POZNAN UNIVERSITY OF TECHNOLOGY

FACULTY OF COMPUTING AND TELECOMMUNICATION
Institute of Radiocommunications

Doctor of Philosophy Dissertation

NEW MACHINE LEARNING METHODS FOR SPECTRUM SENSING IN WIRELESS COMMUNICATION SYSTEMS

mgr inż. Małgorzata Wasilewska

Supervisor
prof. dr hab. inż. Hanna Bogucka

POZNAŃ 2024

Abstract

Motivated by the increase in mobile communication traffic, the requirement of high data rates, and the associated scarcity of spectrum, the author of this thesis presents her research that led to new approaches to Spectrum Sensing (SS) based on Machine Learning (ML) methods. Contrary to the traditional approach, the author provides solutions to ML-based autonomous SS that takes the radio-channel variations into account, cooperative (Federated Learning (FL)-based) SS with improved performance, and secure FL-based SS robust against data poisoning.

The thesis of this dissertation is the following: *There exist new methods for spectrum sensing in wireless communication systems that are based on machine learning and that are more reliable than the existing ones.* The author proves this by presenting the following original research and contributions.

After the analysis of the state of the art in the field of reliable SS in radio communication channels and the identification of knowledge gaps, new effective, low-cost algorithms for the autonomous ML-supported SS have been proposed. Fifth Generation (5G) system scenario has been considered, in which Resource Blocks (RBs) are subject to sensing for possible secondary use. The proposed algorithms take advantage of the measured Primary User (PU)'s signal energy and the time- and frequency dependencies of the sensed signal RBs occupancy. Next, new effective Deep Learning (DL)-based algorithms have been developed for SS and spectrum occupancy prediction taking into account variations of the wireless channel. Following considerations on the autonomous SS, cooperative FL-based methods have been investigated. A new efficient FL-based SS algorithm has been designed with high performance and allowing incoming users to take advantage of the global FL model without sacrificing their computing resources for model training. Finally, the impact of coordinated and random poisoning attacks on FL-based SS has been evaluated, and a new efficient algorithm has been designed to detect and mitigate such attacks.

The main conclusion of this dissertation is that the original solutions of the author can significantly improve the performance, reliability, and security of SS in the scenarios considered for 5G radio access networks.

Streszczenie

Motywowana wzrostem ruchu w komunikacji mobilnej, wymogiem dużych przepływności danych i związanym z tym niedoborem widma, autorka niniejszej rozprawy doktorskiej przedstawia swoje badania nad nowym podejściem do detekcji widma, tzw. *sensingu* (ang. Spectrum Sensing (SS)) w oparciu o metody uczenia maszynowego (ang. Machine Learning (ML)) . W przeciwieństwie do tradycyjnego podejścia, autorka proponuje metody autonomicznego SS opartego na ML, które uwzględniają zaniki w kanale radiowym, kooperatywnego SS (opartego na uczeniu federacyjnym (ang. Federated Learning (FL)) o zwiększonej niezawodności i algorytm bezpiecznego SS opartego na FL, odpornego na zatruwanie danych.

Teza tej rozprawy jest następująca: *Istnieją nowe metody wykrywania widma w systemach komunikacji bezprzewodowej, oparte na uczeniu maszynowym i jednocześnie bardziej niezawodne niż istniejące.* Autorka udowadnia to przedstawiając poniższe oryginalne badania.

Po przeanalizowaniu aktualnego stanu wiedzy w dziedzinie niezawodnego SS w kanałach komunikacji radiowej i identyfikacji otwartych problemów, opracowano nowe, skuteczne algorytmy dla autonomicznego SS wspieranego przez ML. Rozważono scenariusz systemu 5G, w którym bloki zasobów (ang. Resource Blocks (RBs)) podlegają detekcji w celu ewentualnego wtórnego wykorzystania. Zaproponowane algorytmy wykorzystują zmierzoną energię sygnału użytkownika licencjonowanego (ang. Primary User (PU)) oraz zależności czasowe i częstotliwościowe zajętości RBs. Następnie opracowano nowe, skuteczne algorytmy oparte na głębokim uczeniu maszynowym (ang. DL) do detekcji i przewidywania zajętości RBs z uwzględnieniem zmienności i zaników w kanale bezprzewodowym. W następstwie rozważań dotyczących autonomicznych SS, zbadano kooperacyjne metody SS. Zaprojektowano nowy, algorytm SS oparty na FL, zapewniający wysoką jakość detekcji i umożliwiający nowym użytkownikom pojawiajacym się w sieci możliwość korzystania z globalnego modelu FL bez konieczności poświęcania zasobów obliczeniowych w celu uczenia modeli. Na koniec oszacowano wpływ skoordynowanych i losowych ataków typu zatruwającego na SS oparty na FL i opracowano nowy, efektywny algorytm do wykrywania i łagodzenia takich ataków.

Głównym wnioskiem z tej rozprawy jest to, że oryginalne rozwiązania autorki mogą znacząco poprawić efektywność, niezawodność i bezpieczeństwo SS w scenariuszach rozpatrywanych dla radiowych sieci dostępowych 5G.

Table of contents

List of Figures

List of Tables

List of Acronyms

3GPP	Third Generation Partnership Project
5G	Fifth Generation
AI	Artificial Intelligence
AML	Adversarial Machine Learning
API	Application Programming Interface
ARFCN	Absolute Radio Frequency Channel Number
AWGN	Additive White Gaussian Noise
BPNN	Back Propagation Neural Network
BPSK	Binary Phase Shift Keying
BS	Base Station
CA	Context-Awareness
CNN	Convolutional Neural Network
CQI	Channel Quality Indicator
CRAT	Context-Aware Radio Access Technology
CR	Cognitive Radio
CRN	Cognitive Radio Network
CSI	Channel State Information
CSS	Cooperative Spectrum Sensing
DL	Deep Learning
DoS	Denial of Service
DP	Differential Privacy
DT	Decision Tree
ED	Energy Detection
EPA	Extended Pedestrian A Model
EV	Energy Value
EVA	Extended Vehicular A Model
FC	Fusion Center
FDD	Frequency Division Duplex
IFFT	Inverse Fast Fourier Transform
FL	Federated Learning

