 - \gamma^t) \sum_{t=1}^{T} \alpha^{t-1} \mathbb{E}_{\eth}^{\pi} c(s_t, a_t), \tag{12}$$

then

$$C_{\gamma}(\eth, \pi) = \bar{\lim}_{T \rightarrow \infty} C_{\gamma}^T(\eth, \pi). \tag{13}$$

However, due to the existence of finitely many states and actions, the limit, i.e., $\bar{\lim}$, indeed exists such that (13) becomes

$$C_{\gamma}(\eth, \pi) = (1 - \gamma^t) \sum_{t=1}^{T} \gamma^{t-1} \mathbb{E}_{\eth}^{\pi} c(s_t, a_t), \tag{14}$$

and the expected average cost can be defined as follows:

$$C_{ea}^T(\eth, \pi) = \sum_{t=1}^{T} \mathbb{E}_{\eth}^{\pi} c(s_t, a_t) = \bar{\lim}_{T \rightarrow \infty} C_{ea}^T(\eth, \pi), \tag{15}$$

where the left hand-side represents the finite horizon and the right hand-side represents the infinite horizon. Then, letting $C(\eth, \pi)$ to stand for any of the above average costs, and $C(\pi) : X \rightarrow \mathbb{R}$ will represent a vector whose state, s_t, entry is $C(s, \pi)$. Then, the cost function that is related to immediate cost d is defined in a similar way. For example, the finite horizon cost that is related to d_k, $k = 1, 2, \cdots, K$, is defined as follows:

$$D^{T,k}(\eth, \pi) = \sum_{t=1}^{T} \mathbb{E}_{\eth}^{\pi} d_k(s_t, a_t). \tag{16}$$

Then, for a K-dimensional vector of real numbers, the constrained problem can be defined as one of finding a policy that minimizes $C(\eth, \pi)$, subject to constraint $D(\eth, \pi) \leq V$. It should be noted that $C(\eth, \pi)$ and $D(\eth, \pi)$ stand for one of the expected costs defined above [67].

G. APPLICATION OF REINFORCEMENT LEARNING IN SPECTRUM MANAGEMENT

The application of RL strategies in spectrum management is problem specific, and this section will categorically review outstanding RL contributions in traditional wireless networks, CRNs, and network slicing. In spectrum management problems, more especially CRNs, the most important thing is the accurate estimation of the channel state information (CSI). This is because the accuracy of the CSI has a direct impact on the performance of channel training and estimation algorithms. Channel training in AI-based CR systems helps avoid the estimation of large dimension channel matrices by flexibly exploiting advanced signal processing techniques such as signal subspace estimation [68]. A typical application of the RL strategy in mobile and wireless networks that uses channel estimation to perform RA using Q-learning and MDPs is illustrated in FIGURE 5 below.

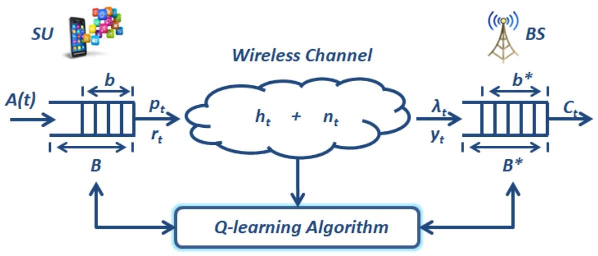

FIGURE 5. An exemplary application of RL and Q-learning concepts in spectrum management.

FIGURE 5 above shows a generic application of the RL strategy in power management where mobile devices are transmitting over a dedicated wireless channel [71]. Here, the system consists of a source queue of finite length B, i.e., each mobile device has a queue associated with it, and a destination queue of finite length B^*, i.e., at the BS. At the source, mobile devices can select parameters contemplated for maximizing transmission throughput without upsetting the power consumption. Assuming a non-saturated source of users [72], user behavior is represented using the buffer state, b, channel state, h_t, and the allocated transmission power, p_t, as well as the transmission rate, r_t. Then, the power management decision parameters at the BS include the channel output, y_t, the arrival process defined in terms of the arrival rate, λ_t, the effective capacity, C_t, defined by the buffer state, b^*. In this application, the environment can be defined as the laws of physics that govern the communication through electromagnetic waves, and the protocols of the wireless environment that process the actions and determines their consequences, i.e., rewards or penalties [73]. The instantaneous configurations that the mobile devices can find themselves in are predominantly the channel conditions which then significantly affect their throughput, i.e., their rewards. Thus, it is assumed that a Q-learning algorithm is the agent operating on both the source and destination and gives feedback and updates about the environment. Since some of the environmental parameters may not be observable, the channel estimation relies on a basis expansion model of the channel which reduces the number of parameters to be estimated. Then depending on the buffer state, b^*, at the BS, the agent has to allocate resources to the different mobile users, an action, a_t, which will be a single move out of a set of possible actions, $\mathcal{A}$, that the agent can take at a given instant, t. Such an action would include, among others, transmission admission/rejection/termination, cell handover, or even channel handoff in terms of CRNs [74]. Therefore, a transmission action executed in the current channel conditions warrants some reward, r_t, in the form of achievable throughput, which takes the environmental observation to the next state, s_{t+1}.

There are several research works that considered the application of RL strategies in solving problems in wireless networking. One of the applications of MDPs was in wireless sensor networks (WSNs), where the objective was to obtain energy-efficient sensor alternatives for data exchange and gathering in cooperative multi-hop WSNs using data aggregation. Here, the transmission delay, expected network congestion, and energy consumption were the performance metrics used in decision-making. In WSNs, a data query is used to disseminate commands, i.e., queries, from the BS to the intended sensor nodes for them to retrieve their readings. In [75], a probabilistic scheme that selects a set of wireless sensor nodes that should answer to user queries was proposed. Here, the authors formulated the problem as a parametric partially observable MDP where the optimization metric was the average long-term rewards. The system state was formulated using a vector consisting of the data attributes for each sensor node. The sensor nodes would then choose actions between either answering or refraining from answering queries. A random access problem in a single cell was considered from a stochastic game perspective in [76]. The transmission and reception were formulated using multiple uplink transmitters sharing a single slotted and synchronous classical collision channel and a single receiver. The state space of the system was defined as the number of packets that are waiting at each transmit buffer, or the holding time, that is the length of time that each packet has been in the transmission buffer. The reward/cost was derived based on the buffer holding cost, which is the number of time slots that the transmitted packet spent in the buffer, and the throughput, which is the number of packets that were successfully transmitted. In their work, both selfish and cooperative users were allowed to select retransmission strategy-based performance requirements such as throughput, delay and transmission costs. The system transition had a stochastic component that captured the arrival process, which is the number of packets arriving within a certain time slot. In [77], stochastic games were approached from the queuing theory perspective where mobile users requiring service had to choose on whether they want to be served by a private, but slow service provider or by a public but powerful one. The state space of the system consisted of the current system load for both service providers, and the cost is the time required to service each mobile user. This application of stochastic games provides useful insights into achieving wireless Big Data solutions that can be useful in analyzing specific situations and suggesting proper behavior to the participants.

The application of the cost-constrained problem in wireless communications is usually when there are costs that need to be evaluated. For example, a contribution in [78] considered a problem where two types of costs can be incurred; the buffer overflow cost as well as the packet holding cost. Here, the primary objective of the cost-constrained problem was on minimizing the expected average operating costs, subject to the expected average holding cost. Another generalization of this application includes the expected average and the discounted costs studied in [79], where the existence of an optimal constrained randomized stationary policy for which the two stationary policies differ on at most one state. The approach used in [79] has many advantages in wireless communications, because: (i) there is no requirement of any a priori knowledge on channel statistics and traffic arrival distribution in order to determine the jointly optimal power-control, adaptive modulation and coding, as well as the policies regarding power management policies; (ii) as opposed to conventional RL strategies, it exploits partial system information, so, there is less information that require to be learned; and (iii) it also obviates the action exploration requirement, which severely limits the adaptation speed and run-time performance of conventional RL algorithms.

Also, a call control algorithm in which communication links have variable capacity that supports multiple service classes was proposed in [80]. In the study, the authors modeled the problem as a CMDP and used an RL algorithm with the use of Lagrangian and state aggregation approaches to solve it. The novelty of this approach is that it could control class-level QoS in terms of both call blocking and dropping probabilities. Similar problems that were solved using this kind of treatment include packet communication systems with reject options and single-server queue with service rate control such as in [78], a stochastic optimization problem for joint data admission control-power allocation to maximize the throughput such as in [81]. A fully autonomous and distributed RL scheme was applied to solve a user association problem in a congested integrated access backhaul (IAB) network in [82]. An instantaneous load-based bandwidth partitioning was proposed where a cognitive engine would judiciously determine network congestion before initiating a bandwidth split. Here, the effect of network congestion rate was learned using Q-tables, which was determined in terms of the learning rate and discount factor. Then, a reactive load balancing technique that uses an exponentially weighted moving average was applied to track the SBS load.

1) COGNITIVE RADIO NETWORKS

A channel sensing order problem for a cognitive multi-channel network was investigated [85]. Since the brute-force algorithm used to find the optimal sensing order requires great computational effort, the authors proposed an RL strategy to dynamically search for the optimal sensing order. One key feature of the proposed algorithm relates to its adaptability to changes in channel characteristics by learning from past actions. This feature makes the mechanism immune to possible changes in channel availabilities due to changes in PU activity patterns and channel qualities. The performance of the proposed algorithm was evaluated through simulations and compared with other schemes. The results obtained are close to the optimal value provided by the brute-force algorithm, and showed superiority to the other schemes. A priority-based multi-agent system for spectrum allocation using a reserved allocation method was proposed for spectrum allocation in [86]. Here, the proposed scheme gathers environmental information to be used in spectrum sharing decision making. In order to achieve this, multiple swarm agents were deployed in every node with the objective of acquiring and storing radio data. The primary task of each swarm agent was to use the gathered information for judging proper channel allocation. The results reported in this contribution show that the repository system enables the system to achieve maximized spectrum utilization. A Q-learning-based dynamic optimal band and channel selection scheme was proposed in [87]. The authors considered the environmental state and the system demands as the state and then select suitable channels for the required data rate. The simulation results confirm that the system can dynamically select bands and channels that maximize the available transmission time and capacity according to the desired performance. In another contribution, the authors in [88] proposed an efficient routing protocol that addresses two design challenges, i.e., (i) transmitting packets via a stable route, and (ii) ensuring that minimal interference is imposed to PUs. A generalized version of the Q-learning algorithm that exploits PU behavior was proposed. The infinite time horizon was divided into periods, then into sub-cycles corresponding to stages. This was based on the assumption that the statistical model parameters of PU activities will not change during each sub-cycle. From the simulation results, it transpired that imposing excessive interference to PUs results from lack of attention to the multi-stage periodic PU behavior. The performance results also confirmed that the proposed routing approach outperforms existing baselines in terms of throughput and minimal interference.

2) RADIO NETWORK SLICING

Addressing the issue of fixed RA mechanisms, which result in low resource utilization and violation of user QoS due to the fluctuation in network demands is a MDP problem. As a result, the authors in [89] proposed a resource management system for network slicing where a dynamic resource adjustment algorithm based on RL from each tenant's point of view was proposed. The resource management was modeled as an MDP, then a Q-learning-based dynamic resource adjustment algorithm was proposed. The proposed algorithm was aimed at maximizing the profit of tenants while ensuring the QoS requirements of end-users. The simulation results obtained demonstrated that the dynamic resource adjustment algorithm significantly increased the profit of tenants compared to existing fixed RA methods while satisfying the QoS requirements of end-users. In another contribution in [90], the authors proposed an admission control algorithm for infrastructure providers to consider the service level agreements (SLA) of the different tenants, as well as their traffic usage and user distribution. This was done to address the problem of isolation between network slices and the allocation of resources across them. In order to enhance the overall process by means of learning using an RL technique that considers heterogeneous mobility and traffic models among diverse slices, the authors designed the following building blocks: (i) traffic and user mobility analysis, (ii) a learning and forecasting scheme per slice, iii) optimal admission control decisions based on spatial and traffic information, and iv) a reinforcement process to drive the system towards optimal states. The results obtained through simulations indicate that the proposed approach provides substantial potential gains in terms of system utilization while not violating the tenants' SLA by relying on appropriately tuned schemes.

The authors in [91] proposed a constrained RL-based approach in order to address the lack of accurate resource orchestration models and hidden problem structures. The RA problem was focused on resource virtualization and the problem was formulated as a CMDP, subject to both cumulative and instantaneous constraints. Then, an adaptive constrained RL algorithm based on interior-point policy optimization was employed to deal with cumulative constraints. The instantaneous constraints were handled by projecting the RA decision generated by the RL algorithm to its nearest feasible decision. The performance evaluation of the proposed framework proved its effectiveness in RA and outperforms other

TABLE 4. Application of reinforcement learning techniques in spectrum management.

Application	Objective(s)	State(s)	Action(s)	Algorithms(s)	Refs
Wireless network	Minimizing transmit power	Channel & traffic statistics	Power & rate allocation	Q-learning algorithm	[67]
	Data acquisition, raise confidence levels	Node attribute values	Query, no query	Stochastic gradient descent	[75]
	Power & rate adaptation	Buffer occupancy, SIR, data rate	Re-transmission strategy selection	Constrained MDP, Q-learning	[76]
	Power control, overflow costs	Channel & traffic statistics	Power management	Constrained MDP, Post-decision state	[78]
	Improving class-level QoS	Link capacity, class arrival frequency	Request admission decision	Constrained MDP, Q-learning	[80]
	Power control, maximizing throughput	CSI, data arrival, queue length	Power management	Effective block error rates	[81]
	Congestion control, maximizing throughput	Number of users, QoS requirements	Request admission decision	Q-learning	[82]
	End-to-end learning, minimize convergence time	SNR in AWGN and RBF channels	Transmitter normalization	Q-learning algorithm	[83]
	Policy learning, minimize average slowdown	Resource demand	Scheduling	Q-learning algorithm	[154]
	Traffic routing, minimize E2E delay	Packet arrivals, queue length	Send, wait, route selection	Q-learning algorithm	[84]
Cognitive radio	Time scheduling, maximizing throughput	Channel state information, energy state	Optimal scheduling policy	MDPs, Q-learning	[152]
	Optimal sensing order	Sensing order and sensed channel position	Adjust and update Q-table	Q-learning	[85]
	Maximizing spectrum utilization	Channel occupancy, capacity of repository	Gather information, allocate channels	Multi-armed Bandit reinforcement learning	[86]
	Maximizing capacity & transmission time	Channel state, SU demands	Select best channels	Q-learning	[87]
	Improving the routing protocol	Physical neighbors, data flows	Connect flow to best path	Generalized Q-learning	[88]
Network slicing	Priority-based slicing, minimizing scheduling delay	Resource availability, resource demand	Resource scheduling	Q-learning	[151]
	Tenants' profit maximization	Number of tenants QoS requirements	Dynamic resource adjustment	MDPs, Q-learning	[89]
	Optimal admission control	Traffic usage, user distribution	Admission control decision	MDPs, Q-learning	[90]
	Resource virtualization	User types, traffic demand	Bandwidth allocation decision	CMDP, adaptive constrained RL	[91]
	Slice virtualization, high revenue returns	Slice requests, number of required resources	Slice admission and setup	Constrained RL	[92]

baselines in terms of service demand guarantees. Another contribution in [92] proposed a slice admission strategy based on the RL strategy in the presence of services with different priorities. Considered here were slices from different mobile service providers, virtualized over the same RAN. The authors proposed a RL-based admission policy that learns the services with the potential of bringing high revenue returns and low SLA violation penalties. The policy was able to adapt to different conditions in terms of SLA penalties and priority services. In the performance evaluation, the proposed policy was compared with two deterministic policies, which it outperformed by a 55% margin. A summary of RL algorithms used in spectrum management is tabulated in TABLE 4 below.

H. THE PROBLEM WITH REINFORCEMENT LEARNING

Wireless networks are controlled through feedback signals in order to avoid instability and malfunctioning, which can be a challenge in distributed spectrum management. The main challenge with distributed spectrum management is the simultaneous handling of wireless parameters such as the operating frequency band, the employed symbol modulation, and the coding rate to ensure an optimal wireless network operating point that is both energy-efficient and does not cause harm to PUs [93]. In a distributed CR environment, different network entities are required to make local and autonomous decisions such as spectrum access, channel allocation, and power control in order to achieve different objectives. Such objectives include, but not limited to, throughput maximization, latency, and energy consumption minimization. This kind of decision-making problems for CR systems could best be modeled using either the MAB or the MDP techniques. Using these techniques, the problems could be solved using either RL or DRL strategies.

However, even though RL strategies offer lightweight solutions to these problems, they usually exhibit poor convergence by taking longer to reach the optimal policy. Due to the fact that future generations of wireless networks will be larger and more complex, operating in more distributed and diverse environments, exploring and gaining knowledge of the entire system using RL strategies could be time-consuming. Due to this complexity, the time factor is the one that will increase the network latency. Also, the increase in dimensionality and computational complexity, problems involving network control will become very difficult to handle and result in high energy consumption. As a result, the dynamic and uncertainty of the network status, the coexistence and coupling among different wireless devices with disparate requirements makes the manageable range of RL to suffer from the curse of dimensionality. Thus, deep architectures such as DL open a new era for the extension and development of RL into DRL in order to address the curse of dimensionality [94].

V. INTRODUCTION TO DEEP ARCHITECTURES

The past decade has seen escalating interest in deep representations of DL and DRL. This increased interest has resulted in the emergence of new applications incorporating hierarchies of convolutional neural networks (CNNs), long short-term memory (LSTM), as well as auto-encoders [95]. In terms of DRL, dueling network representations manifested by exploiting the successes in RL and DL architectures in the form of deep Q-learning networks (DQNs) and double DQNs (DDQNs) [155]. Deep architectures can be defined as hierarchical structures that combine DL and RL into a hierarchy of functions with the objective of attaining the knowledge to be used in learning a hierarchical decomposition of spatial environments. In the context of deep architectures, DRL, DQNs and DDQNs are a step ahead of all AI strategies in terms of achieving better decision-making. Due to their capability of storing useful information for future replay makes them suitable for long-term planning, see Section **VI-B1**. Thus, integrating deep architectures into distributed environments like CRNs will not only impact the CR transmission technologies in terms of protecting PUs from excessive interference, but will also pose a significant impact on the way in which wireless networks are controlled. Deep architectures will enhance the coordination of CR devices and also improve the accuracy of spectrum management in vertically integrated networks. Due to the dependence of deep architectures on DL, a discussion of DL is required in order to build a solid foundation.

A. THE DEEP LEARNING TECHNIQUE

The DL technique is the most prevalent AI strategy consisting of networks that are capable of learning, unsupervised, from data that is either unstructured or unlabeled through a hierarchy of NNs. A block diagram comparison of the DL concept and its relationship with traditional ML is illustrated in FIGURE 6 below.

As illustrated in FIGURE 6, contrary to traditional ML techniques, DL consists of a DNN block as the only computational block between the input and the output. ML has an additional learning rule that performs feature selection prior to the DNN computational block. In the case of supervised ML, DL can be applied to eliminate the feature engineering step. Here, the DL technique translates the input data into compact intermediate representations akin to principal components, and derive layered structures that remove representation redundancy.