GAN Generative Adversarial Network
GMM Gaussian Mixture Model
GSM Global System for Mobile Communications
ICIC Inter-Cell Interference Coordination
IoT Internet of Things
IQ In-Phase and Quadrature
kNN k-Nearest Neighbors
LR Logistic Regression
LSTM Long Short-Term Memory
LTE Long Term Evolution
MBR Maximum Bit Rate
MCS Modulation and Coding Scheme
MITM Man-in-the-Middle
ML Machine Learning
NB Naive Bayes
NN Neural Network
OFDM Orthogonal Frequency-Division Multiplexing
PA Primitive Algorithm
PCA Principal Component Analysis
PU Primary User
PUE Primary User Emulation
QCI Quality of Service Class Identifier
QoE Quality of Experience
QoS Quality of Service
RAT Radio Access Technologies
RB Resource Block
RF Random Forest
RGB Red, Green, Blue
RI Rank Indicator
RL Reinforcement Learning
RNN Recurrent Neural Network
ROC Receiver Operating Characteristic
RSS Received Signal Strength
SDR Software-Defined Radio
SNR Signal-to-Noise Ratio
SOM Self-Organizing Map
SP Spectrum Prediction

SS Spectrum Sensing
SSDF Spectrum Sensing Data Falsification
SU Secondary User
SVM Support Vector Machine
SVR Suppor Vector Regression
UE User Equipment
UL uplink

Introduction

Motivation

In today's wireless communication systems, the radio spectrum has become a scarce resource. According to [1], the global mobile network data traffic reached 151 EB per month (including Fixed Wireless Access (FWA) services) in the second quarter of 2024. It will grow to approximately 466 Exabytes per month in 2029, and 5G networks will carry 75 percent of that traffic. There will be 9.3 billion mobile subscriptions (including 5.6 billion of 5G ones) and 38.8 billion connected devices constituting the Internet of Things (IoT), out of which 6.7 billion will be connected via cellular networks. These facts and predictions imply challenges in utilizing radio spectrum resources and planning their allocation to wireless connections. Effective, intelligent, and possibly cognitive methods of sensing available spectral resources, as well as algorithms for accessing them dynamically and responsively, are needed.

The idea of Cognitive Radio (CR) was proposed two decades ago to enhance the operation of radio devices and networks by embedding operational environment awareness and artificial intelligence in them. The aim was to increase the use of radio resources in consideration of the constantly growing number of wireless transmissions. A CR user is called a Secondary User (SU), while a licensed-system (in a given frequency band) user is called a Primary User (PU). SUs attempt to gain radio environment awareness to opportunistically access the radio resources, temporarily not used by PUs. Radio-environment awareness, in particular, the awareness of spectrum occupancy and transmission conditions, enables the optimization of the SUs' transmission and protection of the PUs' transmission from the interference possibly generated by SUs. This allows for the maximization of spectrum usage while keeping the interference level observed by PUs in the acceptable range. To achieve radio-environment awareness, intelligent spectrum sensisng and management methods are considered to reuse frequency bands at a certain time and location (so-called *spectrum gaps* or in a more generic sense *white spaces*) when they are not used by licensed users [198].

The Spectrum Sensing (SS) methods are supposed to determine whether PUs are transmitting or not, and hence, enable the SU's transmission. SS commonly refers to the multiplicity of methods of obtaining the spectrum-occupancy awareness of SUs. Based on this awareness, CR should make intelligent decisions on transmission and reception-related actions, and constantly improve these decisions by learning from experience.

One of the challenges is that traditional SS methods, are unable to take full advantage of the time, frequency, or spatial dependencies that exist in detected signals, which results in a rather limited performance of these methods. The AI-based and ML algorithms that can find intricate features in the input data and recognize

present signals are being considered to improve the performance of traditional SS.

The ML-based methods pose another challenge, which is to determine how to train the ML models so that they are useful for correct spectrum detection and prediction to enable its reusability. The training process requires sufficient amounts of data. Additionally, the supervised type of ML algorithms is better suited for a problem of SS than unsupervised ML, which requires correct *a priori* labeling of the spectrum occupancy data. Moreover, the ML model used for spectrum occupancy decisions must be trained to reflect also the changing radio environment. The ML algorithm needs to be able to adapt to the changes occurring in the wireless channel. The answer to the aforementioned challenges can be to apply a cooperative and adaptable ML-based SS algorithm in which multiple SUs cooperate by collecting smaller amounts of data that are characterized and influenced by their local radio environment, and by creating a common SS ML model that would work well in different wireless scenarios. Such an approach is called Federated Learning (FL).

Finally, the algorithm security challenges appear while considering cooperative and intelligent SS solutions. The process of cooperative (including FL-based) SS can get hijacked by malicious devices in order to disturb the proper spectrum-sharing process. Attackers may falsify the presence of the spectrum gaps (opportunities) in order to use detected resources for their own transmission. Alternatively, the attackers may aim at falsification of the occupied spectrum in order to encourage SUs to use frequencies occupied by PUs and cause interference to the PUs transmission.

Motivated by the increased mobile communication traffic, required high data rates, and associated spectrum scarcity, the author of this thesis presents her research that led to new approaches to SS based on ML methods. Contrary to the traditional approach the author provides solutions to ML-based autonomous SS that takes the radio-channel variations into account, cooperative (FL-based) SS with improved performance, and secure FL-based SS robust against data poisoning.

Dissertation thesis and main goals

The thesis of this dissertation is the following:

There exist new methods for spectrum sensing in wireless communication systems that are based on machine learning and that are more reliable than the existing ones.

The main goal of the thesis is to propose such methods and in particular:

- To analyze the state of the art in the field of reliable SS in radio communication channels, and project the knowledge gaps against original solutions presented in Chapters 2–5; (This goal is addressed in Chapter 1.)

- To develop the effective, low-cost algorithm for the autonomous ML-supported SS, that takes advantage of the measured PU's signal energy and the time- and frequency dependencies of the sensed signal spectrum; (This goal is addressed in Chapter 2.)

- To develop the effective, DL algorithms for SS and spectrum occupancy prediction that take the wireless channel variations into account; (This goal is addressed in Chapter 3.)

- To develop an efficient FL-based SS algorithm, suitable for varying radio propagation conditions; (This goal is addressed in Chapter 4.)
- To evaluate the impact of coordinated and random poisoning attacks on FL-based SS, and to design an efficient algorithm to detect and mitigate such attacks. (This goal is addressed in Chapter 5.)

Dissertation Outline

In Chapter 1, the author of this thesis presents an overview of the state of the art of the ML-based SS. Firstly, the Context-Awareness (CA) notion and meaning of CA are discussed. Then, ML methods suitable for acquiring CA and particularly for SS are discussed. Next, considerations are provided for spectrum pattern recognition and prediction methods. The survey of literature categorizes published works under multiple aspects. This categorization is presented in the form of tables in Appendix A. Finally, in this chapter, the CA design issues, trade-offs, challenges, and recommendations are discussed.

Chapters 2 through 5 present the original contributions of the author of this thesis. Each of them examines a problem of ML-based spectrum sensing and each of them focuses on different aspects of this problem.

In Chapter 2, the author of this thesis focuses on autonomous SS. Some typical SS methods are described, particularly the ED method, which is used as a base for further analysis and experiments. The enhancement of ED is proposed that consists in the application of a supervised ML algorithm, after the energy measurement and possible detection, that takes advantage of the time and frequency dependencies contained in the received PU signal spectrum. Moreover, the measured Energy Values (EVs) instead of the hard ED decisions are proposed as input for the ML step of the SS method. The proposed supervised ML algorithms are kNN and Random Forest (RF). When based on EVs they turn out to perform better than the standard ED and other considered supervised ML algorithms applied.

In Chapter 3, the author still considers the autonomous SS, however, DL algorithms are analyzed. By examining the use of DL to enhance SS, Spectrum Prediction (SP), and fading level estimate based on the calculation of the energy value, this chapter builds on the analysis of classical ML methods from the previous chapter. First, the author designed several DL-based SS and SP techniques, incorporating the Neural Network (NN), the Recurrent Neural Network (RNN), and the Convolutional Neural Network (CNN). Their use and effectiveness in SS and SP are compared, and the benefits and drawbacks are analyzed. A straightforward baseline approach (called Primitive Algorithm (PA)) is suggested for comparing the outcomes of SP. Out of all analyzed DL methods, CNN-based SS and SP shows the best performance in terms of the probability of correct detection and prediction of the spectrum occupancy and the probability of false alarm. In the second part of this chapter, the focus is on one chosen DL method, namely CNN applied for both SS and SP. The addition introduced is a fading level estimation at the SU receiver, also based on the DL algorithm. The threshold of fading level is introduced to reject decisions biased by deep fading and protect the PU transmission. This threshold is optimized for the best performance of CNN-based SS and SP.

Chapter 4 is devoted to cooperative SS involving multiple sensors capable of ML-based decision-making. The author proposes a new, iterative, and adaptive process

of federated modeling, namely FL for SS. She examines how well this method performs under different wireless channel conditions. Sensors participating in this process collect their local data, create local CNN models, and then contribute to the building of corporate (federated) models. One of the benefits of this method is that only the weights of the local models are transmitted to the central FL server, and consequently a lower volume of information traffic is required. Moreover, data privacy is guaranteed *by design*. The possibility of applying this FL-based SS approach in changing radio environment conditions is also examined. As presented, the proposed FL method has shown good performance (in terms of high probability of detection and low probability of false alarms) and adaptability to these conditions. Despite the fact that FL sensors (nodes) models are suited to individual mean SNR values, the corporate models they generate are well applicable in the case of multiple SNR values from a given range. Another important advantage of the proposed method is that it enables new incoming SUs to perform intelligent sensing without the need for extensive data collection and CNN model training. Instead, these SUs can take advantage of earlier created FL models.

Chapter 5 expands the FL-based SS methods with additional security considerations. The FL algorithms, although known for protecting the private data of participating users, are still prone to some types of attacks that the author will present in this chapter. The main emphasis is placed on the topic of poisoning attacks, mainly label flipping attacks that can interfere with the local training process, which in turn damages the global ML model created by FL algorithm. In this chapter, the author considers a new poisoning attack algorithm that specifically targets FL-based SS, by applying knowledge on sensed signal statistics. In addition, the author proposes a novel method of detecting these types of attacks that is based only on testing the similarities of the local models.

Finally, this dissertation is concluded in Chapter 6, in which the main findings of the author regarding the proposed ML-based spectrum sensing methods are summarized.

Author's published contributions

The main contributions of this dissertation summarized above have been published in several articles listed below.

Papers in international journals:

1. Małgorzata Wasilewska, and Hanna Bogucka, "Protection Against Poisoning Attacks on Federated Learning-based Spectrum Sensing," accepted to *IEEE Journal on Selected Areas in Communications* Special Issue on Zero Trust for Next-Generation Networking (planned publication: second quarter of 2025).

2. Małgorzata Wasilewska, Hanna Bogucka, and H. Vincent Poor. "Secure federated learning for cognitive radio sensing." *IEEE Communications Magazine* 61.3 (2023), pp. 68-73.

3. Salim Janji, Adam Samorzewski, Małgorzata Wasilewska, and Adrian Kliks. "On the placement and sustainability of drone FSO backhaul relays." *IEEE Wireless Communications Letters* 11.8 (2022), pp. 1723-1727.

4. Małgorzata Wasilewska, Adrian Kliks, Hanna Bogucka, Krzysztof Cichoń, Julius Ruseckas, Gediminas Molis, Aušra Mackutė-Varoneckienė, Tomas Krilavičius"Artificial Intelligence for Radio Communication Context-Awareness." *IEEE Access* 9 (2021): 144820-144856.