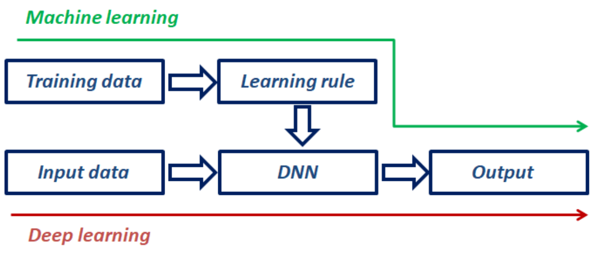

FIGURE 6. Comparison of the DL and ML concepts, supervised ML at the top and DL at the bottom.

Contrary to ML methods, the hierarchical function of a DL system allows AI systems to process data using a non-linear approach by allowing each computation progress through the computational process of a hierarchy of its layers. In a DL technique, the hierarchy of NNs or multi-layer NNs form a DNN, hence the *deep* qualifier. Thus, a NN consisting of two or more hidden layers is considered a DNN [96]. Since the computational DNN for a ML technique usually consists of a single hidden layer, it is considered to be shallow [97]. Another class of artificial neural networks (ANNs) used in DL techniques is the recurrent neural network (RNN), which uses its internal states to process sequences of inputs. The RNN has memory that is able to capture the hidden layer outputs from the previous time step; as a result, the outputs of each layer are determined by both its current inputs and the hidden states of the previous time step. Thus, the RNN is dependent on time for over longer time periods, which makes it suitable for signal compression and reconstruction as well as time-series problems. Such a DL architecture has achieved great performance in the areas of supervised, unsupervised and reinforcement learning, but there are still some largely untapped potential in solving mobile and wireless networking problems. The technical details of DL models that are discussed in the sequel are provided for enthusiastic readers who seek deeper understanding of DL architectures.

B. TRAINING A DEEP LEARNING MODEL

DL models are trained to use NNs of a hierarchy of NNs to accurately learn the mapping function of inputs and outputs. The accuracy of DL techniques is achieved by updating the weights of the NN in response to the errors that the model makes during the training process. Updates are then made by adjusting the weights in order to continually reduce the errors until an acceptable model is achieved. This process is the most challenging part of using DL algorithms and is the most challenging and time consuming. This process of a DL model is shown using a convolutional NN (CNN) structure in FIGURE 7 below.

As shown in FIGURE 7, the principle of inference which is the feed-forward/forward passing, and learning which is the backward propagation (BP) processes on an NN are illustrated. Here, a two-dimensional (2D) CNN has been

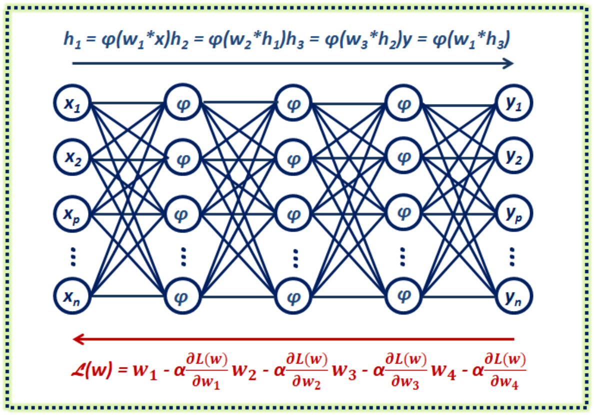
$h_1 = \varphi(w_1 * x) h_2 = \varphi(w_2 * h_1) h_3 = \varphi(w_3 * h_2) y = \varphi(w_1 * h_3)$
x_1
x_2
x_p
x_n
φ
y_1
y_2
y_p
y_n
$\mathcal{L}(w) = w_1 - \alpha \frac{\partial L(w)}{\partial w_1} w_2 - \alpha \frac{\partial L(w)}{\partial w_2} w_3 - \alpha \frac{\partial L(w)}{\partial w_3} w_4 - \alpha \frac{\partial L(w)}{\partial w_4}$

FIGURE 7. **Feedforward/forward passing (i.e., inference) and backward propagation (i.e., learning) processing in a CNN.**

described in mathematical terms, where the weights are learned by minimizing a loss function $\mathcal{L}(w)$ using a gradient descent (GD) method through chain rule and BP. The problem of synthesizing NNs that can serve as the state and the output feedback laws to achieve control objectives that can be specified as reachability of a target set. The reachability is defined as the region of stability for the target set around an equilibrium point. Temporal logic is used in writing the logical formulas that are applied in training the weights of the feedforward-feedback network shown in FIGURE 7 above from training samples. The loss function being computed is learned using stochastic gradient descent (SGD) algorithm, which is discussed below.

1) STOCHASTIC GRADIENT DESCENT

The stochastic gradient descent (SGD) algorithm is the most prevalent and efficient technique used in the training of DL models such as DNNs and other non-convex models. As a result, the theoretical properties of SGD are well understood for the optimization of both convex and non-convex objective functions. However, in the latter, it can be related to other assumptions on objective functions such as the error-bound conditions and the Polyak-Lojasiewicz conditions [98]. However, there exists a huge gap between the theoretical understanding of SGD algorithms and their promising practical behavior in the non-convex learning settings, which is exemplified in the setting of training highly non-convex DNNs. Despite that, due to its low computational complexity per iteration, it is also used in solving optimization problems in a variety of ways, including traditional ML problems as well as those applied in signal processing. In each SGD iteration, a gradient is first computed based on a randomly selected sample, then model parameters are updated along the negative gradient direction of the current iterate - a process called BP. The training of a DL model can be viewed as a feed-forward and feedback control process known as backward propagation, as discussed in Sections **V-B2** and **V-B3** below.

2) FEEDFORWARD PROPAGATION

During the feedforward or forward propagation process, the input data are fed in the forward direction of the NN, and in the progression each hidden layer accepts that data and processes it as per the activation function and passes it to the next layer. This process is carried out until the output is obtained at the output layer [99]. With reference from FIGURE 7, the input layer accepts the inputs $\mathbf{x} = \{x_1, x_2, \cdots, x_n\}$, processes it through the operation $h_1 = \phi(w_1 * x)$, where $\phi(\cdot)$ represents a non-linear transformation - an activation function, the term w_1 denotes the weight index of the weights of the first hidden layer, and h_1 represents the output of the first hidden layer [100]. The output h_1 is passed over to the next hidden layer in the hierarchy, which also applies the same non-linear transformation to learn and create the next statistical model as output h_2. The iterations continue to the third layer to learn and create h_3, as the output of the third hidden layer, until the output has reached an acceptable level of accuracy, where the output $\mathbf{y} = \{y_1, y_2, \cdots, y_n\}$ is produced at the output layer. In a CNN with more hidden layers, this is subsequently fed as input to the next hidden layer, proceeding layer by layer, until the output y is achieved at the end. The feedforward propagation process shown in FIGURE 7 is as follows:

$$\begin{aligned} h_1 &= \phi(w_1 * x); \quad h_2 = \phi(w_2 * h_1); \\ h_3 &= \phi(w_3 * h_2); \quad y = \phi(w_4 * h_3), \end{aligned} \tag{17}$$

where the expression $(\cdot * \cdot)$ denotes a convolution operation, and y is the output.

3) BACKWARD PROPAGATION

With reference to the top of FIGURE 7 above once again, the bottom of the figure presents the opposite of what is discussed above - the backward propagation (BP) process. The BP algorithm is a supervised learning technique that is based on the GD method whose objective is to minimize the error of the DNN by moving the computation down the gradient of the error curve [101]. This algorithm is a special case of the chain rule of derivation, where α denotes the learning rate, the asterisk $*$ represents a convolution operation - all used to obtain the gradients $\partial \mathcal{L}/\partial w$ in each hidden layer. Therefore, in order to train the CNN approximately, the loss function $\mathcal{L}(w)$ is used as a measure of the distance that the output y is from the actual ground truth y^*. This operation concludes the objective of training, which is to obtain the best weights $\mathbf{w}$ that will minimize the loss function as follows:

$$\begin{aligned} \mathcal{L}(w) = w_1 - \alpha \frac{\partial L(w)}{\partial w_1} w_2 - \alpha \frac{\partial L(w)}{\partial w_2} w_3 \\ - \alpha \frac{\partial L(w)}{\partial w_3} w_4 - \alpha \frac{\partial L(w)}{\partial w_4}. \end{aligned} \tag{18}$$

Therefore, the training objective is based on obtaining the best weights $\mathbf{w}$ that minimizes the loss function $\mathcal{L}(w)$ and the parameters of the DL model are learned using SGD with mini batches. The loss function is learned using and during the BP algorithm over the weights of the last hidden layer, and also updates the weight through the following computation:

$$w_4 = w_4 - \alpha \frac{\partial \mathcal{L}(w)}{\partial w_4}. \tag{19}$$

The same computational operation is performed for each weight in the direction indicated by the gradient following the chain rule until the gradient descent eventually leads to a set $\mathbf{w}$ that minimizes the loss function $\mathcal{L}(w)$ [99].

C. APPLICATION OF DEEP LEARNING MODELS IN SPECTRUM MANAGEMENT

Recent breakthroughs in DL over the past few years have been met with high enthusiasm by the telecommunications research community. As the most important enabling strategy for AI, the DL technique has had successful applications in areas such as medical diagnosis, speech recognition, natural language processing, computer vision, communication network domains, as well as search engines, to name just the prominent ones. Because of their computational prowess, DL techniques have gained popularity across different industries where they are used to perform a plethora of complex tasks. Examples of these complex tasks where DL has attained remarkable achievements include computer vision, where it is difficult to characterize real world images and also in wireless channels with rigid mathematical models [102]. However, all these are data-driven approaches. The application of DL strategies in wireless communications require features with specific characteristics in order for them to function properly. In as much as DL strategies are predominantly data-driven, in wireless networking problems the data-driven approaches cannot work in isolation, without complimenting traditional design techniques. Thus, the need for signal processing techniques, such as mathematical programming and nature-inspired techniques, whose objective is to prepare proper input dataset, compatible with the ML algorithm requirements, will always be there.

In mobile and wireless networking, DL techniques equips mobile devices with the capability of gathering the information required in the configuration of their parameters for rapid adaptation in different environments. Since applications in spectrum access require the imposition of non-trivial boundedness assumptions on the gradients at all the iterates encountered during the learning process, which however depend on the realization of the process being optimized. Answering the design questions in telecommunications, wireless communications in particular, requires careful analysis and implementation that attends to the details of real-world scenarios such as interference levels, spectrum access, and spectrum/channel handoffs. In terms of spectrum access, it is a goal-directed phenomenon that requires DL techniques to better perform service classification to enable for accurate prediction.

1) THE MULTI-LAYER PERCEPTRON

The multi-layer perceptron (MLP) is the simplest of all DL structures that has found applications in communication systems. A simple MLP consists of at least three layers, with computational units densely connected in each layer, hence requiring the configuration of a substantial number of weights [101]. It should however be noted that it is only MLPs with more than a single hidden layer that qualify to be regarded as DL structures. As discussed in the above section, the BP algorithm has become the most popular and effective learning model for multi-layered networks such as MLPs. Given an input vector $\mathbf{x}$, a standard MLP can perform the operation given in (2) with the objective of improving the non-linearity of the DL model. The versatility of the MLP allows it to be employed for all ML purposes, even though it has high implementation complexity and low convergence efficiency. In order to train a MLP, the set of parameters that need to be learned are in the set $\mathbf{w} = \{W, b\}$, where the BP algorithm is used to perform parallel training for improving the model efficiency. MLPs are widely in DL problems, either as a baseline or are integrated into more complex architectures. MLPs built to be integrated into complex structures can be employed in assisting in feature extraction models built for specific objectives in mobile and wireless networks such as classification or clustering. One example of this integration is in the final layer of a CNN that is used for classification purposes, and such an architecture can potentially be explored for analyzing continuously changing mobile environments.

A series of effective schemes have been developed using MLP structures to solve different problems and overarching have been reported. A key generation and certification technique using MLP in wireless communication of data/information was proposed in [103]. In their proposed technique, both the sender and receiver used identical MLPs, and both perceptrons began the synchronization process by exchanging control frames. During the synchronization stage, messages for integrity testing on synchronization were carried out to obtain synchronized identical weight vectors. After this initial procedure, the identical and synchronized weight vectors then form the key for encryption/decryption. In [104], a complex MLP model for multi-user detection in space division multiple access-orthogonal frequency division multiplexing (SDMA-OFDM) systems was proposed to perform channel approximation and multi-user detection simultaneously. Here, the authors proposed a technique for minimizing the high computational complexity of classical techniques such as maximum likelihood. Such classical techniques are known to suffer from high computational complexity, while the loss function, usually in the form of an MMSE yields poor performance results as the number of users begin exceeding the number of receiving antennas. The authors in [105] applied the Levenberg Marquardt training algorithm on MLPs to be used for mobile positioning, which was supposed to be compliant with the Global System for Mobile (GSM) network operations in urban environments. Here, the received signal strength was used to evaluate the accuracy, the cost, the reliability and coverage, which were the key performance metrics. Real data obtained from field measurements was used in evaluating the performance of this technique, and the results obtained met the practical positioning defined by the Federal Communication Commission (FCC) accuracy requirements.

A non-intrusional medical quality of experience (QoE) prediction model was proposed in [106]. This was a QoS-aware, content-aware, and device-aware model that uses MLPs to act as a platform for maintaining and optimizing diagnostic quality through a device-aware adaptive video-streaming mechanism. This application was designed for small cell network deployments and its training involved the use of unseen data sets consisting of input variables such as QoS, content features and display device characteristics. To validate the efficiency of this model, subjective tests were carried out by medical experts in the form of a medical QoE which was measured using the mean opinion score (MOS). The prediction accuracy of this model was statistically evaluated via correlation coefficient and the root mean squared error (RMSE) and the results showed to be successful in measuring medical QoE closer to the visual perceptions of medical experts. In [107], a secondary user-experience-oriented RA for real-time CRN purposes was proposed. A deep neuroevolution (DNE) technique was proposed to advance DL capabilities by addressing its built-in limitations in dynamic resource allocation. Since the maximization of transmission rates and the minimization of latency are conflicting objectives that require balanced optimization, the phenotypic plasticity of transmission rates and delay constraints inside the MLP were used to achieve a stable learning framework. Using this technique, the stability of dynamic RA and SU satisfaction was achieved when the number of SUs was increased.

2) THE RESTRICTED BOLTZMANN MACHINE AND DEEP BELIEF NETWORKS

The restricted Boltzmann machine (RBM) is typically an energy-based undirected graphical model whose architectural structure consists of one visible layer and one hidden layer. The RBM was originally intended for unsupervised learning purposes and its application in DL problems marks the point where DL meets the physics field of thermodynamics. According to the RBM architecture, which is discussed in [108], each computational unit can only assume binary values, and the probability that a binary state of a visible neuron i is set to 1 can be given as follows:

$$P(h_i = 1|x) = \frac{1}{1 + e^{-W \cdot x} + b_i}, \tag{20}$$

where W represents the weight, h, and x respectively represent the hidden and visible units of the RBM which are assumed to be conditionally independent of each other. When RBMs are stacked together, the resulting architecture is referred to as a deep belief network (DBN), which achieves superior performance in time-series forecasting. DBNs usually consist of probabilistic generative NNs composed of multiple layers of RBMs [109].

The application of this energy-based and the probabilistic model in spectrum management can be found in [110], where the authors studied spectrum occupancy reconstruction by sampling from a Markov random field. With the goal of predicting the latent binary values of spectrum occupancy, the RBMs were effectively trained using the Metropolis-Hastings algorithm through successive episodes of Gibbs sampling. An application of DBNs that consisted of a three-layer stack of RBMs was proposed in [109] for application in time-series prediction. In their contribution, the authors used the deep network of RBMs for capturing the features of the input space of the time-series data. The stack of RBMs was pre-trained using energy functions and in order to fine tune the connection weights between the visible and hidden layers, it was trained using a GD algorithm. A DBN architecture consisting of multiple layers of RBM auto-encoders (AEs) was used for predicting traffic volume in the internet of vehicles (IoVs) in [111]. Here, the time-series data was obtained from roadside units and was used by the three-layer DBN for extracting and learning key input features used for constructing a model for predicting traffic flow. In their contribution, the authors used the deep network of RBMs to broadcast time-critical traffic information in vehicular communications. The proposed architecture was trained using a firefly algorithm in order to optimize the DBN topology and the learning rate parameters.

3) THE LONG SHORT-TERM MEMORY AND ITS VARIANTS

The long short-term memory (LSTM) network is a type of RNN that was designed in order to address the vanishing gradient problems that are associated with conventional RNN architectures such as the auto-regressive integrated moving average (ARIMA) [112]. The ARIMA is another architecture used for time-series purposes, which uses a naive predictor in its predictions. After successfully addressing the vanishing gradient problems of the ARIMA, the LSTM became a specific kind of scheme in DL that effectively overcomes this problem by using three regulators of the flow of information inside the LSTM unit. These three regulators are gates referred to as the input gate, output gate, and forget gate, which help in the learning of long-range dependencies in time-series data. Thus, the learning of long-range dependencies embedded in time-series data, which are an obstacle for most prediction algorithms, are solved using the LSTM network. A single memory cell of the LSTM architecture is illustrated in FIGURE 8 below.

The LSTM cell shown in FIGURE 8 operates in the following way: The input gate, labeled as i^t, takes new input points from the outside and processes them as newly available data. The forget gate, labeled as f^t, is the one that decides when to forget the output results of the previous cell and thus selects the optimal time lag for the incoming input sequence. Then, the output gate, labeled with o^t, takes all the computed results, and generates an output for the LSTM cell. The input gate of the memory cell (*not shown in the figure*) takes input from the output of the LSTM cell in the last iteration. Because of its need for memory, the forget units gives the memory cells the ability to determine the time when certain information

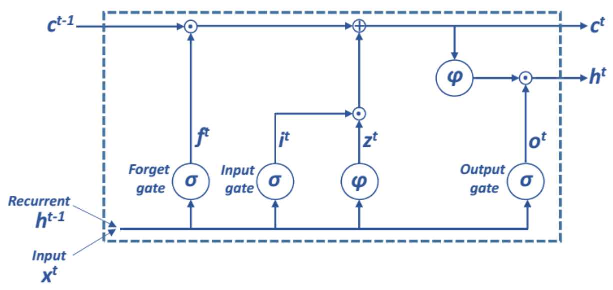

FIGURE 8. A single memory cell of an LSTM architecture.

has to be forgotten, hence determines the optimal time lags. Thus, the LSTM is more effective in the utilization of model parameters in the training of predictive models for large-scale traffic matrix predictions and converges quickly to give state-of-the-art prediction performance.