5. Małgorzata Wasilewska, and Łukasz Kułacz. "Machine Learning-Based Small Cell Location Selection Process." *Journal of Telecommunications and Information Technology* No. 2 (2021), pp. 120-126.

6. Małgorzata Wasilewska, and Hanna Bogucka. "Space-time-frequency machine learning for improved 4G/5G energy detection." *International Journal of Electronics and Telecommunications* (2020) Vol. 66, No. 1, pp. 217–223.

7. Małgorzata Wasilewska, Hanna Bogucka, and Adrian Kliks. "Federated learning for 5G radio spectrum sensing." *Sensors* 22.1 (2021): 198.

8. Małgorzata Wasilewska, and Hanna Bogucka. "Machine learning for LTE energy detection performance improvement." *Sensors* 19.19 (2019): 4348.

Book chapters:

1. Małgorzata Wasilewska, Hanna Bogucka, and Adrian Kliks, "Spectrum Sensing and Prediction for 5G Radio" [In:] Z. Deze, et.al. (eds), Big Data Technologies and Applications. WiCON 2020, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 371, pp. 176-194, Springer, Cham, ISBN: 978-3-030-72801-4,

Papers published in the proceedings of the international conferences:

1. Małgorzata Wasilewska, and Hanna Bogucka. "Deep learning for Improved Spectrum Occupancy Prediction with Fading Estimation in 5G Radio." IEEE International Conference on Communications (ICC 2023). 28 May – 01 June 2023, Rome, Italy.

Papers published in the national journals and in the proceedings of national conferences:

1. Małgorzata Wasilewska, and Hanna Bogucka. "Metody uczenia maszynowego dla poprawy jakości detekcji sygnału LTE." *Przegląd Telekomunikacyjny+ Wiadomości Telekomunikacyjne*, nr 6/2019, str. 274-278 (2019).

2. Małgorzata Wasilewska, and Hanna Bogucka. "Analiza wpływu ataków zatruwających na detekcję zasobów radiowych bazujących na uczeniu federacyjnym." *Przegląd Telekomunikacyjny+ Wiadomości Telekomunikacyjne*, nr 4/2023, str. 198-203 (2023).

3. Małgorzata Wasilewska, and Hanna Bogucka. "Inteligentna predykcja stanu zajętości widma z uwzględnieniem wpływu zaników." *Przegląd Telekomunikacyjny+ Wiadomości Telekomunikacyjne*, nr 4/2022, str. 481-485 (2022).

4. Małgorzata Wasilewska, and Hanna Bogucka. "Predykacja zajętości kanału LTE za pomocą rekurencyjnej sieci neuronowej." *Przegląd Telekomunikacyjny+ Wiadomości Telekomunikacyjne*, Nr 7-8/2020, str. 269-273 (2020).

Chapter 1

State of the Art

In the era of ubiquitous information access and pervasive communication networks, systems and nodes must be aware of their context of operation, utilizing information on ambient networks, links, devices, and applications. This context awareness will improve the efficiency of existing services and the provision of personalized services. For example, networks will need to be more aware of the application requirements, Quality of Experience (QoE) and Quality of Service (QoS) metrics, local (or more global) conditions of operation, and apply specific ways to adapt the application flows to meet users' needs under specific environmental conditions. The context-based adaptations of various transmission and network parameters must consider the device-level, user-level, link-level, network- and application-level context. The context information consists of different parts and components, each affecting the decision-making process's steps differently. More specifically, various parts that constitute the context are related to the following levels:

1. hardware platforms, which pose specific hardware constraints and implementation issues,

2. radio environment conditions in terms of location-specific parameters, wireless channel quality, spectrum availability, other-users characteristics and signal features, traffic patterns, interference levels,

3. required performance QoS parameters that can be identified in all layers of the system protocol stack and are considered to be the basis for the evaluation of decisions made,

4. network management policies as a set of rules used to control, among others, the behavior of nodes, manage available resources, regulate interference to other deployed systems, and obtain identified trade-offs.

Note that in communication system management, (1) and (4) are constraints, (2) are changing radio-environment parameters and characteristics, and (3) are the goals of optimization. In this thesis, the author concentrates specifically on the radio context, i.e., on (2).

Apart from and beyond the gathering of context-information-building data, radio-environment awareness can be enriched with reasonable overhead by using suitable ML methods and AI embedded in communication networks, either at the network edge or in a central entity, or a dedicated subsystem. In this chapter, such

ML/AI methods are surveyed, and their suitability for particular radio-environment contexts and applications is discussed.

This chapter is organized as follows. In Section 1.1, CA is precisely defined, and its significance and main limitations are explained. The context information life cycle and various domains for context information gathering are also explained. In Section 1.2, ML methods suitable for acquiring CA are shortly discussed. The state of the art in ML algorithms applied for SS, both autonomous and cooperative SS is presented in Section 1.3. This is followed by the considerations on the spectrum pattern recognition and prediction methods in Section 1.4 that are based on specific time-, frequency-, and space dependencies in the spectrum occupation. Finally, in Section 1.5 discusses the CA design issues, trade-offs, challenges and recommendations, while Section 1.6 summarizes this chapter.

1.1 The Significance of Context Awareness in Wireless Communications

CA, being a basis for pervasive computing for more than thirty years [181, 126], has recently been considered an up-and-coming yet challenging concept in the domain of wireless communications [117, 96, 207]. Although the algorithms applied in contemporary wireless systems try to adjust the transmission profile to varying channel conditions (such as adaptive bit and power loading [182, 55, 97, 17], advanced modulation schemes [192, 25], already standardized Modulation and Coding Scheme (MCS) [10, 81], close- or open-loop power control [75, 136]), the application of AI/ML toolboxes fosters the implementation of situation-aware communication systems. The possibility of collecting vast and rich information about the surrounding (radio) environment delineates new communications paradigms, where user-centered transmission is adjusted to the current communication context. Practically, the data may be collected from dedicated sensors or directly from end-user terminals (as it is done currently in, e.g., minimization of drive test schemes in cellular networks [131]), applying the concept of crowd-sensing or crowd-sourcing [86, 95]. Access to the gathered data allows us to construct the context of the considered scenario, which can be further used to adjust the communication link and network profile. However, as the amount of gathered data increases (due to a high number of data sources, increased frequency of data collection, or increased amount of observed data per single read), context-aware communications have to deal with problems typical for *big data processing* [186, 35, 22, 28]. Clearly, the lack of efficient big-data processing makes the collected information useless. *Raw data*, generated directly by sensors, must be processed, managed, checked for consistency, and complemented to create the complete context information.

1.1.1 Radio Communication Awareness

Various definitions of the term *context* can be found in the literature. The Merriam-Webster's Dictionary provides a general definition of it - a context refers to "*the interrelated conditions in which something exists or occurs: environment, settings*" [2]. Similarly, the historically significant definitions related to computing and communications, such as in [145, 24], define the term *context* in a specific situation or refer to some certain use case. In particular, the authors of [145] describe

context as specific locations, identities of close objects and creatures, and changes to them. However, in a broader sense, context can be specified operationally as proposed in [44], and for the sake of clarity, the quote of this definition is cited verbatim below:

"Context is any information that can be used to characterize the situation of entities (i.e., whether a person, place, or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically the location, identity, and state of people, groups, and computational and physical objects" [44].

Finally, in [6], context is presented as a set of interrelated events between which logical and timing relations can be identified. The events are classified into discrete (such as starting a call) and continuous (executing the call). Assuming a set of interrelated events specifying a given context, the logical relations are defined as the Boolean formula of the appearance of these individual events. For example, context is said to be a unit context when all constituent events must be understood as *true*.

Given the above discussion, it is worth analyzing the meaning of *context awareness* for completeness. One may say that a system or algorithm will be aware of an existing context if it uses the context (and all related data) to provide detailed information to the end-user. In this sense, a context-aware wireless system will support various features and possess specific attributes, such as the ability to observe the surrounding environment, sense it, perform data acquisition and processing, and finally react.

The terms "context" and CA should be adjusted accordingly regarding radio communications. In particular, radio communication context will be a set of information and data that characterize the communication-related situation of the network or the entire wireless system. Thus, radio communication context will be constituted by such descriptors as geographical location, identity and state of wireless nodes (persons or things, base stations, transmission points, etc.), status of wireless channels, transmission requirements, and system performance. Again, the radio context may also be presented as a set of interrelated events describing the functioning of the wireless transceivers and the whole network. Thus, radio communication context awareness will be the ability (of a device, a network, or a system of networks) to observe the surrounding radio and geographical environment, sense it, perform data acquisition and processing, and react accordingly. It is evident that the ability to be radio-communication context-aware is an inherent feature of the cognitive radio and cognitive networks [116]. Access to rich context information about the surrounding radio environment leverages the application of more tailored communication schemes. Contemporary wireless communication systems utilize such data to some extent. One may think of adapting transmit power (through so-called open or close control loops) or selecting the best modulation-and-coding scheme depending on the instantaneous channel conditions. However, more advanced strategies have been considered in the context of cognitive radio and cross-layer cooperation, i.e., where advanced information exchange across the protocol stack has been proposed [83, 150, 141].

It is possible to define various kinds of radio context information. First, following [126], it may be divided into two fundamental classes - primary and secondary. As the former is defined as the set of information retrieved without any data fusion operation (one may think of directly accessible information such as received signal

strength), the latter refers to any information that can be derived based on the primary context data (such as the location of a user computed based on any triangulation scheme). Furthermore, various smaller classes of radio context information can be specified, i.e., location, time, identity, and activity information. Regardless of the exact classification of context data, it is worth summarizing the examples of radio context information available in contemporary wireless systems (mainly cellular networks, but also wireless local and personal area networks). They are typically used for describing the observed or predicted signal quality, assessing the measured signal power, defining the best method of adaptive signal processing, or just describing the generic system setup. These are presented in Tab. 1.1. One must notice that this table is incomplete, and many other parameters can be specified. However, considering them jointly, and particularly with association to a specific location and time, detailed context awareness can be achieved by contemporary systems. At the same time, further exploration and processing of various types of information may increase overall CA and, consequently, lead to better exploitation of available resources at the expense of data processing (hardware resources and related energy consumption). The author believes that AI/ML are excellent tools for exploiting acquired data, extracting useful information contained in these data, and enriching the context awareness of a considered system.

TABLE 1.1: Examples of radio context information utilized in various contemporary wireless networks

Name	**Description**	**Class**	**Similar Descriptors**
ARP - Allocation and Retention Priority	Specifies relative importance compared to other bearers for the allocation and retention of a new bearer	Secondary	QCI (QoS Class Identifier), GBR (Guaranteed Bit Rate), MBR (Maximum Bit Rate), AMBR (Aggregate Maximum Bit Rate), AC (Access Categories)
ARFCN - Absolute radio-frequency channel number	Allows for the identification of the communication channel number and center frequency	Secondary	5G NR ARFCN (variant for New Radio), EARFCN (E-UTRA Absolute Radio Frequency Channel Number), UARFCN (UTRA Absolute Radio Frequency Channel Number),
CQI - Channel Quality Indicator	Indicates the quality of the wireless channel	Secondary	CSI (Channel State Information)
CSO (Cell Selection Offset)	Defined in various ways, used for cell range extension in heterogeneous networks	Primary	cellIndividualOffset, q-offsetCell
MCS - Modulation and Coding Scheme	Specifies used modulation and coding configuration	Secondary	MCS (Modulation and Coding Set - in IEEE 802.11 networks)
RI - Rank Indicator	Indicates the number of layers and signal streams transmitted in the downlink (LTE)	Secondary	PMI (Precoding Matrix Index), CQI (Channel Quality Indicator)
RSS - Receives Signal Strength	Measured power of the received signal	Primary	RSSI (Received Signal Strength Indicator), RSRP (Reference Signal Receive Power), RSRQ (Reference Signal Received Quality), RCPI (Received Channel Power Indicator), Signal Strength, SS-RSRP(Synchronization Signal RSRP), OSTP (OFDM Symbol Transmit Power), RX (Received Power, e.g. in Bluetooth) etc.
SNR - Signal to Nosie Ratio	Describes the ratio between the power of the wanted signal within certain band and the noise power observed in this band	Primary	SINR (Singal-to-Interference-plus-Noise Ratio)
QCI - QoS Class Identifier	Defines the quality of packet communication provided by cellular networks (LTE)	Secondary	5QI (5G QoS Identifier)
Generic physical parameters defining system	Fundamental information such as center frequency, bandwidth, start and stop frequency, maximum allowed power	Any	Various modifications of these parameters