In modeling the LSTM for time-series applications, the output layer of the LSTM cell is modeled using a linear regression layer. Therefore, if the input time-series, the hidden layer state of the memory cell, and the output time-series are represented as $X = (x_1, x_2, \cdots, x_n)$, $H = (h_i, h_2, \cdots, h_n)$, and $Y = (y_1, y_2, \cdots, y_n)$, respectively. Thus, the computation procedure of an LSTM cell, with the mean squared error (MSE) loss function is given as follows [113]:

$$MSE = \sum_{t=1}^{n}(y_t - p_t)^2, \tag{21}$$

where y_t represents the actual output and the term p_t represents the projected traffic flow prediction. In order to minimize the training error while simultaneously avoiding local minimal points, Adam optimization [114], which is a modification of SGD optimization with adaptive learning rates, is applied for backpropagation through time (BPTT). Therefore, the LSTM structure resolves the decaying error back-flow issue common in many time-series problems where accurate predictions are required and traditional architectures such as MLP fail. Another version of the LSTM is the random connectivity LSTM (RCLSTM) which is based on stochastic connectivity between neurons. Compared to the traditional LSTM, the RCLSTM can achieve significant breakthroughs in the architecture formation of NNs, since it exhibits a certain level of sparsity. This LSTM variant can satisfyingly reduce the computational complexity, hence suitable for latency-stringent applications. However, this improvement demonstrates that its prediction accuracy is comparable to that of the conventional LSTM, no matter how much the number of training samples and the length of input sequences can be changed.

Apart from time-series forecasting, the LSTM network has had extremely successful applications in time-dependent signal processing problems such as speech recognition, joint source-channel coding design, as well as machine translation [115]. As a result, in 2015 it found its first application in vehicular traffic flow prediction, and subsequently in mobile and wireless communication networks for the prediction of data traffic. It was then postulated that even user mobility traces could benefit directly from the capabilities of LSTM network. In [116], it was proposed for use in proactive caching for mobile edge computing (MEC), where the authors applied it to improve the prediction accuracy of content popularity. The motivation of their contribution was due to the fact that the prediction accuracy and content popularity of cached content are typically unknown and tend to vary over time. In [115], an LSTM scheme was used together with a module generator in spectrum prediction for a latency-optimized field-programmable gate array. Compared with the ARIMA, the LSTM scheme obtained superior results in radio frequency spectral prediction. Another contribution in [166] proposed a time-series prediction scheme for data mining, where the RCLSTM model was used in the extraction of useful information records to determine future values of certain parameters.

The authors in [118] proposed a DL scheme for addressing the trade-offs between QoS and energy saving using balanced optimization. Here, the authors used a DL-based computational-resource-aware energy consumption technique that uses an exploration technique of the system state-space and traffic load prediction to come up with a balanced and better trade-off between QoS and energy saving. The performance of the exploration technique was evaluated against a traditional random tree technique, which it outperformed with a 9% margin. In another contribution, the LSTM was used in the design of an online optimization algorithm called energy-aware and adaptive management, abbreviated ENAAM by [119]. The ENAAM algorithm was proposed for energy saving and QoS provisioning by exploiting the short-term traffic load and harvested energy forecasts using an LSTM network. Based on foresight control policies where the BSs and virtual machines (VMs) could be switched ON/OFF dynamically with the objective of saving network energy, this contribution was inspired by the convergence of communication and computing that led to the emergence of the MEC paradigm. With the computing resources, which are supported by VMs and distributed at the edge of the mobile network, the results obtained indicated that the BSs were able to ensure reliable and ultra-low latency services and reduced the network energy consumption.

4) THE AUTO-ENCODER

Auto-encoders (AEs) are actually unsupervised learning techniques that also pose as a supervised learning technique in disguise, while at the same time it is also an unsupervised-*ish* DL technique. AEs were initially designed for unsupervised learning purposes in an attempt to copy the inputs to their corresponding outputs. Their underlying principle of AEs is to learn the compact representation of input data for the purpose of dimensionality reduction [120]. However, they are also as supervised learning techniques since it does have a target value (i.e., original inputs), thus it is considered to have some degree of supervision. On the other hand, it is considered an unsupervised learning technique since the target value is not in addition to the input data. Yet another characteristic of an AE is that it belongs to the family of NNs, but it is also very closely related to principal component analysis (PCA) [121]. Here is how the relationship between the application of AEs in unsupervised learning and DL can be understood: (i) AEs are an unsupervised algorithm that is similar to PCA; (ii) in DL problems, it is used to minimize an objective function similar as in PCA; (iii) it is a NN; and (iv) the target output of a NN is its input. Since an AE in DL problems is used for mapping the original input data from the input layer into a code, so that it can be able to recover the data that closely matches the original input data at the output layer [122], it is a DL technique. In summary, AEs are unsupervised learning in design, but the depth of their architecture renders them DL techniques.

A typical application of an AE in an end-to-end communication system was discussed in [123], where the transmitter communicates with the receiver over a time-varying wireless channel. Here, the AE was modeled as a feed-forward NN with several layers, and the transmitter and receiver were represented using fully connected DNNs that were jointly optimized over an additive white Gaussian noise (AWGN) channel. The time-varying frequency response of the wireless channel was characterized in terms of the time delays, the Doppler frequency shifts, as well as channel gains, all of which were assumed to vary randomly in the modeling. As a result, the channel was represented using a single layer that provides the likelihood transfer function $p(y|x)$ between the transmitter and receiver. Here, the receiver receives the channel output signal y and processes it to the received signal r via several transformations. Other extensions of the same AE-based concept were applied in multi-user communications over interference channels for orthogonal frequency division multiplexing (OFDM) systems with multipath channels in [124]. It was also used in the transceiver operation of a purely data-driven DL method without the use of accurate CSI in [125]. There is also another variation of the AE, namely the de-noising AE [126], which is a basic AE architecture that takes partially corrupted inputs in a random manner in order to address the identity-function risk, which the AE then has to recover or de-noise.

5) THE STACKED AUTO-ENCODER

The last of the DL applications uses a hierarchy of stacked AE (SAE), which consists of a stack of multiple layers of sparse AEs. A SAE employs greedy-wise training and has since been utilized with a Gibbs softmax activation layer to fine-tune a sub-band power allocation model in [127]. This hierarchical model of AEs is used to extract high-level features as well as correlations from input data through its multiple layers of non-linear processing units, which improves the accuracy of prediction. DL architectures that have been built in this way can be trained to reduce computational and time complexity, since once the model has been trained it can be able to achieve multiple objectives without the need of retraining and can also make inferences within the order of milliseconds. The computation of distinct features from the input data has since become easy with the advent of the SAE. For example, modulation identification and the classification of the transmitted signals in modern and intelligent systems such as CRs have remained a huge challenge using traditional methods, but they have since become easy through the use of the SAE.

A spectrum sensing algorithm for detecting OFDM signals using DL and covariance matrix graph by exploiting the proficiency of DL algorithms in image processing was proposed in [128]. The proposed spectrum sensing scheme was conducted by firstly approaching the detection of the OFDM signals through the analysis of the structural characteristics of the covariance matrix. Here, the covariance matrix was transformed into a gray-level representation, after which the gray-scale map of the covariance matrix was established. Then a CNN that was designed based on a LeNet-5 network was utilized to learn the training data so as to hierarchically obtain more abstract features. During testing, the test data was fed as input into the trained SAE model to complete the spectrum sensing process of OFDM signals. Another SAE application on spectrum sensing problems based on time domain signals was proposed in [129], as is shown in FIGURE 9 below.

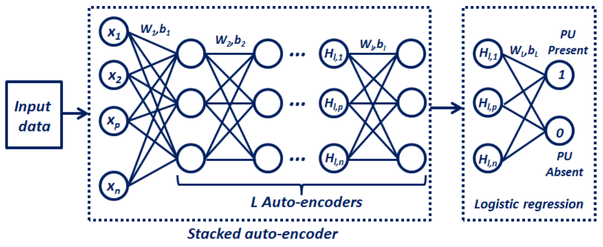

FIGURE 9. Implementation of time-domain signals in spectrum sensing using a SAE.

Here, the authors proposed a novel spectrum sensing framework for OFDM signals to address the issues of noise uncertainty, time delay and carrier frequency offsets suffered by conventional OFDM. The stacked AE was designed to extract the hidden features of OFDM signals and to use these features to classify the OFDM user's activities. In another implementation, the authors proposed to improve the spectrum accuracy under low SNR conditions using time-frequency domain signals as shown in FIGURE 10 below.

Here, despite the cost of higher computational complexity, higher spectrum sensing accuracy was achieved compared to the time domain implementation shown in FIGURE 9 above. In [130], a blind spectrum sensing method based on DL was studied in order to improve spectrum sensing in low SNR situations without any a priori information of the licensed users. In this contribution, three kinds of NNs were used together: a CNN, LSTM and fully connected NNs and resulted in improved performance compared to the traditional

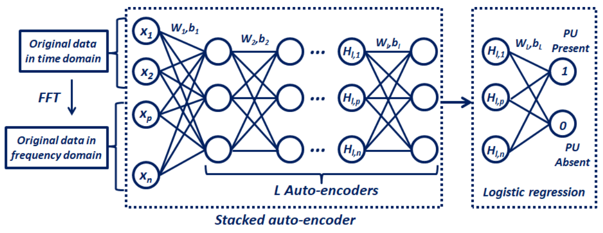

FIGURE 10. Implementation of time-frequency signals in spectrum sensing using a SAE.

energy detector, especially in low SNR regimes. The effect of different LSTM memory layers is also analyzed and explored the reason why the DL-based detector achieved better performance. The motivation for using several LSTM layers was to establish a more efficient model for the probability distribution of the observed sequence of HMMs, and to also extract the timing features of the signals as well as to distinguish the signal and noise from the timing regularity of the input data. The SAE model consisted of several LSTM layers and a regularization layer, whose role was to speed up the training of the model as well as improve on the regularization capability of the model. The fully connected NNs uses a stack of multilayer NNs to form a DNN model in order to refine the output features of the LSTM and also attenuate the influence of task-independent features on the decision results. The last layer of the model, which is the decision layer of the entire network, used a linear NN.

The authors of [131] proposed an energy saving and QoS provisioning scheme to address the issue of energy consumption in distributed CRNs. The authors focused on single BS management as the objective of their study, where physical resource blocks were considered as the spectrum units to be allocated to SUs. Here, the dynamic resource allocation problem was solved using a bipartite matching algorithm in order to obtain the resource consumption efficiency, then a predictive control scheme that uses a SAE method was employed and was utilized to predict the traffic load of the BS towards achieving better energy saving. Then, under quite general assumptions about the traffic arrival and service processes, the arrival of workload and their departure at the BS were defined using the Markovian arrival process and general service process, respectively. The possible impatience of packets waiting in the buffer was also taken into account in terms of the required transmission deadlines and two scheduling schemes were employed at service. As a result, the BS processor was treated as a hybrid switching system, toggling between the two packet scheduling schemes, first-come-first-served and processor sharing all treated with mean slowdown, depending on the forecast traffic load and required computational requirements. The simulation results that were obtained indicated that the proposed predictive control scheme achieved superior energy saving using the processor sharing even when the traffic load was increasing.

In an intelligent application of the DL technique to offer efficient load balancing in network slicing, the authors in [132] proposed a DeepSlice model. The implementation of the proposed model used a NN implementation for in-network DL to manage network load efficiency and availability through prediction. The available network key performance indicators (KPIs) were used to train the model to analyze incoming traffic and predict the network slice for unknown device types. The proposed model was able to make smart decisions in terms of selecting the most appropriate network slice, even in the case of network failure. However, fulfilling the flexibility, agility, and intelligence towards the provisioned services and infrastructure management tasks in increasingly heterogeneous, dynamic, and large-dimensioned networks is a challenging task that contradicts the existing network slicing solutions. To tackle this issue, the authors in [133] proposed a DL solution using a two-stage slicing optimization model with time-averaged metrics to safeguard the network slicing in the dynamic network. Here, it was assumed that prior knowledge about the environment was unknown, but could be partially observed at run-time. However, because of the unknown future system realizations before decision-making, obtaining an offline solution for this problem proved to be intractable, the DL technique was augmented with Lyapunov theories. This augmented approach enabled the system to learn safe slicing solutions using historical records and run-time observations. This proved to be always feasible and near optimal up to a constant additive factor as the results demonstrated a 2.6 times improvement over three other baseline algorithms. A comparison of the algorithms discussed is shown in TABLE **5** below.

D. ADVANTAGES OF DEEP LEARNING FOR SPECTRUM MANAGEMENT

Since it is evident from the discussion in the above section that DL strategies have not fully penetrated spectrum management, this section discusses the most pervasive advantages of DL that can be very instrumental in solving mobile and wireless networking problems, especially in spectrum management.

1) MULTIMODAL INFORMATION UNDERSTANDING

Multimodal learning refers to integrating a variety of modes or methods in executing certain processes. As opposed to its unimodal counterpart, which considers the understanding of a single mode of data, multimodal information understanding entails the harnessing of multiple modalities of information [134]. This approach to learning and understanding data combines extant theories of evidence accumulation in order to develop an integrated framework for modeling multiple processes manifesting in real-time. This approach to information learning and understanding will be very useful in solving a variety of problems in urban computing, where a number of contextual scenes need to be learned in real-time. The presentation of an urban computing scenario and information aggregation is elaborately discussed in future research directions in Section **VII-D**. DL, with its rich family of

TABLE 5. Application of deep learning techniques in mobile and wireless communications and spectrum management.

Application	Objective(s)	Deep learning method	Algorithm(s)	Refs
Speech & Image processing	Image reconstruction	Multilayer perceptron	Generalized delta rule	[101]
	Discriminative learning	Stacked denoising auto-encoder	Backward propagation	[126]
	Phonetic labeling of acoustic frames	Long short-term memory	Asynchronous stochastic gradient descent	[113]
Network security	Cryptography	Multilayer perceptron	Key generation & certification	[103]
	Network intrusion detection	Sparse auto-encoder, principal component analysis	Backward propagation	[120]
Radar & Wireless communications	Channel approximation	Complex multilayer perceptron	Differential evolution algorithm	[104]
	Mobile location estimation	Robust multilayer perceptron	Levenberg-Marquardt algorithm	[105]
	User mobility & time-series prediction	Random connectivity long short-term memory	Root mean squared error	[166]
	End-to-end signal reconstruction	Auto-encoder	Backward propagation	[123]
	Constellation design	Auto-encoder	Backward propagation	[137]
	Multicarrier systems	Auto-encoder	Stochastic gradient descent	[124]
	Channel agnostic end-to-end learning	Deep neural networks	Gradient descent	[125]
	Multi-cell resource allocation	Stacked auto-encoder	Stochastic gradient descent	[127]
Embedded systems & IoT	Compressive training energy harvesting	Convolutional neural networks	Stochastic gradient descent	[146]
	Proactive caching	Bidirectional deep recurrent neural networks	Adam optimization method	[116]
	Telemedicine QoE prediction	Multilayer perceptron	Stochastic gradient descent	[106]
	Chaotic time-series forecasting	Deep belief networks	Gradient descent	[109]
	Time-critical IoV information broadcasting	Deep belief networks	Firefly algorithm	[111]
Mobile cloud/edge computing	Resource management & energy harvesting	Long short-term memory	Energy-aware & adaptive algorithm	[119]
	Energy & performance efficient computation offloading	Deep neural networks	Tensorflow	[147]
Cognitive radio networks	Spectrum occupancy reconstruction	Restricted Boltzmann machines	Scaled stochastic gradient descent	[110]
	Spectral efficiency prediction	Long short-term memory	Mean squared error	[115]
	Resource management & predictive control	Model-based deep learning	Long short-term memory, stacked auto-encoder	[118], [131]
	Spectrum sensing	Convolutional neural networks	LeNet-5 network	[128]
	Spectrum sensing	Recurrent neural networks	Stacked auto-encoder	[129]
	Spectrum sensing	Convolutional NNs, Long short-term memory	Tensorflow backend	[130]
	Spectrum sensing	Ensemble classifier	AdaBoost algorithm	[143]
Network slicing	Efficient load balancing	General DL NN	Random forest algorithm	[132]
	Safeguard network slicing	Seq2Seq LSTM	Lyapunov algorithm	[133]

methods, which encompasses different NN models, hierarchy of probabilistic models, and a wide variety of supervised and unsupervised feature learning algorithms enables for the development of fundamentally new ways to think about communication systems. The area of mobile and wireless communications is a field that is rich with expert knowledge on how wireless channels of different types can be modeled in order to compensate for various hardware imperfections. Due to the capability of DL models to enable the implicit capturing of the intricate structures of large-scale data, they have helped in designing of optimal signaling and detection schemes that ensure reliable data transfer [168]. While it is an impossible task to write robust algorithms that can handle multiple tasks and handle large amounts of data of different modes using traditional optimization techniques, DL makes the implementation of algorithms that can learn to accomplish such tasks beyond the level of accuracy of traditional methods a possibility [136]. As a result, with DL, it is possible to design algorithms that enable straightforward analytic algorithms for symbol detection for a variety of channels and modes of information. Such models can be used to implicitly capture the intricate structures of large-scale data through detecting the different constellation symbols from wireless channels other than AWGN [137].

2) MULTIVARIATE REPRESENTATION OF DATA

Multivariate data representation entails representing data with different topology orders and different variables. A simple example of multivariate data is geometric data, which refers to multivariate data that is represented by coordinates, topology, metrics and order [139]. The DL technique is effective in handing geometric mobile data using a rebranded technique called geometric DL [138]. Geometric DL is the niche field under the umbrella of DL that aims to build NNs that can learn from non-Euclidean data. This niche field is a conundrum for the other ML techniques, since by far most DL is performed on Euclidean data, which is either 1D or 2D. Since all that can be observed in the real-world exists in three-dimensions (3D), multivariate representation of the data using geometric DL will enable the data to reflect that. Therefore, it is about time that DL approaches get to that level since the use of flying platforms such as unmanned aerial vehicles (UAVs) is rapidly growing in wireless networks [140]. Using this technique, mobile data related to mobile user locations as well as network connectivity can be represented using point clouds on graphs to reveal important geometric properties. Therefore, utilizing such an architecture has a great potential of revolutionizing the way mobile data is analyzed.

3) FEATURE EXTRACTION AND PATTERN RECOGNITION

Feature extraction, mainly associated with dimensionality reduction, entails the conversion of any given input data into a set of features [141]. Feature extraction usually begin with an initial set of consistent data, then develops some borrowed values, also known as features. The borrowed values are expected to be descriptive, and to simplify the consequent learning and observed steps [142]. DL methods, through the use of DNNs, are able to automatically extract high-level features through the layers of different depths from data that have complex structures and inner correlations. The importance of feature extraction and pattern recognition can be amplified in the context of spectrum sensing for PU signal classification, where the pattern of signals generated by heterogeneous sources can easily be recognized [143]. Since some signals are often noisy and usually exhibit some non-trivial spatial/temporal patterns, labeling them using feature engineering would require outstanding human effort and skills [144], DL would prove beneficial for this task. The role of DL in such problems would be to reduce the cost of the hand-crafted feature engineering in the processing of heterogeneous and noisy mobile data. The attractive advantage of DL in this task is its effectiveness in recognizing patterns in unlabeled data. Supervised learning also has exceptional capabilities in execution of pattern recognition tasks; however, it only thrives with labeled data. However, most of the current mobile systems generate either unlabeled or semi-labeled data, which presents a quantitatively different situations for supervised learning [145]. Supervised learning cannot utilize unlabeled data explicitly while DL provides a variety of methods for this task, such as the RBMs, which allow for the exploiting of unlabeled data in order to learn the useful patterns from the data in an unsupervised manner. Examples of applications include data clustering, data distributions approximation, unsupervised or semi-supervised learning, as well as one/zero-shot learning. This amazing capability of DL makes it attractive in solving mobile and wireless communication networking problems than the other ML methods. This is because it provides a means for achieving most of what is impossible to achieve using the other ML techniques.