1.1.2 Context Information Life Cycle

It is essential to note that context awareness may have various time scales. Some data will be valid for a short period, whereas others will represent long-term trends. This phenomenon is often referred to as information aging [84], and recently, the age-of-information metric has been considered a tool for measuring the applicability of specific data while stating their validity. However, the age-of-information entails updating such data, creating a so-called context life cycle [126]. In computer science, two main terms are typically discussed: data life cycle management and information life cycle management. In the case of radio communication awareness, such a life cycle may define how radio data passes from one stage to another. In contrast, a stage represents the validity of the data and its aging. For example, four life cycle stages can be identified: radio context acquisition - modeling - reasoning - dissemination, as presented in, e.g., [126]. Another approach is to identify these phases as data collection - classification - processing and storage - sharing and dissemination, as presented in Fig. 1.1. However, much more advanced schemes are possible. For example, [61] proposed a more advanced scheme, where such phases as data collection, classification, handling and storage, release, and backup are identified, as graphically presented in Fig. 1.1. In any way, there is a need to update the collected information (permanently, periodically, or on request). As the author discusses further in this chapter, the data collection and processing process for increasing local and global context awareness in an advanced wireless communication system should be steered by appropriate AI/ML tools.

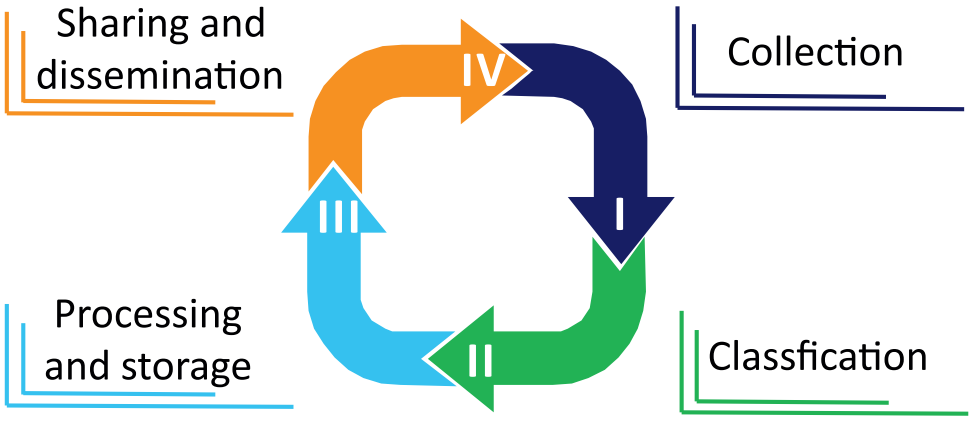


FIGURE 1.1: Generic cycle for radio context information management

1.1.3 Various Domains for Context Information Gathering

Table 1.1 gathers commonly used metrics or parameters for defining the instantaneous radio communication context. They may be broadly classified as information related to power management (such as Received Signal Strength (RSS), SNR), channel quality measurements (such as Channel Quality Indicator (CQI), Channel State Information (CSI)), network configuration (such as Absolute Radio Frequency Channel Number (ARFCN)), selected signal processing schemes (such as Rank Indicator (RI) or MCS) or traffic characterization (e.g., Maximum Bit Rate (MBR), Quality of Service Class Identifier (QCI)). However, as mentioned above, the list is

incomplete, and the number of parameters used in contemporary wireless networks is continuously growing. From the point of view of creating rich radio context information, it is worth investigating new domains of network functioning, which can be used for context information gathering and enrichment.

First, following the findings made in the cognitive radio domain, information about other transmissions can be gained through single-node or cooperative SS. Although the ultimate aim of traditional SS is to detect the presence or absence of primary users (i.e., the users of wireless systems licensed to occupy a specific frequency band at a particular location and time), this scheme can be extended to the detection of any existing transmission in the vicinity or even more - to detect specific features of these transmissions (such for example the type of signal, applied modulation scheme). The SS process may be realized by each device independently or cooperatively between communication-network nodes, and it may also be assumed that dedicated sensing nodes (deployed and devoted only for this purpose) are applied to improve data collection. Consequently, rich radio-context information may be inferred using various techniques for prospective big-data processing. SS is then one of the essential new domains of radio context exploration that could be inherently integrated into future wireless networks. Sec. 1.3 contains an analysis of critical findings in applying AI/ML tools for data gathering through SS.

Besides detecting the presence or absence of other ongoing transmissions, there is a possibility of enriching context information by recognizing various signal features. These signal features could include modulation recognition (e.g., if it is a single carrier or multi-carrier) or identification of types of signals (if it is, e.g., a signal of a 3G, 4G, or 5G network or another type of a distinctive local or wider area network). Exploiting such context information may serve multiple communication tasks, e.g., being able to transmit using OFDM (orthogonal-frequency division multiplexing) subcarriers orthogonal to the detected ones or spread-spectrum techniques not affecting the detected narrow-band signal with a detected modulation type.

Radio context information is highly dependent on (geographical) localization. Thus, the incorporation of user localization techniques is of paramount importance in wireless communication systems. As in the previous cases of discussed context information domains, AI/ML tools can also be applied here to improve user localization.

Finally, utilizing available radio resources (among others, time-frequency chunks and allowable power) may be improved when the transmission patterns of other existing transmissions are known. These can be used to specify the types of other parallel signals and, consequently, better adjust planned transmission to such a radio communication context. The above-identified domains are graphically presented in Fig. 1.2.

One should observe that the increased amount of collected context information from new domains increases the overall processing complexity of system management (see Fig. 1.3). On the one hand, the richer the radio context information, the better the adjustments to the system setup; on the other hand, there is a significant increase in the complexity of such a system. Thus, there is a need, first, for accurate acquisition of the context information and second, for advanced (big) data processing. In both cases, applying AI/ML tools may provide reliable solutions.

Furthermore, as the network density increases by introducing mmWave picocells and femtocells, the problem of determining which Radio Access Technologies (RAT)

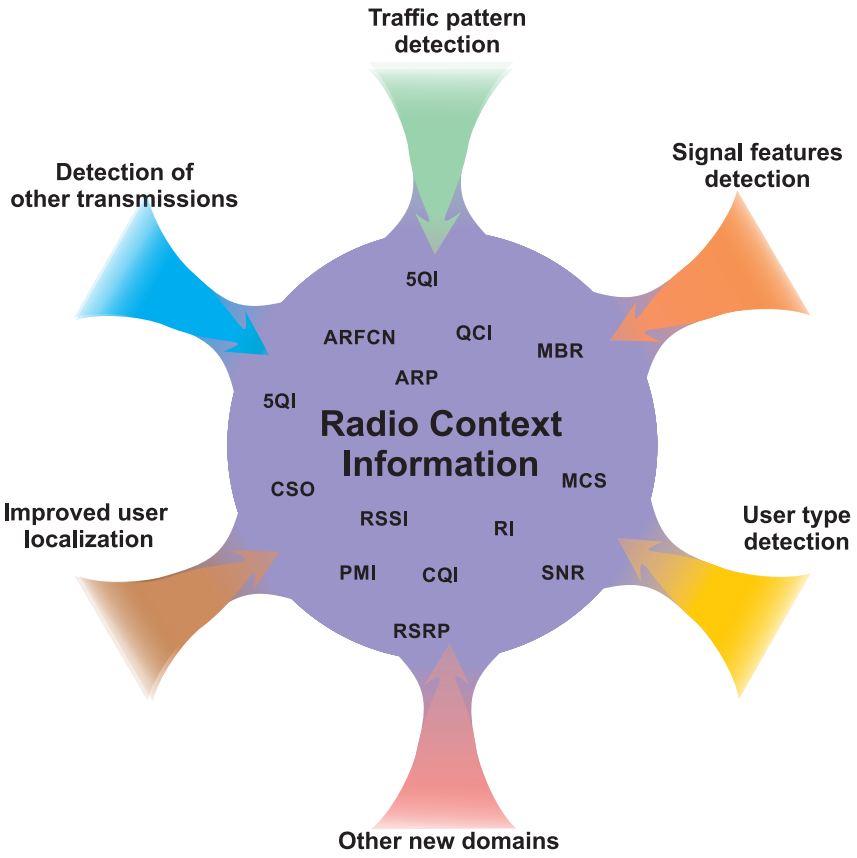


FIGURE 1.2: Enrichment of radio context information by adding new context information domains (Figure source: [180])

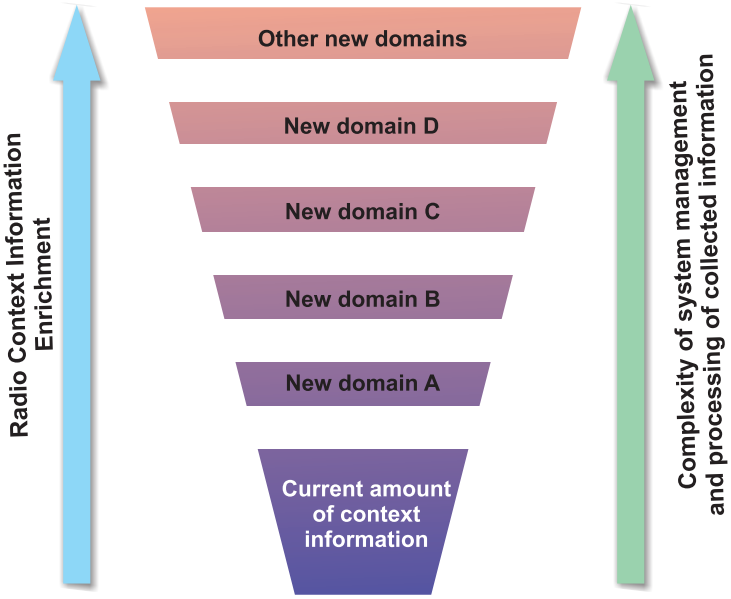


FIGURE 1.3: Improved radio context information leads to increased complexity of system management (Figure source: [180])

a user should use for cell discovery at a given time becomes more complex. New methods are proposed, such as based on the Context-Aware Radio Access Technology (CRAT) [58]. A mathematical model of CRAT, considering the user and network context, is derived, adopting an analytical hierarchical process for weighting the importance of the selection criteria and Technique for Order of Preference by Similarity to Ideal Solution [72] for ranking the available RATs. The simulation shows that this approach outperforms the conventional A2A4-RSRQ approach, used in Long Term Evolution (LTE), regarding the number of handovers, average network delay, throughput, and packet delivery ratio by 20-100 %.

1.2 Machine Learning Methods for Context Awareness

As mentioned in the previous section, ML methods are crucial in context information gathering and utilization. Below, the author outlines the basics of these methods and their application to build and enrich context awareness in radio communication systems. The possible applications of different methods for context awareness are displayed in Table 1.2.

Machine learning methods can be broadly organized into supervised, unsupervised, and reinforcement. Figure 1.4 shows a schematic grouping of various machine learning methods.