4) COMPRESSED REPRESENTATION OF DATA

Compressed representation of data, or data compression, is an old signal processing technique of encoding information using fewer bits than in its original representation. While compressive representation is not easy to achieve using other ML paradigms such as linear regression and random forests, compressive representations learned by DNNs can be shared across different tasks. Because of the vast amounts of data at our disposal that require to be operated on continuously, regardless of the storage and access distribution and respond quickly to new information, DL techniques can be very beneficial in model compression and accelerated data retrieval using DNNs [146]. Thus, for purposes of reducing the huge amounts of overheads in data storage and processing in mobile networks, data compression can be instrumentally beneficial in compressing the size of data while maintaining their integrity and utility. In this case, it would be possible to train a single model to fulfill multiple objectives without requiring complete model retraining for the different objective tasks. Therefore, it can be argued that compressed representation of data is essential for mobile and wireless network engineering, as it reduces computational and memory requirements of mobile systems when performing multi-task learning applications [147]. Even though there is an explosive proliferation of DL applications in mobile and wireless communications, by contrast, the application of DL strategies in spectrum management is not straight-forward, except for spectrum sensing. Although DL strategies enable for the creation of machines that have high accuracy in specific tasks, they are still limited in making certain decisions. They are relatively weak in problems beyond classification and dimensionality reduction, and this has limited their applicability in the wireless network economics involved in spectrum access. Hence the need for DRL strategies.

VI. DEEP REINFORCEMENT LEARNING

Deep reinforcement learning (DRL) strategies are an extension of RL strategies that embrace the advantage of using DNNs as powerful function approximators. DRL is a reward (or throughput) maximization technique that uses either CNNs, ANNs or DNNs to represent Q-networks, and to train this Q-network to predict future reward. In a DRL scheme, the DNN is the agent that is used to improve the accuracy and speed of learning for RL algorithms in dealing with the high dimensional and continuous control problems through the continuous interaction with the environment. As a result, DRL strategies are able to provide autonomous decision-making mechanisms for the different network entities by learning and building from previous knowledge. In a DRL architecture, the DNN agents learn the mapping of the state-action pairs to rewards by using coefficients (i.e., weights) in order to approximate the function that relates the inputs onto the outputs [148]. This kind of a learning mechanism consists of finding the right weight coefficients by iteratively adjusting them along gradients that promise to reduce the error. The optimization is carried out using a DNN via logistic sigmoid activation functions and the error is propagated back to the first hidden layer to improve the optimization process. However, it should be noted that CNNs and DNNs perform differently in DRL strategies, so the choice of the type of NN that fits the application is very paramount.

A. DEEP REINFORCEMENT LEARNING ALGORITHMIC DESCRIPTION

In a DRL strategy, the initialization of the DNN coefficients is performed in a random/stochastic manner from the beginning. Performing an action depends on the future actions and states, thus the duty of the DNN agent is to rank the possible actions to execute and subsequently choose the best action using the policy, π. When a stochastic policy is applied, $\pi(s)$ maps the state to a distribution all possible action at the given state. Then, the action selected at time t is $a = \arg\max_{a\in A} \pi(s)$. For example, when using a distinct Q-learning method, the agent learns the Q-function, i.e., state-action, instead of policy. The reward is then defined by reward function, $r(s_t, a_t)$. Ignoring the effect of environmental noise disturbances, the system state can be defined using the packet arrival rate, s_{t_k} and the level of resource utilization at the BS, s_{t_u}, such that the state space is the union of the two, as follows:

$$s_t = s_{t_k} \cup s_{t_u}. \tag{22}$$

Then, assuming that there are M equally split physical resource blocks (PRBs) at the BS, the resource assignment can be defined as the action, such that the action space is defined as follows [149]:

$$a_t = \{1, 2, \cdots, |M|\}. \tag{23}$$

Then, considering the effects of power consumption, the available PRBs, and the reliability of the BS, the instantaneous reward function can be defined as follows:

$$R_t = \omega_1 P(t) - \omega_2 PRB(t) - \omega_3 R_e(t), \tag{24}$$

where the weighting factors, ω_1, ω_2, and ω_3, are negative indicating their influence on the overall reward. The term $P(t)$ is the total power consumption, denoting the energy cost component. The term $PRB(t)$ denotes the number of available resources, which depend on the number of packets still in the system, thus inversely correlated with the packet latencies - representing a negative component of the total reward. Finally, the term R_e represents the system reliability. Then, using the discrete-time based definition of $Q(s, a)$ given in (5), the DRL agent selects the action, a_t, using an ϵ-greedy policy. At the next, $t+1$, decision epoch, the Q-value updates using both the t-th and the $t+1$-th estimates as follows:

$$\begin{aligned} Q_{t+1}^{(k)}(s, a_t^{(k)}) = \bar{\alpha}_t Q_t^{(k)}(s, a_t^{(k)}) \\ + \alpha_t \left[R_t^{(k)}(s, a^{(k)} i_t) + \gamma^t Q_t^{(k*)}(s') \right], \end{aligned} \tag{25}$$

where the term $\bar{\alpha}_t \triangleq (1 - \alpha_t)$, $Q_t^{(k*)}(s')$ is the Q-value of the k^{th} SU that corresponds to the maximum $Q_t^{k*}(s') = \max_b Q_t^{(k)}(s', a')$ in the new state s' after taking and executing action $a_t^{(k)}$. In the case of (25), having assigned the values of the expected rewards, the Q-function has to select the state-action pairs that have the highest Q-value. Then, using feedback from the environment, the DNN agent utilizes the difference between the expected reward and the ground truth reward. This difference, known as the TD error, is used to adjust the DNN weights towards improving the interpretation of the state-action pairs. This operation results in an even more complete expression of a Q-function that takes into account not only the immediate rewards resulting from an executed action, but also the delayed ones that may be returned several time steps deeper into the sequence of the algorithm's trajectory. Thus, when the Q-function is called on any given state-action pair, the call of a nested Q-function is required in order to predict the value of the next state. However, this is dependent on the Q-function of the state after that one, and so on. This characteristic makes DRL strategies to be very useful in applications where agents need to act on time-series data in time-series modeling where RNNs are one of the state-of-the-art models present [112]. A typical implementation of an RA scheme in CRNs using the DRL strategy is shown in FIGURE 11 below.

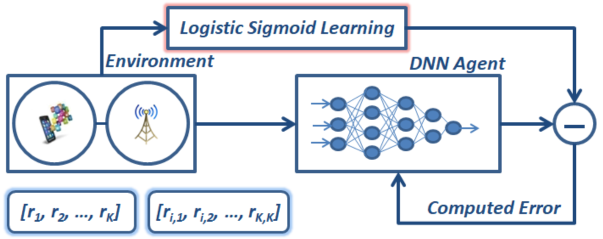
Logistic Sigmoid Learning
Environment
DNN Agent
$[r_1, r_2, \ldots, r_K]$
$[r_{i,1}, r_{i,2}, \ldots, r_{K,K}]$
Computed Error

FIGURE 11. A resource allocation procedure using the DRL algorithm. Using a DNN and a logistic function.

Dynamic spectrum management shown in FIGURE 11 above is a goal-directed phenomenon, which, due to the decision-making involved, DL algorithms cannot solve alone. This is an exemplary application of DRL in CRNs, extended from FIGURE 5, where the plurality of users (i.e., SUs) and number of BSs is indicated by the circles around them. Here, the terms $r_1, r_2, \cdots, r_K$ represent the transmission rates required by the K SUs, $r_{i1}, r_{i2}, \cdots, r_{KK}$ denotes the available data rates allocated by the BS to the SUs. In CRNs, each SU is an agent that acts by independently executing a DRL algorithm to select and opportunistically access the vacant licensed channel. The set of all actions depends on the number of channels that can be accessed, such that for N number of channels, the set of channels is denoted as $\mathcal{N} = \{1, 2, \cdots, N\}$. As a result, the action space is represented as $\mathcal{A} = \{a_1, a_2, \cdots, a_n\}$ that each SU can select from at time step $t \in T$, where n denotes the channel index. The set of all states, represented as $\mathcal{S} = \{s_1, s_2, \cdots, s_n\}$, denote the state of the channel selected by the SU. Therefore, the utility obtained after channel access is represented using the reward $\mathcal{R} = \{r_1, r_2, \cdots, r_{K,K}\}$, which represent the available rates offered by the BS. At this point, the problem becomes one of finding a policy that will maximize the received discounted reward V_i with a discount factor γ^t, using the value function as follows [94]:

$$V_k(s, \pi) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}(R_t^{(k)} | \pi, s_0 = s), \tag{26}$$

where the term $s_0 = s$ represents the start state. In spectrum management, spectrum access in particular, each SU seeks to find an optimal policy that will maximize its satisfaction (i.e., throughput), while adhering to the predefined SINR constraints. It does so by selecting an action from the set of all the available actions and follows a policy that will adapt its transmission power and other related transmission parameters such as the modulation scheme. It then observes the changes in the system in terms of the achieved reward as well as the new state. Assuming that each SU does not have any knowledge about the actions taken by other SUs or even the strategies that they follow, or even the effect of joint actions on states, it has to consider other SUs as part of the environment. The SU keeps improving its decision until it finally arrives at an optimal policy that maximizes the expected sum of discounted rewards using the appropriate Bellman equation. Thus, by using the Bellman optimality principle, equation (26) above can be solved by taking the optimal action if all the strategies thereafter are optimal, as follows:

$$V_k^*(s, \pi^*) = \max_a \left[R^{(k)}(s, a) + \gamma^t \sum_{s'} p(s'|s, a) V_k(s', \pi^*) \right], \tag{27}$$

where $p(.)$ represents the transition probability function that is determined by the power allocation of the SUs, and $V_k^*(s, \pi^*)$ can be approached by the Q-function and updated as in (25). A DRL application was investigated in [151] for solving resource management problems in a network slicing scenario, which included a radio resource slicing as well as a priority-based core network slicing. Also, a DRL-based approach for circumventing the missing gradient problem when training transmitters was proposed in [83]. Here, the transmitter was regarded as the agent to be trained to convert source data into transmit symbols, while the wireless channel and the receiver were treated as the environment. At each instant of time, the transmit data are treated as the system state that the transmitter has to observe, while the transmit signals are taken as the actions executed by the transmitter. Then, the end-to-end loss on each sample was computed at the receiver and fed back to the transmitter agent as the reward from the environment. At the end, this approach helped the transmitter to learn the optimization of the end-to-end loss without the use of gradients from the wireless channel, but through the use of the policy gradient algorithm. Another example was the use of the DRL strategy in deriving an optimal time scheduling policy for the gateway by back-scattering in order to deal with large state and action spaces, which was proposed in [152]. This shows the capability of DRL algorithms, which makes it versatile in being modeled under different strategies. The formulation in FIGURE 11 uses logistic sigmoid techniques as the algorithm activation functions. When the Q-learning algorithm is used, the DRL strategy becomes a DQN [153], which is discussed in the following subsection.

B. DEEP Q-LEARNING NETWORKS WITH EXPERIENCE REPLAY

In the previous section, it has been elaborated how DRL strategies are the most conservative in most of its applications. Now that the application of DNNs in RL has been discussed, another important concept is the extension of the DRL strategies called DQNs. These are powerful algorithms with an exceptional ability of creating cheat sheets for DRL strategies. However, care must be taken in creating the cheat sheet - it must not be too long. With a very long cheat sheet, things might quickly spiral out of control. Using a DQN is like taking some random actions and learning from them through the Q-value function and it is a regression problem (i.e., ℓ_2-loss is used) where two networks are used for training. However, when the learning objective is to maximize the expected cumulative reward, either policy gradients or the Monte-Carlo methods are used. With this feature, the DQN technique is a very important advancement of the DRL strategy, and the flowchart that details this procedure is illustrated in FIGURE 12 below [155].

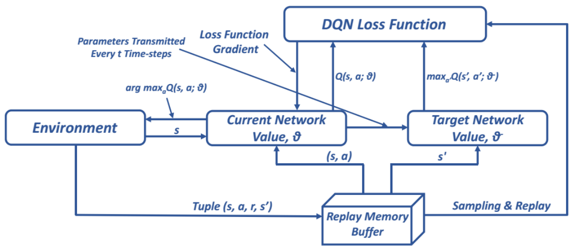

FIGURE 12. Flowchart of a DQN algorithm with experience replay.

In the DQN strategy, when the optimization focus is on a class of RL algorithms that learn by performing gradient descent on policy parameters, policy gradients are used in the place of the cost criteria. The flowchart of a DQN shown in FIGURE 12 above is supposed to learn a strategy that will lead to the best possible reward using experience replay. The DQN uses two separate MLP networks to act as Q-network approximators, denoted by θ and θ^{-}, respectively. With action selection performed using the action-value function, the experience replay will assist the DRL agent to figure out which actions to execute.

1) THE OPTIMIZATION OBJECTIVE OF DQNs

The objective of the DQN is to obtain solutions that can effectively address high-dimensional and continuous control problems through the use of DNNs acting as powerful function approximators. The difference between the ordinary DRL strategy and a DQN algorithm is it combines Q-learning with two DNNs, which it uses as separate Q-value networks, as is seen in FIGURE 12 above. DRL with experience replay, also known as DQNs with policy gradients, is an extension of the traditional RL method, which is aimed at reducing training time by storing a number of transitions to be sampled from later for the agent to learn from. Then, the objective of DQNs with experience replay is to address the problems that lead to a DRL agent misunderstanding the environment. The only thing that can result in a DRL agent's misunderstanding of the environment are consecutive interdependent states that also look very similar [155]. Since the objective is that of maximizing the expected cumulative discounted reward, the gradient of the objective function is given as follows [154]:

$$\nabla_{\theta}\mathbb{E}_{\pi_{\theta}}\left[\sum_{t=0}^{\infty}\gamma^{t}r_{t}\right] = \mathbb{E}_{\pi_{\theta}}\left[\nabla_{\theta}\log\pi_{\theta}(s,a)Q^{\pi_{\theta}}(s,a)\right], \quad (28)$$

where $Q^{\pi_{\theta}}(s, a)$ is the expected cumulative discounted reward obtained from deterministically selecting an action a under the guidance of the policy π_{θ}. However, in the case of the Monte-Carlo technique, the agent must sample multiple trajectories and then uses an empirically computed cumulative discounted reward, v_t, as an unbiased estimate of $Q^{\pi_{\theta}}(s_t, a_t)$. The result of this is the well-known REINFORCE algorithm reported in [60], which is used to update the policy parameters using gradient descent as follows:

$$\theta \leftarrow \theta + \alpha\sum_{t}\nabla_{\theta}\log\pi_{\theta}(s_t, a_t)v_t. \quad (29)$$

Here, the input state s_t, which might be a clip of a time-series, is mean-value-normalized and the value of Q for a given state-action pair. This is the estimated expectation of total future rewards, i.e., reward or cost for the present task, discounted at the current step. The first step initializes the network $\mathbf{P}$ with random parameters θ, and the procedure predicts the action probabilities $P(A|A;\theta)$ and samples an action following policy π. This leads to the maximization of $\sum_t \log P(y_t|x_t;\theta)$ defined as follows:

$$J(\theta) = \sum_{t}\log P(y_t|x_t;\theta)\cdot\hat{A}_t, \quad (30)$$

which is performed at every time step until the last episode using x_t and y_t as training examples. Then, at the last hidden layer, the error $\frac{\partial}{\partial\theta}J(\theta)$ is propagated back to the first hidden layer for an update of the cost function

$$\nabla J(\theta) = \sum_{t}\underbrace{\nabla\log P(y_t|x_t;\theta)}_{\text{Actual - Predicted}}\cdot\hat{A}. \quad (31)$$

In the Monte-Carlo method, the term $\hat{A}_t$ is defined as the advantage which, when it is positive, pushes up the probabilities for all actions, otherwise it pushes them down. Using policy gradients is like learning the optimal behavior directly from the experiences, i.e., using value function; and it is a classification problem, i.e., the maximum log likelihood is used with some minor changes. DQNs with policy gradients are also referred to as DRL strategies with experience replay.

2) TRAINING A DQN WITH EXPERIENCE REPLAY

Except for the experience replay, the procedure is the same as the DRL - a DNN is used in approximating the Q-value function where the state is given as the input to generate an output consisting of a Q-value of all possible actions. Here, the computation of the loss function is actually a regression problem, where an MSE is regressed from the predicted Q-value and the target Q-value, $\hat{Q}$, which is calculated as follows:

$$\mathcal{L}(\theta_i) = \mathbb{E}[(\hat{Q}_i(s,a;\theta_i) - Q_i(s,a;\theta_i^{-}))^2], \quad (32)$$

where $\hat{Q}_i(s,a;\theta_i)$ is the action-value function approximator, and $Q_i(s,a;\theta_i^{-})$ represents the target action-value function approximator. The parameters θ_i represents the current network parameter, such that the target Q-value $\hat{Q}$ can be utilized to determine the gradient of the loss function $\mathcal{L}(\theta_i)$ with respect to θ. The same narrative applies to the previous network parameter θ_i^{-}. However, since the knowledge of the target or actual value is not present, it becomes an ordinary RL problem. Then, it can be argued that the system is predicting its own value, but since the reward $r_t \in \mathcal{R}$ is an unbiased and true reward, the network will update its gradient using backward propagation until it finally converges. In every iteration, the previous network parameter, θ_i^{-}, is replaced by the current and updated parameter θ_i. The network parameters are updated using the SGD algorithm, and the DQN subsequently completes the learning mechanism in (32) by computing the cost function using an SGD algorithm as follows:

$$\mathcal{L}(\theta_i) = \mathbb{E}\left[(r_i(s,a) + \gamma^t\max_{a'\in\mathcal{A}}(\hat{Q}_i(\hat{s},\hat{a};\theta_i^{-})) - Q_i(s,a;\theta_i))^2\right]. \quad (33)$$

Similar to the traditional RL strategy, the action selection procedure of a DQN follows the usual ϵ-greedy approach. But in order to ensure a sufficient exploration of all actions, the Gibbs softmax action selection strategy is applied, which is given as follows: $\pi(s,a) = \frac{e^{Q(s,a)/\tau}}{\sum_{a\in A}e^{Q(s,a)/\tau}}$. This is an off-policy technique borrowed from the Boltzmann distribution, where the parameter τ represents the temperature parameter controlling the expected reward for the probability of a given action executed. The reward $r_t(s,a) \in \mathcal{R}$ that is obtained by following the prescribed policy either results in the satisfaction (i.e., maximized reward) of the agent or penalties if certain constraints were not met.

Thus, in order to solve (25) using this technique, assuming that a DNN is employed as an efficient nonlinear approximator that will be used to estimate the action-value function $Q_i(s, a; \theta_i) \approx Q_i^*(s, a)$, using the advantage of experience replay, the sampling procedure is illustrated in FIGURE 13 below [94]:

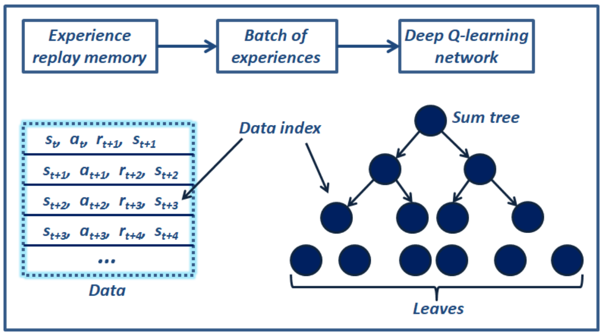
Experience replay memory
Batch of experiences
Deep Q-learning network
Sum tree
Data index
$s_t, a_t, r_{t+1}, s_{t+1}$
$s_{t+1}, a_{t+1}, r_{t+2}, s_{t+2}$
$s_{t+2}, a_{t+2}, r_{t+3}, s_{t+3}$
$s_{t+3}, a_{t+3}, r_{t+4}, s_{t+4}$
...
Data
Leaves

FIGURE 13. An illustration of a memory object using a sumtree to represent the sampled data.

In FIGURE 13, at each sampling instant, the experience, $e_i^t = (s_i^t, a_i^t, r_i^{t+1}, s_i^{t+1})$ of the agent is stored in an experience replay buffer memory $D : D_i^t = e_i^1, \cdots, e_i^t$. Thus, at each time-step, the network parameters θ_i are updated using a mini-batch of random samples of transitions $p_i = (s_i, a_i, r_i, \hat{s}_i)$ from the replay memory D_i.

To explain this, take the example of an autonomous vehicle that is being trained on how to drive itself on a straight road. If the first section of the toad is straight, the DRL will learn that and not how to deal with curved sections that may appear later. It will, however, save that initial experience as a straight road experience in its memory. Then, if it reaches a curved section along the road, it also learns driving around a curved road and save that experience in memory. The good thing about experience replay is that the knowledge experiences are not put through the DRL agent immediately, but are saved in memory buffer first. These experiences are characterized by the state that the vehicle was in, the action that it executed, the state that it arrived at, and the reward that it received. Then, once the vehicle reaches a certain threshold, this is where the concept of experience replay becomes very important. The vehicle will be instructed by the agent to learn from it by sampling experience from the batch of saved experiences. The DRL agent can randomly select uniformly distributed samples from the batch of experiences to learn from. The reason of random sampling the experiences is to break the bias that comes with the sequential nature of the environment.

C. APPLICATION OF DEEP Q-LEARNING NETWORKS IN SPECTRUM MANAGEMENT

When executing the procedure illustrated in FIGURE 11 above using a DQN, the agent learns to perform resource allocation tasks and obtains high rewards by choosing better actions. At each time-step $t \in T$, the DNN agent is given observations in previous time-steps and then it chooses an action, which results in a reward. This procedure can be performed based on either univariate or bivariate methods. In the univariate case, the model is tested on whether it can capture the underlying dynamics, while in the bivariate case the model is tested on whether it can utilize the hidden relationship among inputs. For both cases, the inputs are positive values and a typical application of DQN and experience replay in a CRN environment can be done as illustrated in FIGURE 13 above.

An evolutionary game theoretic framework was combined with DRL in distributed DSA to improve spectrum management through effectively utilization of spectrum resources [156]. The main framework utilized in this method is a DQN where individual users independently used the DQN algorithm for channel selection. Also, a replicator dynamic that uses an evolutionary game theory strategy was added to setup the reward function so that the DRL can effectively balance the collaborations among SUs. This improved the spectrum utilization and reduced the collision rate among SUs. A joint routing and resource management scheme for addressing energy consumption and transmission delays was proposed in [157]. Here, an energy-efficient cross-layer design was done using an apprenticeship DRL scheme, and in order to guarantee energy-efficient operations and also compress huge action space, a dynamic adjustment rating concept was introduced in order to efficiently regulate transmission power using a multi-level transition mechanism. Then, a technique called prioritized memories DQN from demonstrations (PM-DQfD) was utilized to speed up the convergence and reduce memory occupation. In the same contribution, the PM-DQfD was applied to the cross-layer routing design in order to improve power efficiency subject to routing latency reduction. In another contribution in [158], a sensing-throughput trade-off scheme that allows SUs to detect the presence of PUs and transmission data simultaneously was proposed. Here, a DRL-based joint spectrum sensing and power control algorithm for downlink communications in a cognitive small cell was utilized to adapt in unknown environments and achieved performance closer to a genie-aided method with the optimal spectrum utilization, especially in high-SNR regimes.

The authors in [159] proposed a DRL framework for solving a spectrum allocation problem in a large-scale IAB network. The objective here was to maximize the sum log-rate of all UE groups. Thus, the available spectrum resources were divided into several orthogonal sub-channels, whereby both the donor base station (DBS) and all the IAB nodes had to share the same spectrum resource for allocation. The DBS utilized those sub-channels for access links of associated user equipment (UE) as well as for backhaul links of associated IAB nodes, while IAB nodes could utilize all the sub-channels for its associated UEs. The spectrum allocation problem was formulated using a mix-integer and non-linear programming. To tackle the time-varying nature of the IAB network, a DRL method was proposed by integrating an actor-critic spectrum allocation scheme and DNN. This was to achieve real-time spectrum allocation for different IAB deployment scenarios.

The authors in [160] proposed a DRL-based technique for CR DSA that performs distributed joint multi-resource allocation to satisfy the primary link interference constraint while simultaneously maximizing the performance of the secondary network. The satisfaction of SUs was measured using the mean opinion score metric to enable seamless integrated RA for dissimilar traffic. The RA problem was solved using a DQN algorithm. Here, the learning process was improved by incorporating transfer learning to the learning procedure and simulation results showed a substantial reduction of the iterations towards convergence compared to a DQN without transfer learning. A DQN framework that used a QoE-driven CR scheme where SUs offered heterogeneous traffic to the network was proposed in [94]. Here, docitive learning schemes were used to improve SU perception of the QoE through transfer learning. SU experience evaluation of multimedia services on-the-fly was done using the mean opinion score for heterogeneous traffic. The DQN framework demonstrated that it can combine deep exploration with DNNs for exponentially faster learning and convergence. The work in [94] was improved using an SU-experience-oriented resource allocation scheme for efficient real-time CR processes in [107]. The objective of the authors was to improve CR adaptation and intelligent RA using deep neuroevolution (DNE). The DNE technique combined DRL with an evolutionary algorithm in order to improve RA stability when the number of SUs increased. A stable learning framework was achieved by introducing the phenotypic plasticity of transmission rates and delay constraints inside an MLP.

The 5G and MEC have recently been envisaged to serve various emerging use cases with diverse multiple requirements such as radio, transport, and computation. In this way, the provisioning of network slices within the two paradigms requires end-to-end resource orchestration, which is very challenging. Since AI at the edge enable real-time decision-making, it is central to achieving objective of network slicing, as such it is paramount for promoting the long-term sustainability of mobile and wireless networks. To this effect, the authors in [161] proposed a decentralized resource orchestration system for dynamic end-to-end network slicing using the DRL strategy called EdgeSlice. The proposed scheme consisted of a performance coordinator and multiple orchestration agents for efficiently orchestrating end-to-end resources. The role of the performance coordinator was to manage the resource orchestration agents in order to avoid SLA violations in the network slices. It does so by learning the resource demands of network slices and then orchestrates the RA accordingly to optimize the performance of the slices under constrained networking and computing resources. Then, a radio, transport, and computing manager were designed to dynamically configure the end-to-end resources at runtime. The performance evaluation of the EdgeSlice was evaluated through both experiments and trace-driven simulations, and the results indicated that the EdgeSlice outperforms baseline algorithms in terms of improved performance, scalability, as well as compatibility.

A joint allocation of spectrum, computing, and resource storage in an MEC for supporting different vehicular applications was proposed in [162]. Here, the authors considered two MEC architectures and multi-dimensional resource optimization problems. Since these architectures have high computational complexity and very long problem-solving time, the problem was divided into two sub-problems. Since the two formulated problems were computationally intractable in real-time for the RL strategy, a DRL was utilized to transform and solve them using deep deterministic policy gradient (DDPG) and hierarchical DDPG (HDDPG)-based algorithms. Using these two hierarchical architectures, the network dynamics could be learned automatically via offline training, and appropriate RA decisions could be obtained to satisfy the QoS requirements of the different vehicular applications. The obtained simulation results indicate that the proposed DDPG and HDDPD-based resource management schemes could converge within acceptable training episodes and also outperformed the DPG-based schemes in terms of both QoS satisfaction and latency/QoS satisfaction ratio.

In another multi-use case scenario, a double use case network slicing method for enabling adaptive and automated slicing was studied in [163] for efficient RA in dynamic vehicular and smart city environments. The primary objective was to dynamically allocate limited resources at the network edge to vehicular and smart city users with heterogeneous latency and computing demands. To efficiently utilize the limited resources, the authors developed a model based on a cluster of nodes coordinated by an edge controller. For individual service requests within a cluster, the edge controller decides on whether the request has to be served locally or referred to the cloud. In order to adaptively learn the optimal slicing policy, the problem was formulated as an infinite-horizon MDP and solved using a DRL strategy. The results showed that the proposed scheme could quickly learn the optimal policy in different scenarios and different design objectives. In order to address the dynamic coupled RA problem faced when using model-based optimization methods, such as queuing-theoretic techniques, the authors in [164] proposed a novel DRL strategy for bandwidth and virtual machine (VM) allocation. The optimization problem was formulated by setting two service types: (i) service upon arrival, and (ii) batch service. Buffers were introduced to address the problem of allocating multiple resources simultaneously. The overall problem was solved as a constrained problem and the performance evaluation step; two workload traces were used, one for CPU and the other one for bandwidth requests. The proposed scheme was then evaluated with four scenarios of resource budget, and the results indicated an improved overall resource utilization.

An end-to-end architecture network slicing RA algorithm suitable for multi-slice and multi-service scenarios using a DQN strategy was proposed in [165]. In order to achieve a dynamic allocation of resources that maximizes the number of access users, the proposed algorithm considers the RAN and core network slices. A mixed integer programming problem that needs to be adjusted according to a non-static environment was used to build the model. Then, due to the capability of a DQN to perceive environmental changes and make dynamic decisions, it was used to solve the mixed integer programming problem. Since the reward value of the DQN had to be computed under each decision, the problem was divided into the core and access sides, and dynamic knapsack and link mapping algorithms were used to obtain the overall reward. The simulation results obtained indicated that the DQN scheme provides an average access rate higher than 97%. When compared with optimal allocation schemes of the access side, a 9% increase in average access rate was obtained for delay-constrained slices, while a 5% increase was observed for rate-constrained slices in a dynamic environment.

In [166], a network slicing scenario consisting of several slices in a RAN where BSs share the same physical resources was considered. Here, the variation of service demands was considered as the environmental state, while the allocated resources were considered as the actions. A generative adversarial network-powered deep distributed Q network (GAN-DDQN) was proposed to reduce the effects of the annoying randomness and noise embedded in the received service level agreement satisfaction ratio and spectrum efficiency. The role of the GAN-DDQN was to learn the action-value distribution by minimizing the discrepancy between the estimated action-value and the target action-value distribution. Then, a reward-clipping mechanism was used to stabilize the GAN-DDQN against the effects of widely-spanning utility values. A dueling GAN-DDQN that uses a specially designed dueling generator was then developed in order to learn action-value distributions by estimating the state-value distributions as well as the action advantage function. Finally, we verify the performance of the proposed GAN-DDQN and Dueling GAN-DDQN algorithms through extensive simulations.

The authors in [167] considered the network slicing problem as a game-theoretic radio access network (RAN)-only slicing model. Here, the physical infrastructure was split into slices that provide computation and communication functionalities. Using the game-theoretic framework, a limited number of channels were auctioned across scheduling slots to mobile users of multiple tenants, i.e., service providers. Each tenant behaves selfishly in order to maximize its expected long-term reward from competing with the other tenants for channel orchestration. This provides its mobile users with the opportunities to access the computation and communication slices. The problem was the formulated using stochastic games where the decision-making of each tenant was based on the global network dynamics as well as the joint control policy of the other tenants. An abstract stochastic game model among the tenants was constructed with local conjectures of channel auctions in order to approximate the Nash equilibrium solutions. A per-tenant MDP was formulated and linearly decomposed in order to simplify the decision making for each tenant. Then a DRL strategy was used to derive an online scheme to approach the abstract control policies. In the performance evaluations, two observations were made: (i) each tenant was able to behave independently with the conjectures of other tenants' behaviors; (ii) the decisions on channel auction and computation offloading as well as packet scheduling decisions could be made sequentially. These two observations indicated that the proposed framework could find optimal abstract control policies and achieved significant performance gains compared to other baselines.

Leveraging the knowledge and insights retrieved from the data, the authors in [168] developed a novel ML-based scheme for dynamic resource scheduling for network slicing. The objective of the developed scheme was to achieve automatic and efficient resource optimization and end-to-end service reliability. However, obtaining user-related data that is crucial for understanding user behavior and requests is difficult due to privacy issues. In addressing this issue, a DRL strategy was leveraged to extract knowledge from experience by interacting with the network and enabling dynamic adjustment of the resources allocated to various slices. This was done to maximize the resource utilization while guaranteeing the QoS. The experimental results showed that the proposed resource scheduling scheme allocates resources dynamically to multiple slices and the corresponding QoS requirements were met.

With the concept of performance isolation between network slices requiring fine-grained resource reconfiguration, which leads to extremely high computational complexity. which affects the operations of all the different layers of the protocol stack, a problem which was revisited in [169]. Here, the authors investigated the fine-grained reconfiguration within the core network slice with the objective of minimizing long-term resource consumption. However, due to the curse of dimensionality of the DRL strategy, the problem seemed intractable even with the conventional DQN architecture as the multi-dimensional discrete action space is difficult to explore efficiently. In order to address the curse of dimensionality, a discrete branching dueling Q-learning network (BDQN), which incorporate a branching architecture into the conventional DQN in order to decrease the number of estimated actions was proposed. Then, an intelligent network slice reconfiguration algorithm (INSRA) was developed based on the discrete BDQN, and was evaluated through extensive simulation experiments. The numerical results indicated that the INSRA can minimize the long-term resource consumption and also achieve high resource efficiency compared with several baseline algorithms.

In a multi-use case scenario, the network slicing problem needs to accommodate the diverse resource demands and various performance metrics. A multi-use case network

TABLE 6. Application of deep reinforcement learning techniques in spectrum management.

Application	Objective(s)	State(s)	Action(s)	Algorithm(s)	Refs
Cognitive radio	Improving QoE using transfer learning	Number of SUs, number of channels	Number of allocated resources	Deep Q-learning networks	[94], [160]
	Improving resource allocation stability	Number of SUs, Size of DNN	Dynamically allocate resources	Deep neuroevolution	[107]
	Balancing SU collaboration	Competition among secondary users	Channel auction	Evolutionary game theory, Deep Q-learning networks	[156]
	Reducing energy consumption and transmission latency	Resource demands, power management state	Dynamic resource adjustment, regulate transmission power	Apprenticeship DRL, prioritized memory DQN from demonstrations	[157]
	Improving spectrum sensing throughput	Number of primary channels	Dynamic power adjustment	Deep Q-learning networks	[158]
Network slicing	Avoiding SLA violations	Resource demands, computing resources	Dynamically configure E2E resources	Decentralized Deep reinforcement learning	[161]
	Improve latency/QoS satisfaction ratio	Number of vehicles, QoS requirements	Automatically learn network dynamics	DDPG and HDDPG	[162]
	Enabling automated network slicing	Heterogeneous latency, computing demands	Offload traffic to cloud or edge	Deep reinforcement learning	[163]
	Simultaneous allocation of multiple resources	Service types, number of VMs	Dynamically allocate virtual machines	Deep reinforcement learning	[164]
	Maximize number of access users	Number of slices, Number of services	Map slices to services	Deep Q-learning networks	[165]
	Improving SLA satisfaction	Variation of service demands	Number of allocated resources	GAN-DDQN, and dueling GAN-DDQN	[166]
	Provision of computation & communication	Number of channels, number of tenants	Computation offloading packet scheduling	Deep reinforcement learning	[167]
	End-to-end service reliability	User behavior, dynamic user requests	Dynamically adjust resources	Deep reinforcement learning	[168]
	Minimizing long-term resource consumption	Number of slices, number of services	Fine-grained resource reconfiguration	Discrete-branching DQN	[169]
	Effective allocation of resources	Resource demands, performance metrics	Learn optimal policy, shape reward	Deep deterministic policy gradient	[170]
	Guarantee SLA on different services	Varying user demands	Number of allocated resources	Deep Q-learning networks	[171]
	Improve mean slice satisfaction requirement	User mobility, slice fluctuation	Service switching	Deep reinforcement learning - Apex-X	[172]

slicing problem was studied in [170] in order to effectively allocate network slices to the different use cases. The authors proposed a DeepSlicing technique that integrates alternating direction method of multipliers and the DRL strategy. The DeepSlicing technique entails that the problem is decomposed into several sub-problems, and the authors decomposed the network slicing problem into a master problem and several slave problems. A deep deterministic policy gradient (DDPG) algorithm with an augmented state space and reward shaping to enable for the coordination of the DDPG agents in solving the constrained RA problem was used. The master problem was solved based on convex optimization, while the slave problems were solved using a DRL strategy which learns the optimal RA policy. The performance of the proposed algorithm was extensively evaluated through network simulations and the results obtained significantly outperformed the baseline method and closely approaches the optimal solution.

In another contribution, the authors in [171] considered a scenario with several slices in a radio access network (RAN) where the BSs share the same physical resources. In order to solve the problem, they leveraged a DQN algorithm, where the advantages of the actor-critic (A2C) functionality were used to handle the varying demands. In their formulation, the varying demands were treated as the environmental state, while the allocated resources as the environmental action. In order to perceive the environmental state in the midst of user mobility, a LSTM was incorporated into the A2C functionality. Then, the LSTM-A2C algorithm was used to track user mobility and improve system utility. The results obtained show that the proposed LSTM-A2C improves spectrum efficiency and also guarantees the SLA on the different services in cases with large fluctuations in user requests.

The provision of resources in network slicing depends on the number of slices, but the allocated resources cannot be accurate if the number of slices keep on changing. In the radio access network, the number of slices controlled by the BS fluctuates in terms of user ingress and egress from the BS coverage area and service switching on the respective sets of user requirements. In order to address this problem, the authors in [172] proposed a radio access network slicing method that flexibly allocates access resources using a DRL strategy called Ape-X. The proposed method makes optimal RA independent of the number of slices using a DRL strategy that learns the best action for each state through trial and error. In achieving the independence from the number of network slices, Ape-X manages resources on a one-slice-by-one-agent basis. The advantage of the Ape-X is that the agents are employed in parallel and the model that learns various environments can be generated through trial and error of multiple environments. In addition to the Ape-X method, they designed a model that satisfies the slicing requirements without over-provisioning resources. This additional model made it possible to optimally allocate resources independently of the number of slices by changing the number of agents. The proposed method was evaluated in multiple test scenarios, and the evaluation results indicated a mean satisfaction of slice requirements of approximately 97%. A comparison of DRL algorithms used in dynamic spectrum management is shown in TABLE 6 below.

VII. CONCLUSION AND RECOMMENDATIONS FOR FUTURE RESEARCH

In this article, a prospective review of the application of deep architectures in CRNs were performed. Based on the recent advancements in the application of deep architectures in spectrum management, it has been found that DRL strategies are indispensable tools in enabling intelligent spectrum management. It was also realized that the recently adopted variant of DRL - the DQN has so far seen limited application in spectrum management. DQNs have a great potential of revolutionizing future spectrum management by empowering DL techniques by exploiting and extracting the rich features of DRL strategies. In order to realize the level of intelligence required in spectrum management, the objective should be to create efficient algorithms that unite function approximation and target optimization by mapping state-action pairs to expected rewards [56], [150].

A. IMPROVING SERVICE PROVISIONING THROUGH PREDICTIVE ANALYTICS

In order to meet the greater demand through willingness to understand subscriber requirements and ensure targeted customer offerings, while maintaining best quality networks despite increasing costs, it is imperative to look at predictive analytics. Future generations of mobile and wireless networks will be more sporadic and manifest themselves in several forms such as time, subscriber, location and application [173]. Through the use of thousands of mobile handsets streaming content from the internet, telecommunication operators currently have a mound of data containing continuous information. From the thousands of infrastructure elements and millions of handheld devices, the generated data are huge and makes for an exciting exercise to understand what the data are all about, as well as how it can be used for greater value creation. Usage variables such as access, mobile device, personal computer, length of connection, login frequency, can be presented as percentages in order to obtain better data visibility. However, due to the lack of proper data analysis equipment, this information is far beyond what they can be able to grasp and make sense of. With the IoT forming the key component of next generation mobile and wireless network deployment strategy, there is an escalating requirement for network service providers to understand the needs of their subscribers. If one can add some exploration of this huge amount of data to exploit the paradigm shift brought about by the IoT, and then exponentially multiply the number of data points and devices, this can result in gigantic volumes of data.

Big Data and predictive analytics provide new light to companies who seek to develop greater understanding of its benefits. Thus, looking into Big Data and predictive analytics will enable enterprises to direct their efforts towards efficiently meeting customer needs. For example, in examining the subscriber in terms of their location, one might realize the percentage of users that consume most of the data and their location in the network. In order to achieve this kind of insight into the usage behavior of subscribers, network operators can adopt data analytics techniques and put in place relevant predictive approaches. Using these approaches, analytics can reveal exactly which users are consuming how much of bandwidth and where they are located within the network [174]. In this way, communication and computation resources can be appropriately provisioned for that location, hence improve the QoS and QoE. This level of foresight is the key to not only unlocking the full potential of network slicing and self-organizing networks in the radio access network, but also to maximizing return on investment for software-defined networking (SDN) and network function virtualization (NFV) in the core. Therefore, integrating Big Data analytics into the operation of mobile and wireless networking can be explored through network optimization to reach the objective of improving user QoE. Here, the first objective can be the Big Data-driven framework for mobile network optimization; the Big Data characteristics collected from both the user side and the network side and presented together.

B. IMPROVING DATA PRIVACY THROUGH FEDERATED INTELLIGENCE

With the proliferation of AI-based applications and services, data privacy and security have become the most topical and critical challenges in Big Data. Using traditional methods that employ the centralized approach, data is collected and aggregated in a data center (DC) in order to train ML-based learning models. However, the IoT and Big Data era, where systems are predominantly generating distributed and essential data, transferring such vast amounts of data to a DC seems cumbersome [175]. As such, federated intelligence has emerged as a prospective solution to address this problem by facilitating distributed collaborative learning. Federated intelligence involves the application of federated machine learning (FML) techniques in collaborative training without disclosing original training data. In FML, the training data is distributed unevenly across all the learning agents, instead of being centralized [176]. Then, instead of the learning agents sharing their training data, they use their local data to train ML models using SGD algorithms [177]. After training, only a summary of their data, i.e., gradients, are shared with the central entity such as the MEC server that is co-located with the BS. In return, the MEC server then computes a global model by averaging the individual models into an aggregated model, which is then shared back to the individual learning agents via the BS. This application is primarily inspired by the expected applications of 5G networks where information privacy is paramount and communication costs are the principal constraint. A graphical representation of the processing steps of the SGD algorithm applied in secure spectrum sensing is shown in FIGURE 14 below [110].

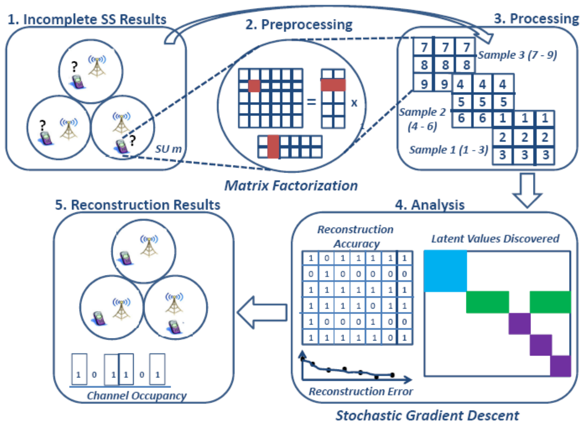

FIGURE 14. Spectrum occupancy data processing using matrix factorization and stochastic gradient descent.

The training technique shown in FIGURE 14 above was applied in spectrum sensing for distributed cooperative CRN environments. This FML technique can serve as a security defense mechanism against Byzantine attacks in CRNs. This is because each learning agent, i.e., user devices, independently computes an update of its current local model using the data set stored in its memory and then communicates only the gradient to the MEC server. At the MEC server, the local updates from each user device are aggregated into a single global model that is broadcast back to the learning agents.

Federated intelligence typically reduces computation costs as well as the two-way communication costs where communication efficiency was of utmost importance. The SGD algorithm the perfect choice for this application for the following reasons: (i) structured updates, where updates are learned directly from a restricted space that is parameterized using a smaller number of variables that are either of low-rank or random masked; and (ii) sketched updates, where full models are learned and then compressed using a combination of quantization, random rotations, and sub-sampling before sending them to the MEC server. This can reduce the communication cost by several orders of magnitude since it enables the collaborative training of ML models and also enables DL for mobile edge network optimization.

C. INTELLIGENTIZATION OF CORE AND EDGE NETWORK FUNCTIONS

With the era of network softwarization (i.e., 5G) already at its height, the era of intelligentization (i.e., 6G) is quickly making its way into the wireless communications space, BS infrastructure and network configurations have to be virtualized. In both generations, i.e., 5G and 6G, network operations are characterized by specifications of higher data rate demands and higher quality of experience (QoE) on the user side. This is paralleled by the low complexity and continued cost reduction of radio access and packet core on the network side. In order to offer better QoE for their subscribers, telecommunication companies have to instantiate predictive analytics to avoid network disruptions. This entails the automation of network operations and maintenance through data analytics. To this effect, consider the typical edge computing-enabled wireless network in FIGURE 15 below.

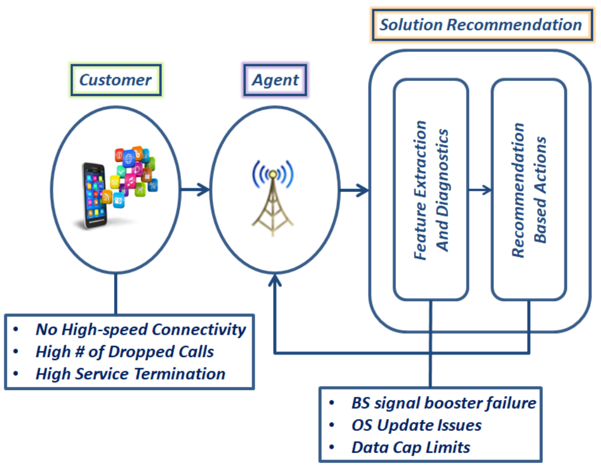

FIGURE 15. An edge computing and cache-enabled wireless network management framework.

In FIGURE 15 above, the general assumption is that the BS is co-located with an MEC server hosting a number of virtual machines (VMs) that account for its total computational resources [178]. Here, network data analysts can be able to visualize network performance in terms of KPIs. This kind of radio network performance is reported continuously through the radio network information service (RNIS) to the data center [179]. In case network performance has degraded to inconvenience network subscribers, urgent and prompt attention might be required. The grievances from the subscribers may be of poor service, resulting to problems like: (i) poor network connectivity, (ii) high number of dropped calls, and (iii) high service termination - to name just a few. From a technical perspective, these complaints are not different from one another, but might require different technical solutions. However, from a customer service perspective, these similarities and/or differences might not be clear, hence proper analytics are needed. For example, using data visualization techniques, the best possible diagnosis might be (i) signal booster failure, (ii) operating update issues, and (iii) contract data cap limits.

Assuming that the provider's customer service department receives a large volume of such grievance from already impatient subscribers in real-time. Taking the subscribers' impatience into account, the process of going through such massive data, extracting patterns, diagnosing the problems, and assigning field technicians to the different sites might be labor intensive and time consuming. Therefore, the whole process can be automated as follows:

1) DATA MINING AND INFORMATION EXTRACTION

Using data mining techniques such as clustering algorithms similar complaints can be grouped together for collective diagnosis. It must, however, be noted that the grievances received from the subscribers might be in text format, some preliminary pre-processing might be required in order to be a data set that is ready for processing and analysis. For the preliminary pre-processing step, a natural language processing and lexicon processing algorithm might be used to ''mine'' similar text structures. After grouping similar textual information using classification algorithms to observe some correlations in the text data for further processing.

2) DATA ANALYTICS AND DIAGNOSTICS

Using data mining techniques, a lot of information can be extracted, classified in terms of their similarities and differences and diagnostic techniques applied in each cluster of information to diagnose problems. In the diagnostic step, the classified data is processed in order to obtain a good diagnostic visibility of the network problems. From the customer service side, this visibility can assist the service providers with information such as regions of the network, BS sites, user behavior and applications. Using this information, diagnosis of what the real problems might be can be performed.

3) SOLUTION RECOMMENDATION

Within a short space of time potential solutions are evaluated and the best one is recommended and commissioned through the use of virtualized functions. For example, problems such as software maintenance and upgrades can be virtualized instead of allocating a technician on site. In this case, the software agent runs a solution recommendation module (SRM) and sends recommendations to the decision module, which proposes a network function virtualization (NFV) to the relevant BS site [180].

However, due to the nature of the diagnosed problems, the recommendation processes differ from one another due to the difference in user behavior in different network regions. As a result, the agent that handles the recommendation process must reside at the MEC in order to dynamically launch tasks of relevant VM instances to resolve dynamic problems. Therefore, the VM launching can be managed using a DRL strategy as contemplated in [181].

D. ENABLING QUANTUM MACHINE LEARNING IN URBAN COMPUTING

The increasing development of ubiquitous technologies are producing a wealth of information, reflecting different features of our lives over the last decade. Quantum computing (QC) and AI, combined together into quantum machine learning (QML), can revolutionize urban computing. The term ''Urban Computing'' is framed by the aspirational narratives surrounding smart cities and the technologies developed for its operation [182]. The fact that technology has truly become pervasive implies that processes have to shift from single systems to large-scale heterogeneous systems. This heterogeneity involves many devices and individuals collaborating over different spatial and temporal scales to cope with the dictates of the IoT [183]. With the dictates of the IoT continuing to demand more efficiency and reliability in connection and communication, AI is forced to push its boundaries towards cyber-physical convergence. Cyber-physical convergence means that the field of mobile and wireless communications has become an inter-disciplinary field [184]. This inter-disciplinary field is shaped by at least five main interacting dimensions that link the technological perspective closely to the social, economic and cognitive sciences. As a result, the traditional deep architectures discussed above may not be able to handle important processes such as control and prediction. Since urban computing applications require instantaneous services, this led to ML techniques being considered as potential enablers of this emerging paradigm. However, with all these interacting dimensions, the mobile and wireless network environment can now be viewed as a quantum state [185]. Thus, the technological drivers that could facilitate such a perspective change are well known - QC and QML.

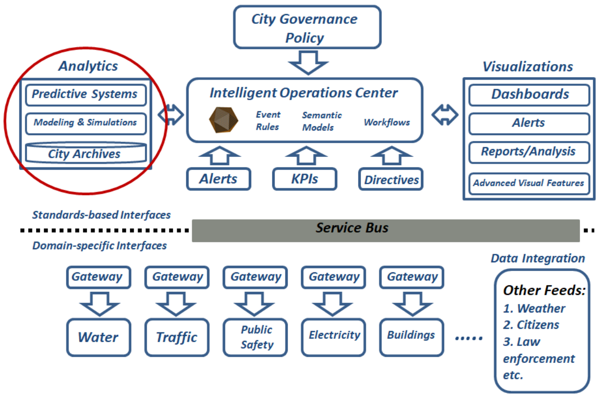

FIGURE 16. An illustration of the urban computing hierarchy.

Unlike the traditional ML techniques, QC and QML and their synergies with communication networks offer a radically different model of computation for engineering control and prediction. Due to their capabilities in combining database techniques with ML algorithms in data mining tasks, QC and QML are the prospective enablers of urban computing. However, in order to unlock the power of knowledge from data across different domains, it is the data analytics layer of the urban computing hierarchy that needs this improvement. For the scope of this article, focus is placed only on the data analytics module of the urban computing hierarchy, which is shown on the left-hand side of FIGURE 16 above. At the data analytics layer of the urban computing hierarchy, a diversity of data mining models and AI strategies are required to cope with the "Tsunami" of urban data. The reason for this lies in the fact that the data analytics layer adapts basic data mining and ML models such as, modeling and simulations to enable predictive analytics. These include the steps of data collection, data classification, and data visualization, as discussed below.

1) DATA COLLECTION

The digital traces emanating through human daily interactions with pervasive computing devices are valuable sources of data for capturing the operational pulses of the city in an astonishing degree of temporal and spatial details. The increasing use of ubiquitous devices, such as mobile phones and vehicle navigation systems, creates a new sensing theory in which humans serve as distributed sensors within the city that can be used to address urban dynamics towards achieving smart cities. The development of smartphones as sensing platforms, with various sensors such as GPS, accelerometers, microphones, as well as cameras, have accelerated this progress. If collected well, these raw data can be useful in developing data analytical tools in order to improve the understanding of urban dynamics and consequently make urban systems more efficient.

2) DATA CLASSIFICATION

In this way, classification models, such as quantum-inspired classifiers, can be utilized to work on the data collected from all these sensors. In order to enable urban systems to better understand both human actions and the environment, the classical data (X) can be mapped onto quantum data ($|\psi_X\rangle$). For example, in the case of quantum-inspired classification, a model is trained using classical ML techniques such as supervised learning. However, in the case of quantum kernel classifiers, the model is kernel-ed on a quantum device, then measurements are performed to convert the output labels back to classical [186].

3) DATA VISUALIZATION

Due to the massive and dynamic nature of city life, the properties of the classified data, i.e., spatio-temporal data, may consist of multiple disparate datasets, such as spatial distance, spatial hierarchy, temporal closeness, period and trend. Visualizing these multiple datasets using traditional ML techniques presents itself as a huge challenge. One of the main challenges regards the modeling of human activity, which is a key obstacle in differentiating contextual conditions and user characteristics encountered in a large-scale sensing environment such as this one. In order to address this obstacle, the discriminative features in the data from the classifiers need to be extracted in order to recognize different human activities, varying from user to user. In order to be able to fuse the knowledge from this cross-domain data fusion methods such as multi-view-based, probabilistic dependency-based, similarity-based, as well as transfer-learning-based data fusion techniques can be used [187]. QML techniques have disrupted the field of AI with its capability in dealing with probabilistic states using new ways of coding and new algorithms for information processing.

REFERENCES

[1] M. H. Alsharif, A. H. Kelechi, M. A. Albreem, S. A. Chaudhry, M. S. Zia, and S. Kim, "Sixth generation (6G) wireless networks: Vision, research activities, challenges and potential solutions," *Symmetry*, vol. 12, no. 4, p. 676, Apr. 2020.

[2] R. Chávez-Santiago, M. Szydełko, A. Kliks, F. Foukalas, Y. Haddad, K E. Nolan, M. Y. Kelly, M. T. Masonta, and I. Balasingham, "5G: The convergence of wireless communications," *Wireless Pers. Commun.*, vol. 83, no. 3, pp. 1617–1642, Mar. 2015.

[3] H. Yu, L. Gao, Y. Li, X. Gan, X. Wang, Y. Xu, W. Chen, and A. V. Vasilakos, "Information sharing in spectrum auction for dynamic spectrum access," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Honolulu, HI, USA, Nov. 2009, pp. 1–5.

[4] P. Steenkiste, D. Sicker, G. Minden, and D. Raychaudhuri "Future directions in cognitive radio network research," in *Proc. NSF Workshop Rep.*, Mar. 2009, vol. 4, no. 1, pp. 1–2.

[5] *ITU-T Recommendation G.114 One-way Transmission Time*, document ITU G.114, 2003.

[6] S. Haykin, "Cognitive radio: Brain-empowered wireless communications," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 2, pp. 201–220, Feb. 2005.

[7] M. van Steen and A. S. Tanenbaum, "A brief introduction to distributed systems," *Computing*, vol. 98, no. 10, pp. 967–1009, Oct. 2016.

[8] B. S. Awoyemi, B. T. J. Maharaj, and A. S. Alfa, "Solving resource allocation problems in cognitive radio networks: A survey," *EURASIP J. Wireless Commun. Netw.*, vol. 2016, no. 1, pp. 1–14, Jul. 2016.

[9] S.-Y. Lien, K.-C. Chen, Y.-C. Liang, and Y. Lin, "Cognitive radio resource management for future cellular networks," *IEEE Wireless Commun.*, vol. 21, no. 1, pp. 70–79, Feb. 2014.

[10] T. O. Olwal, K. Djouani, and A. M. Kurien, "A survey of resource management toward 5G radio access networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1656–1686, 3rd Quart., 2016.

[11] *A Technology Vision*, White Paper. Accessed: May 11, 2020. [Online]. Available: https://www.huawei.com/5gwhitepaper/

[12] Z. Zhang, K. Long, and J. Wang, "Self-organization paradigms and optimization approaches for cognitive radio technologies: A survey," *IEEE Wireless Commun.*, vol. 20, no. 2, pp. 36–42, Apr. 2013.

[13] F. Ahmed, "Self-organization: A perspective on applications in the Internet of Things," in *Natural Computing for Unsupervised Learning*. Cham, Switzerland: Springer, 2019, pp. 51–64.

[14] Q. Zhang, A. Kokkeler, and G. Smit, "A reconfigurable radio architecture for cognitive radio in emergency networks," in *Proc. Eur. Conf. Wireless Technol.*, Manchester, U.K., Sep. 2006, pp. 35–38.

[15] C. Moy, "High-level design approach for the specification of cognitive radio equipments management APIs," *J. Netw. Syst. Manage.*, vol. 18, no. 1, pp. 64–96, Mar. 2010.

[16] L. Gavrilovska, V. Atanasovski, I. Macaluso, and L. A. DaSilva, "Learning and reasoning in cognitive radio networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 1761–1777, 4th Quart., 2013.

[17] T. E Bogale, X. Wang, and L. B. Le, "Machine intelligence techniques for next-generation context-aware wireless networks," Jan. 2018, *arXiv:1801.04223*. [Online]. Available: http://arxiv.org/abs/1801.04223

[18] M. Matinmikko, M. Mustonen, T. Rauma, and J. D. Ser, "Decision-making system for obtaining spectrum availability information in opportunistic networks," in *Proc. 4th Int. Conf. Cognit. Radio Adv. Spectr. Manage. (CogART)*, New York, NY, USA, 2011, p. 13.

[19] S. S. Alam, L. Marcenaro, and C. S. Regazzoni, "Opportunistic spectrum sensing and transmissions," in *Cognitive Radio and Interference Management: Technology and Strategy*. Hershey, PA, USA: IGI Global, 2013, pp. 1–28.

[20] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Mobile edge computing and networking for green and low-latency Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 39–45, May 2018.

[21] D. Giusto, A. Iera, G. Morabito, and L. Atzori, Eds., *The Internet of Things: 20th Tyrrhenian Workshop on Digital Communications*. Springer, Mar. 2010.

[22] S. Chen, H. Wen, J. Wu, J. Chen, W. Liu, L. Hu, and Y. Chen, "Physical-layer channel authentication for 5G via machine learning algorithm," *Wireless Commun. Mobile Comput.*, vol. 2018, Oct. 2018, Art. no. 6039878.

[23] B. Dickson, "Decentralizing IoT networks through blockchain," TechCrunch, 2016. Accessed: May 17, 2020. [Online]. Available: https://techcrunch.com/2016/06/28/decentralizing-iot-networks-through-blockchain/

[24] J. Kalliovaara, T. Jokela, H. Kokkinen, J. Paavola, and S. S. Moghaddam, "Licensed shared access evolution to provide exclusive and dynamic shared spectrum access for novel 5G use cases," *Cogn. Radio 4G/5G Wireless Commun. Syst.*, vol. 3, pp. 55–72, Nov. 2018.

[25] Z. Qin, H. Ye, G. Y. Li, and B. H. F. Juang, "Deep learning in physical layer communications," *IEEE Wireless Commun.*, vol. 26, no. 2, pp. 93–99, Mar. 2019.

[26] P. H. Kuo, H. T. Kung, and P. A. Ting, "Compressive sensing based channel feedback protocols for spatially-correlated massive antenna arrays," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Paris, France, Apr. 2012, pp. 492–497.

[27] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 7, pp. 1029–1046, Sep. 2009.

[28] V. De Oliveira, "Poisson kriging: A closer investigation," *Spatial Statist.*, vol. 7, pp. 1–20, Feb. 2014.

[29] C. Song, T. Ristenpart, and V. Shmatikov, "Machine learning models that remember too much," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 587–601.

[30] J. Jalden and P. Elia, "Sphere decoding complexity exponent for decoding full-rate codes over the quasi-static MIMO channel," *IEEE Trans. Inf. Theory*, vol. 58, no. 9, pp. 5785–5803, Sep. 2012.

[31] A. Stopczynski, V. Sekara, P. Sapiezynski, A. Cuttone, M. M. Madsen, J. E. Larsen, and S. Lehmann, "Measuring large-scale social networks with high resolution," *PLoS ONE*, vol. 9, no. 4, Apr. 2014, Art. no. e95978.

[32] H. T. Siegelmann and E. D. Sontag, "On the computational power of neural networks," in *Proc. 5th ACM Ann. Works. Comp. Learn. Theory*, Pittsburgh, PA, USA, Jul. 1992, pp. 440–449.

[33] O. Simeone, "A brief introduction to machine learning for engineers," *Found. Trends Signal Process.*, vol. 12, nos. 3–4, pp. 200–431, Aug. 2018.

[34] T. Nakano, "Molecular communication: A 10 year retrospective," *IEEE Trans. Mol., Biol. Multi-Scale Commun.*, vol. 3, no. 2, pp. 71–78, Jun. 2017.

[35] N. N. Srinidhi, S. M. D. Kumar, and K. R. Venugopal, "Network optimizations in the Internet of Things: A review," *Eng. Sci. Technol., Int. J.*, vol. 22, no. 1, pp. 1–21, Feb. 2018.

[36] P. Barry and P. Crowley, *Modern Embedded Computing: Designing Connected, Pervasive, Media-Rich Systems*. Amsterdam, The Netherlands: Elsevier, Feb. 2012, pp. 477–496.

[37] Y. Roh, G. Heo, and S. E. Whang, "A survey on data collection for machine learning: A big data–AI integration perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1328–1347, Apr. 2021.

[38] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Pérolat, D. Silver, and T Graepel, "A unified game-theoretic approach to multiagent reinforcement learning," in *Proc. Adv. Neur. Inf. Proc. Syst.*, 2017, pp. 4190–4203.

[39] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, Nov. 2018, Art. no. e00938.

[40] E. Guresen and G. Kayakutlu, "Definition of artificial neural networks with comparison to other networks," *Procedia Comp. Sci.*, vol. 3, pp. 426–433, Jan. 2011.

[41] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," 2018, *arXiv:1811.03378*. [Online]. Available: http://arxiv.org/abs/1811.03378

[42] R. C. Gonzalez, "Deep convolutional neural networks [Lecture Notes]," *IEEE Signal Process. Mag.*, vol. 35, no. 6, pp. 79–87, Nov. 2018.

[43] Y. Altun, M. Johnson, and T. Hofmann, "Investigating loss functions and optimization methods for discriminative learning of label sequences," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Sapporo, Japan, 2003, pp. 52–145.

[44] X. Zhou *et al.*, "Machine learning and cognitive technology for intelligent wireless networks," *ArXiv*, vol. abs/1710.11240, Oct. 2017.

[45] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 31–119, Jan. 2018.

[46] H. Agirman-Tosun, Y. Liu, A. M. Haimovich, O. Simeone, W. Su, J. Dabin, and E. Kanterakis, "Modulation classification of MIMO-OFDM signals by independent component analysis and support vector machines," in *Proc. Conf. Rec. 45th Asilomar Conf. Signals, Syst. Comput. (ASILOMAR)*, Pacific Grove, CA, USA, Nov. 2011, pp. 1903–1907.

[47] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Sapporo, Japan, Jul. 2017, pp. 1–6.

[48] A. Balatsoukas-Stimming, "Non-linear digital self-interference cancellation for in-band full-duplex radios using neural networks," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Kalamata, Greece, Jun. 2018, pp. 1–5.

[49] N. Strodthoff, B. Göktepe, T. Schierl, C. Hellge, and W. Samek, "Enhanced machine learning techniques for early HARQ feedback prediction in 5G," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 11, pp. 2573–2587, Nov. 2019.

[50] M. Chen, W. Saad, C. Yin, and M. Debbah, "Echo state networks for proactive caching in cloud-based radio access networks with mobile users," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3520–3535, Jun. 2017.

[51] M. Saber, A. E. Rharras, R. Saadane, A. Chehri, N. Hakem, and H. A. Kharraz, "Spectrum sensing for smart embedded devices in cognitive networks using machine learning algorithms," *Procedia Comp. Sci.*, vol. 176, pp. 2404–2413, Jan. 2020.

[52] S. K. Singh, M. M. Salim, J. Cha, Y. Pan, and J. H. Park, "Machine learning-based network sub-slicing framework in a sustainable 5G environment," *Sustainability*, vol. 12, pp. 1–24, Aug. 2020.

[53] M. H. Abidi, H. Alkhalefah, K. Moiduddin, M. Alazab, M. K. Mohammed, W. Ameen, and T. R. Gadekallu, "Optimal 5G network slicing using machine learning and deep learning concepts," *Comput. Standards Interfaces*, vol. 76, Jun. 2021, Art. no. 103518.

[54] N. Salhab, R. Rahim, R. Langar, and R. Boutaba, "Machine learning based resource orchestration for 5G network slices," in *Proc. IEEE GLOBECOM*, Waikoloa, HI, USA, Dec. 2019, pp. 1–6.

[55] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, Nov. 2018.

[56] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," 2017, *arXiv:1708.05866*. [Online]. Available: http://arxiv.org/abs/1708.05866

[57] Z. Akbari and R. Unland, "A novel heterogeneous swarm reinforcement learning method for sequential decision making problems," *Mach. Learn. Knowl. Extraction*, vol. 1, no. 2, pp. 590–610, Apr. 2019.

[58] P. Dayan and K. C. Berridge, "Model-based and model-free Pavlovian reward learning: Revaluation, revision, and revelation," *Cognit., Affect., Behav. Neurosci.*, vol. 14, no. 2, pp. 473–492, Jun. 2014.

[59] R. Cimurs, J. H. Lee, and I. H. Suh, "Goal-oriented obstacle avoidance with deep reinforcement learning in continuous action space," *Electronics*, vol. 9, no. 3, p. 411, Feb. 2020.

[60] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neu. Inf. Proc. Syst. (NIPS)*, 2000, pp. 1057–1063.

[61] S. Sugiyama, *Statistical Reinforcement Learning: Modern Machine Learning Approaches*. Boca Raton, FL, USA: CRC Press, 2015.

[62] E. Altman, *Constrained Markov Decision Processes*. Boca Raton, FL, USA: CRC Press, Mar. 1999.

[63] S. Narvekar and P. Stone, "Learning curriculum policies for reinforcement learning," in *Proc. 18th Int. Conf. Auto. Agents Multiagent Syst. (AAMAS)*, Montreal, QC, Canada, May 2019, pp. 1–9.

[64] M. van Otterlo and M. Wiering, "Reinforcement learning and Markov decision processes," in *Reinforcement Learning*. Berlin, Germany: Springer, 2012, pp. 3–42.

[65] C. Boutilier, "Planning, learning and coordination in multiagent decision processes," in *Proc. 6th Conf. Theor. Aspects Rationality Knowl. (TARK)*, Mar. 1996, pp. 195–210.

[66] E. Solan and B. Ziliotto, "Stochastic games with signals," in *Advances in Dynamic and Evolutionary Games*. Cham, Switzerland: Birkhäuser, 2016, pp. 77–94.

[67] D. V. Djonin and V. Krishnamurthy, "Q-learning algorithms for constrained Markov decision processes with randomized monotone policies: Application to MIMO transmission control," *IEEE Trans. Signal Process.*, vol. 55, no. 5, pp. 2170–2181, May 2007.

[68] I. Markovsky, T. Liu, and A. Takeda, "Subspace methods for multichannel sum-of-exponentials common dynamics estimation," in *Proc. IEEE 58th Conf. Decis. Control (CDC)*, Nice, France, Dec. 2019, pp. 2672–2675.

[69] C. Amato, G. Chowdhary, A. Geramifard, N. K. Ure, and M. J. Kochenderfer, "Decentralized control of partially observable Markov decision processes," in *Proc. IEEE 52nd Annu. Conf. Decis. Control (CDC)*, Florence, Italy, Dec. 2013, pp. 2398–2405.

[70] E. Altman, "Constrained Markov decision processes with total cost criteria: Occupation measures and primal LP," *Math. Methods Oper. Res.*, vol. 43, no. 1, pp. 45–72, Feb. 1996.

[71] D. Fiems and T. Phung-Duc, "Light-traffic analysis of random access systems without collisions," *Ann. Oper. Res.*, vol. 277, no. 2, pp. 311–327, Jun. 2019.

[72] V. Pla, A. S. Alfa, J. Martinez-Bauset, and V. Casares-Giner, "Discrete-time analysis of cognitive radio networks with nonsaturated source of secondary users," *Wireless Commun. Mobi. Comp.*, vol. 2019, Jan. 2019, Art. no. 7367028.

[73] C. L. Russell, "5 G wireless telecommunications expansion: Public health and environmental implications," *Environ. Res.*, vol. 165, pp. 484–495, Aug. 2018.

[74] M. B. A. Sadi and A. Nadia, "Call admission scheme for multidimensional traffic assuming finite handoff user," *J. Comput. Netw. Commun.*, vol. 2017, pp. 1–5, Mar. 2017.

[75] S. Chobsri, W. Sumalai, and W. Usaha, "A parametric POMDP framework for efficient data acquisition in error prone wireless sensor networks," in *Proc. 4th Int. Symp. Wireless Pervas. Comput.*, Dublin, Ireland, Feb. 2009, pp. 1–5.

[76] Y. E. Sagduyu and A. Ephremides, "Power control and rate adaptation as stochastic games for random access," in *Proc. 42nd IEEE Int. Conf. Decis. Control*, Maui, HI, USA, Dec. 2003, pp. 4202–4207.

[77] E. Altman, "Applications of dynamic games in queues," in *Advances in Dynamic Games*. Boston, MA, USA: Birkhäuser, 2005, pp. 309–342.

[78] N. Mastronarde and M. van der Schaar, "Joint physical-layer and system-level power management for delay-sensitive wireless communications," *IEEE Trans. Mobile Comput.*, vol. 12, no. 4, pp. 694–709, Apr. 2012.

[79] L. I. Sennott, "Constrained average cost Markov decision chains," *Probab. Eng. Inf. Sci.*, vol. 7, no. 1, pp. 69–83, Jan. 1993.

[80] A. Pietrabissa, "A reinforcement learning approach to call admission and call dropping control in links with variable capacity," *Eur. J. Control*, vol. 17, no. 1, pp. 89–103, Jan. 2011.

[81] T. Le-Ngoc and K. T. Phan, "Joint data admission control and power allocation over fading channel under average delay constraint," in *Radio Resource Allocation Over Fading Channels Under Statistical Delay Constraints*. Cham, Switzerland: Springer, 2017, pp. 29–41.

[82] M. M. Sande, M. C. Hlophe, and B. T. Maharaj, "Instantaneous load-based user association in multi-hop IAB networks using reinforcement learning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Taipei, Taiwan, Dec. 2020, pp. 1–6.

[83] F. A. Aoudia and J. Hoydis, "End-to-end learning of communications systems without a channel model," Apr. 2018, *arXiv:1804.02276*. [Online]. Available: http://arxiv.org/abs/1804.02276

[84] F. Afghah, A. Razi, and A. Abedi, "Stochastic game theoretical model for packet forwarding in relay networks," *Telecom. Syst.*, vol. 52, no. 4, pp. 1877–1893, Apr. 2013.

[85] A. C. Mendes, C. H. P. Augusto, M. W. R. da Silva, R. M. Guedes, and J. F. de Rezende, "Channel sensing order for cognitive radio networks using reinforcement learning," in *Proc. IEEE 36th Conf. Local Comput. Netw.*, Bonn, Germany, Oct. 2011, pp. 546–553.

[86] B. Jaishanthi, E. N. Ganesh, and D. Sheela, "Priority-based reserved spectrum allocation by multi-agent through reinforcement learning in cognitive radio network," *Automatika, J. Control, Meas., Electron., Comput. Commun.*, vol. 60, no. 5, pp. 564–569, Oct. 2019.

[87] S.-J. Jang, C.-H. Han, K.-E. Lee, and S.-J. Yoo, "Reinforcement learning-based dynamic band and channel selection in cognitive radio ad-hoc networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, pp. 1–25, Dec. 2019.

[88] M. S. Shamaee, M. E. Shiri, and M. Sabaei, "A reinforcement learning based routing in cognitive radio networks for primary users with multi-stage periodicity," *Wireless Personal Commun.*, vol. 101, no. 1, pp. 465–490, Jul. 2018.

[89] Y. Kim, S. Kim, and H. Lim, "Reinforcement learning based resource management for network slicing," *Appl. Sci.*, vol. 9, pp. 1–17, Jun. 2019.

[90] V. Sciancalepore, X. Costa-Perez, and A. Banchs, "RL-NSB: Reinforcement learning-based 5G network slice broker," *IEEE/ACM Trans. Netw.*, vol. 27, no. 4, pp. 1543–1557, Aug. 2019.

[91] Y. Liu, J. Ding, and X. Liu, "A constrained reinforcement learning based approach for network slicing," in *Proc. IEEE 28th Int. Conf. Netw. Protocols (ICNP)*, Oct. 2020, pp. 1–6.

[92] M. R. Raza, C. Natalino, P. Öhlen, L. Wosinska, and P. Monti, "Reinforcement learning for slicing in a 5G flexible RAN," *J. Lightw. Technol.*, vol. 37, no. 20, pp. 5161–5169, Oct. 2019.

[93] F. Shah-Mohammadi and A. Kwasinski, "Neural network cognitive engine for autonomous and distributed underlay dynamic spectrum access," Jun. 2018, *arXiv:1806.11038*. [Online]. Available: http://arxiv.org/abs/1806.11038

[94] M. C. Hlophe and B. T. Maharaj, "QoE-driven resource allocation for SUs with heterogeneous traffic using deep reinforcement learning," in *Proc. 2nd IEEE WAC*, Pretoria, South Africa, Aug. 2019, pp. 1–5.

[95] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. ICML*, Jun. 2016, pp. 1995–2003.

[96] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[97] J. Ma, M. K. Yu, S. Fong, K. Ono, E. Sage, B. Demchak, R. Sharan, and T. Ideker, "Using deep learning to model the hierarchical structure and function of a cell," *Nature Methods*, vol. 15, no. 4, pp. 290–298, Apr. 2018.

[98] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition," Jun. 2016, *arXiv:1608.04636*. [Online]. Available: http://arxiv.org/abs/1608.04636

[99] T. Parr and J. Howard, "The matrix calculus you need for deep learning," Feb. 2018, *arXiv:1802.01528*. [Online]. Available: http://arxiv.org/abs/1802.01528

[100] L. Palagi, A. Pesyridis, E. Sciubba, and L. Tocci, "Machine learning for the prediction of the dynamic behavior of a small scale ORC system," *Energy*, vol. 166, pp. 72–82, Jan. 2019.

[101] M. M. Mia, S. K. Biswas, M C. Urmi, and A Siddique, "An algorithm for training multilayer perceptron (MLP) for image reconstruction using neural network without overfitting," *Int. J. Sci. Tech. Res.*, vol. 4, no. 2, pp. 271–275, Feb. 2015.

[102] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comp. Intel. Neurosci.*, vol. 2018, Feb. 2018, Art. no. 7068349, doi: 10.1155/2018/7068349.

[103] A. Sarkar, "Key generation and certification using multilayer perceptron in wireless communication (KGCMLP)," *Int. J. Secur., Privacy Trust Manage.*, vol. 1, no. 5, pp. 27–43, Oct. 2012.

[104] K. P. Bagadi and S. Das, "Multiuser detection in SDMA–OFDM wireless communication system using complex multilayer perceptron neural network," *Wireless Pers. Commun.*, vol. 77, no. 1, pp. 21–39, Jul. 2014.

[105] L. S. Ezema and C. I. Ani, "Artificial neural network approach to mobile location estimation in GSM network," *Int. J. Electron. Telecommun.*, vol. 63, no. 1, pp. 39–44, Mar. 2017.

[106] I. Rehman, M. Nasralla, and N. Philip, "Multilayer perceptron neural network-based QoS-aware, content-aware and device-aware QoE prediction model: A proposed prediction model for medical ultrasound streaming over small cell networks," *Electronics*, vol. 8, no. 2, p. 194, Feb. 2019.

[107] M. C. Hlophe and B. T. Maharaj, "Secondary user experience-oriented resource allocation in AI-empowered cognitive radio networks using deep neuroevolution," in *Proc. IEEE 91st Veh. Technol. Conf. (VTC-Spring)*, Antewerp, Belgium, May 2020, pp. 1–6.

[108] A. Oppermann, "Deep learning meets physics: Restricted Boltzmann machines part I: Theory behind restricted Boltzmann machines, a powerful tool for recomender systems," *Towards Data Sci.*, Apr. 2018. Accessed: Aug. 10, 2020. [Online]. Available: https://towardsdatascience.com/deep-learning-meets-physics-restricted-boltzmann-machines-part-i-6df5c4918c15

[109] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using restricted Boltzmann machine," in *Proc. Int. Conf. Int. Comp.* Berlin, Germany: Springer, Jul. 2012, pp. 17–22.

[110] M. C. Hlophe and S. B. T. Maharaj, "Spectrum occupancy reconstruction in distributed cognitive radio networks using deep learning," *IEEE Access*, vol. 7, pp. 14294–14307, 2019.

[111] S. Goudarzi, M. Kama, M. Anisi, S. Soleymani, and F. Doctor, "Self-organizing traffic flow prediction with an optimized deep belief network for internet of vehicles," *Sensors*, vol. 18, no. 10, p. 3459, Oct. 2018.

[112] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, "An overview and comparative analysis of recurrent neural networks for short term load forecasting," May 2017, *arXiv:1705.04378*. [Online]. Available: http://arxiv.org/abs/1705.04378

[113] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," Feb. 2014, *arXiv:1402.1128*. [Online]. Available: http://arxiv.org/abs/1402.1128

[114] J. Brownlee, "Gentle introduction to the Adam optimization algorithm for deep learning," *Mach. Learn. Mastery*, vol. 3, Jul. 2017.

[115] Y. H. Lee, D. J. M. Moss, J. Faraone, P. Blackmore, D. Salmond, D. Boland, and P. H. W. Leong, "Long short-term memory for radio frequency spectral prediction and its real-time FPGA implementation," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2018, pp. 1–9.

[116] L. Ale, N. Zhang, H. Wu, D. Chen, and T. Han, "Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5520–5530, Jun. 2019.

[117] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Commun. Mag.*, vol. 57, no. 6, pp. 114–119, Jun. 2019.

[118] M. C. Hlophe and B. T. Maharaj, "Optimization and learning in energy efficient resource allocation for cognitive radio networks," in *Proc. IEEE 89th Veh. Technol. Conf. (VTC-Spring)*, Apr. 2019, pp. 1–6.

[119] T. Dlamini, Á. F. Gambín, D. Munaretto, and M. Rossi, "Online supervisory control and resource management for energy harvesting BS sites empowered with computation capabilities," *Wireless Commun. Mobile Comput.*, vol. 2019, Feb. 2019, Art. no. 8593808.

[120] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features dimensionality reduction approaches for machine learning based network intrusion detection," *Electronics*, vol. 8, no. 3, p. 322, Mar. 2019.

[121] D. Berman, A. Buczak, J. Chavis, and C. Corbett, "A survey of deep learning methods for cyber security," *Information*, vol. 10, no. 4, p. 122, Apr. 2019.

[122] H. He, S. Jin, C.-K. Wen, F. Gao, G. Ye Li, and Z. Xu, "Model-driven deep learning for physical layer communications," Feb. 2018, *arXiv:1809.06059*. [Online]. Available: http://arxiv.org/abs/1809.06059

[123] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[124] A. Felix, S. Cammerer, S. Dorner, J. Hoydis, and S. Ten Brink, "OFDM-autoencoder for end-to-end learning of communications systems," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Calamata, Greece, Jun. 2018, pp. 1–5.

[125] H. Ye, G. Y. Li, B.-H.-F. Juang, and K. Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional GAN," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Abu Dhabi, UAE, Dec. 2018, pp. 1–5.

[126] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, Helsinki, Finland, Jun. 2008, pp. 1096–1103.

[127] K. I. Ahmed, H. Tabassum, and E. Hossain, "Deep learning for radio resource allocation in multi-cell networks," *IEEE Netw.*, vol. 33, no. 6, pp. 188–195, Nov. 2019.

[128] M. Zhang, L. Wang, Y. Feng, and H. Yin, "A spectrum sensing algorithm for OFDM signal based on deep learning and covariance matrix graph," *IEICE Trans. Commun.*, vol. E101.B, no. 12, pp. 2435–2444, 2018.

[129] Q. Cheng, Z. Shi, D. N. Nguyen, and E. Dutkiewicz, "Deep learning network based spectrum sensing methods for OFDM systems," Jul. 2018, *arXiv:1807.09414*. [Online]. Available: http://arxiv.org/abs/1807.09414

[130] K. Yang, Z. Huang, X. Wang, and X. Li, "A blind spectrum sensing method based on deep learning," *Sensors*, vol. 19, no. 2270, pp. 1–17, May 2019.

[131] M. C. Hlophe and B. T. Maharaj, "QoS provisioning and energy saving scheme for distributed cognitive radio networks using deep learning," *J. Commun. Netw.*, vol. 22, no. 3, pp. 185–204, Jun. 2020.

[132] A. Thantharate, R. Paropkari, V. Walunj, and C. Beard, "DeepSlice: A deep learning approach towards an efficient and reliable network slicing in 5G networks," in *Proc. IEEE 10th Annu. Ubiquitous Comput., Electron. Mobile Commun. Conf. (UEMCON)*, New York, NY, USA, Oct. 2019, pp. 762–767.

[133] X. Cheng, Y. Wu, G. Min, A. Y. Zomaya, and X. Fang, "Safeguard network slicing in 5G: A learning augmented optimization approach," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 7, pp. 1600–1613, Jul. 2020.

[134] B. M. Turner, J. Gao, S. Koenig, D. Palfy, and J. L. McClelland, "The dynamics of multimodal integration: The averaging diffusion model," *Psychonomic Bull. Rev.*, vol. 24, no. 6, pp. 1819–1843, Dec. 2017.

[135] D. Wang, B. Bai, W. Zhao, and Z. Han, "A survey of optimization approaches for wireless physical layer security," Jan. 2019, *arXiv:1901.07955*. [Online]. Available: http://arxiv.org/abs/1901.07955

[136] A. M. Hemeida, S. Alkhalaf, A. Mady, E. A. Mahmoud, M. E. Hussein, and A. M. B. Eldin, "Implementation of nature-inspired optimization algorithms in some data mining tasks," *Ain Shams Eng. J.*, vol. 11, no. 2, pp. 18–309, Jun. 2020.

[137] F. Alberge, "Deep learning constellation design for the AWGN channel with additive radar interference," *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1413–1423, Feb. 2019.

[138] D. M. Nguyen, R. Calderbank, and N. Deligiannis, "Geometric matrix completion with deep conditional random fields," Jan. 2019, *arXiv:1901.10429*. [Online]. Available: http://arxiv.org/abs/1901.10429

[139] F. Chazal and B. Michel, "An introduction to topological data analysis: Fundamental and practical aspects for data scientists," Oct. 2017, *arXiv:1710.04019*. [Online]. Available: http://arxiv.org/abs/1710.04019

[140] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2334–2360, 3rd Quart., 2019.

[141] S. Dara and P. Tumma, "Feature extraction by using deep learning: A survey," in *Proc. 2nd Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA)*, Coimbatore, India, Mar. 2018, pp. 1795–1801.

[142] V. Thakur and H. Sikarwar, "Deep learning feature extraction for handwritten keyword spotting in historical documents," in *Proc. 2nd Int. Conf. Emerg. Trends Eng. Appl. Sci. (ICETEAS)*, Jan. 2019, vol. 5, no. 1, pp. 11–15.

[143] H. B. Ahmad, "Ensemble classifier based spectrum sensing in cognitive radio networks," *Wireless Commun. Mobile Comput.*, vol. 2019, pp. 1–16, Jan. 2019.

[144] M. A. Alsheikh, D. Niyato, S. Lin, H.-P. Tan, and Z. Han, "Mobile big data analytics using deep learning and apache spark," *IEEE Netw.*, vol. 30, no. 3, pp. 22–29, Jun. 2016.

[145] N. V. Chawla and G. Karakoulas, "Learning from labeled and unlabeled data: An empirical study across techniques and domains," *J. Artif. Intell. Res.*, vol. 23, pp. 331–366, Mar. 2005.

[146] M. Grimaldi, V. Tenace, and A. Calimera, "Layer-wise compressive training for convolutional neural networks," *Future Internet*, vol. 11, no. 1, pp. 7–21, Jan. 2019.

[147] A. E. Eshratifar and M. Pedram, "Energy and performance efficient computation offloading for deep neural networks in a mobile cloud computing environment," in *Proc. Great Lakes Symp. VLSI*, Chicago, IL, USA, May 2018, pp. 111–116.

[148] S. K. Zhou, H. N. Le, K. Luu, H. V. Nguyen, and N. Ayache, "Deep reinforcement learning in medical imaging: A literature review," Mar. 2021, *arXiv:2103.05115*. [Online]. Available: http://arxiv.org/abs/2103.05115

[149] H. Li, R. Cai, N. Liu, X. Lin, and Y. Wang, "Deep reinforcement learning: Algorithm, applications, and ultra-low-power implementation," *Nano Commun. Netw.*, vol. 16, pp. 81–90, Jun. 2018.

[150] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Found. Trends Mach. Learn.*, vol. 11, nos. 3–4, pp. 219–354, 2018.

[151] R. Li, Z. Zhao, Q Sun, I. Chih-Lin, C. Yang, X. Chen, M. Zhao, and H. Zhang, "Deep reinforcement learning for resource management in network slicing," *IEEE Access*, vol. 6, pp. 74429–74441, 2018.

[152] T. T. Anh, N. C. Luong, D. Niyato, Y.-C. Liang, and D. I. Kim, "Deep reinforcement learning for time scheduling in RF-powered backscatter cognitive radio networks," Oct. 2018, *arXiv:1810.04520*. [Online]. Available: http://arxiv.org/abs/1810.04520

[153] R. Liu and J. Zou, "The effects of memory replay in reinforcement learning," in *Proc. 56th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Champaign, IL, USA, Oct. 2018, pp. 478–485.

[154] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. 15th ACM Workshop Hot Topics Netw.*, Atlanta, GA, USA, Nov. 2016, pp. 50–56.

[155] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," Dec. 2013, *arXiv:1312.5602*. [Online]. Available: http://arxiv.org/abs/1312.5602

[156] P. Yang, L. Li, J. Yin, H. Zhang, W. Liang, W. Chen, and Z. Han, "Dynamic spectrum access in cognitive radio networks using deep reinforcement learning and evolutionary game," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Beijing, China, Aug. 2018, pp. 405–409.

[157] Y. Du, Y. Xu, L. Xue, L. Wang, and F. Zhang, "An energy-efficient cross-layer routing protocol for cognitive radio networks using apprenticeship deep reinforcement learning," *Energies*, vol. 12, no. 14, p. 2829, Jul. 2019.
[158] X. Meng, H. Inaltekin, and B. Krongold, "Deep reinforcement learning-based power control in full-duplex cognitive radio networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2018, pp. 1–7.
[159] W. Lei, Y. Ye, and M. Xiao, "Deep reinforcement learning based spectrum allocation in integrated access and backhaul networks," Apr. 2020, *arXiv:2004.13133*. [Online]. Available: http://arxiv.org/abs/2004.13133
[160] F. Shah-Mohammadi and A. Kwasinski, "Deep reinforcement learning approach to QoE-driven resource allocation for spectrum underlay in cognitive radio networks," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Topeka, KS, USA, May 2018, pp. 1–6.
[161] Q. Liu, T. Han, and E. Moges, "EdgeSlice: Slicing wireless edge computing network with decentralized deep reinforcement learning," Mar. 2020, *arXiv:2003.12911*. [Online]. Available: http://arxiv.org/abs/2003.12911
[162] H. Peng and X. Shen, "Deep reinforcement learning based resource management for multi-access edge computing in vehicular networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2416–2428, Oct. 2020.
[163] A. Nassar and Y. Yilmaz, "Deep reinforcement learning for adaptive network slicing in 5G for intelligent vehicular systems and smart cities," Oct. 2020, *arXiv:2010.09916*. [Online]. Available: http://arxiv.org/abs/2010.09916
[164] J. Koo, V. B. Mendiratta, M. R. Rahman, and A. Walid, "Deep reinforcement learning for network slicing with heterogeneous resource requirements and time varying traffic dynamics," in *Proc. 15th Int. Conf. Netw. Service Manage. (CNSM)*, Halifax, NS, Canada, Oct. 2019, pp. 1–5.
[165] T. Li, X. Zhu, and X. Liu, "An end-to-end network slicing algorithm based on deep Q-learning for 5G network," *IEEE Access*, vol. 8, pp. 122229–122240, 2020.
[166] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, "GAN-powered deep distributional reinforcement learning for resource management in network slicing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 334–349, Feb. 2020.
[167] X. Chen, Z. Zhao, C. Wu, M. Bennis, H. Liu, Y. Ji, and H. Zhang, "Multi-tenant cross-slice resource orchestration: A deep reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2377–2392, Oct. 2019.
[168] H. Wang, Y. Wu, G. Min, J. Xu, and P. Tang, "Data-driven dynamic resource scheduling for network slicing: A deep reinforcement learning approach," *Inf. Sci.*, vol. 498, pp. 106–116, Sep. 2019.
[169] F. Wei, G. Feng, Y. Sun, Y. Wang, S. Qin, and Y.-C. Liang, "Network slice reconfiguration by exploiting deep reinforcement learning with large action space," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 4, pp. 2197–2211, Dec. 2020.
[170] Q. Liu, T. Han, N. Zhang, and Y. Wang, "DeepSlicing: Deep reinforcement learning assisted resource allocation for network slicing," Aug. 2020, *arXiv:2008.07614*. [Online]. Available: http://arxiv.org/abs/2008.07614
[171] R. Li, C. Wang, Z. Zhao, R. Guo, and H. Zhang, "The LSTM-based advantage actor-critic learning for resource management in network slicing with user mobility," *IEEE Commun. Lett.*, vol. 24, no. 9, pp. 2005–2009, Sep. 2020.
[172] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno, and H. Mineno, "Flexible resource block allocation to multiple slices for radio access network slicing using deep reinforcement learning," *IEEE Access*, vol. 8, pp. 68183–68198, 2020.
[173] R. G. Duffett, "Influence of social media marketing communications on young consumers' attitudes," *Young Consumers*, vol. 18, no. 1, pp. 19–39, Apr. 2017.
[174] M. S. Hadi, A. Q. Lawey, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Big data analytics for wireless and wired network design: A survey," *Comput. Netw.*, vol. 132, pp. 99–180, 26 Feb. 2018.
[175] N. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy preservation in federated learning: An insightful survey from the GDPR perspective," Nov. 2020, *arXiv:2011.05411*. [Online]. Available: http://arxiv.org/abs/2011.05411
[176] J. Konečný, H. Brendan McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: http://arxiv.org/abs/1610.02527
[177] G. de Vinzelles. (Mar. 16, 2018). *Federated Learning, A Step Closer Towards Confidential AI*. [Online]. Available: https://hackernoon.com/federated-learning-a-step-closer-towards-confidential-ai-7ac4afa9b437
[178] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, and A. Nea, "Mobile-edge computing introductory technical white paper," Mobile-Edge Comput. (MEC) Ind. Initiative, White Paper, Sep. 2014, pp. 854–864, no. 2014.
[179] S. Peng, J. O. Fajardo, P. S. Khodashenas, B. Blanco, F. Liberal, C. Ruiz, C. Turyagyenda, M. Wilson, and S. Vadgama, "QoE-oriented mobile edge service management leveraging SDN and NFV," *Mobile Inf. Syst.*, vol. 2017, pp. 1–14, Jan. 2017.
[180] M. Touloupos, E. Kapassa, D. Kyriazis, and K. Christodoulou, "Test recommendation for service validation in 5G networks," in *Proc. 16th Eur., Medit., Middle Eastern Conf. Inf. Syst.*, Dubai, UAE, Dec. 2019, pp. 50–139.
[181] R. S. Alonso, I. Sittón-Candanedo, R. Casado-Vara, J. Prieto, and J. M. Corchado, "Deep reinforcement learning for the management of software-defined networks and network function virtualization in an edge-IoT architecture," *Sustainability*, vol. 12, no. 14, p. 5706, Jul. 2020.
[182] S. Marvin and A. Luque-Ayala, "Urban operating systems: Diagramming the city," *Int. J. Urban Regional Res.*, vol. 41, no. 1, pp. 84–103, Jan. 2017.
[183] L. A. Amaral, E. de Matos, R. T. Tiburski, F. Hessel, W. T. Lunardi, and S. Marczak, "Middleware technology for IoT systems: Challenges and perspectives toward 5G," in *Internet of Things (IoT) in 5G Mobile Technologies*. Cham, Switzerland: Springer, 2016, pp. 333–367.
[184] B. Bordel, R. Alcarria, T. Robles, and D. Martín, "Cyber–physical systems: Extending pervasive sensing from control theory to the Internet of Things," *Pervas. Mob. Comp.*, vol. 40, pp. 156–184, Sep. 2017.
[185] S. J. Nawaz, S. K. Sharma, S. Wyne, M. N. Patwary, and M Asaduzzaman, "Quantum machine learning for 6G communication networks: State-of-the-art and vision for the future," *IEEE Access*, vol. 7, pp. 46317–46350, 2019.
[186] Z. Abohashima, M. Elhosen, E. H. Houssein, and W. M. Mohamed, "Classification with quantum machine learning: A survey," Jun. 2020, *arXiv:2006.12270*. [Online]. Available: http://arxiv.org/abs/2006.12270
[187] J. Qi, P. Yang, L. Newcombe, X. Peng, Y. Yang, and Z. Zhao, "An overview of data fusion techniques for Internet of Things enabled physical activity recognition and measure," *Inf. Fusion*, vol. 55, pp. 269–280, Mar. 2020.

MDUDUZI C. HLOPHE (Member, IEEE) received the Ph.D. degree in electronic engineering in the area of wireless communications from the University of Pretoria, South Africa, in 2020. He is currently a Postdoctoral Fellow with the Broadband Wireless Multimedia Communications (BWMC), Department of Electrical, Electronic and Computer Engineering, University of Pretoria. His research interests include mathematical modeling of multivariate statistics, classification methods, knowledge discovery, reasoning with uncertainty and inference, predictive analytics and inference with applications in wireless communications, finance, health, and robotics.

BODHASWAR T. MAHARAJ (Senior Member, IEEE) received the Ph.D. degree in engineering in the area of wireless communications from the University of Pretoria. He is a Full Professor and currently holds the research position of Sentech Chair of the Broadband Wireless Multimedia Communications (BWMC), Department of Electrical, Electronic and Computer Engineering, University of Pretoria. His research interests include OFDM-MIMO systems, massive MIMO, cognitive radio resource allocation, and 5G cognitive radio sensor networks.