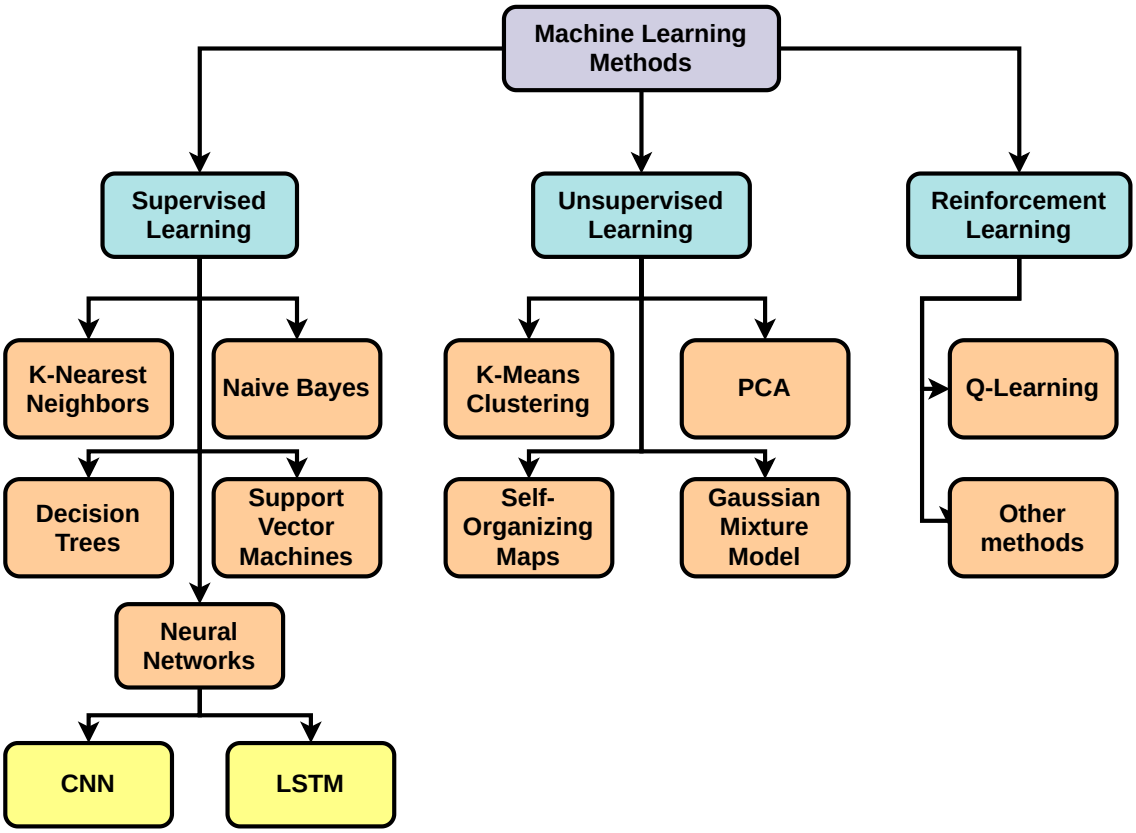


FIGURE 1.4: Most popular machine learning methods used in communication networks.

TABLE 1.2: Application of ML methods for Context Awareness

Category	Method	Examples of applications
Supervised learning	K-nearest neighbors	Single node spectrum sensing [177], cooperative spectrum sensing [115, 161, 147], modulation recognition using feature set based on higher-order statistics [3], indoor user localization [7]
	Naive Bayes	Single node spectrum sensing [165]
	Decision Trees	Modulation recognition [185, 156]
	Support vector machines	Single node spectrum sensing [74], cooperative spectrum sensing [103, 71], cooperative decision scheme for spectrum sensing in vehicular communication environment [29], user classification [157]
	Neural Networks	Single node spectrum sensing [162], modulation recognition [36], multi-carrier modulation recognition [92]
	Neural Networks: CNN	Single node spectrum sensing [123], combining sensing results in cooperative spectrum sensing [88], modulation recognition [206], traffic pattern recognition [26]
	Neural Networks: LSTM	Prediction of channel state [195], modulation recognition [202], prediction of PU's next spectrum state [197]
Unsupervised learning	K-means clustering	Single node spectrum sensing [199], cooperative spectrum sensing [103]
	PCA	Modulation recognition [132], multi-carrier modulation recognition [46]
Reinforcement learning	Q-learning and others	Learning spectrum sensing policy [21], cooperative sensing [101]

1.2.1 Supervised Learning

In supervised learning, a predictive model is constructed using the training data of inputs and corresponding output values. The goal of the model is to minimize the difference between the model output and the actual values. There are many supervised learning methods, which are shortly outlined below.

K-Nearest Neighbors

kNN is a non-linear method where the predicted output is the average of the values of k nearest neighbors of the input. The Euclidean distance is commonly used for the distance metric in the input space. The kNN models are easy to interpret, fast in training, and have a small number of parameters to tune. However, the accuracy of prediction is generally limited. In the domain of radio context awareness, the kNN method has been employed, among other things, for SS [115, 161, 147]. However, the limited accuracy means that kNN is most useful in resource-constrained environments.

Naive Bayes

Naive Bayes (NB) method is based on the Bayes theorem for calculating probabilities using prior probabilities. A Naive Bayes classifier assumes that all features are conditionally independent. It requires a small amount of training data and is recommended when the dimensionality of the input is high.

Decision Tree

Decision Tree (DT) is a flow-chart model in which each internal node represents a test on an attribute. Each leaf node represents a response, and a branch represents the outcome of the test. DT have parameters such as desired depth and the number of leaves in the tree. They do not require any prior knowledge of data and are robust against outliers or label noise in data [54]. The complexity-cost of using a tree is logarithmic in the number of data points provided for training. Decision trees may be biased if some classes dominate the training dataset. Therefore, a balanced dataset is required before fitting. Unlike other methods, decision trees can process categorical and numerical data even without data normalization. Decision trees have been applied for SS [154].

Support Vector Machine

Support Vector Machine (SVM) uses training data to develop a hyperplane that separates classes with the most significant margin. The sample points that form the margin are called support vectors and establish the final model. When a good linear separator cannot be found, kernel techniques project data points into a higher dimensional space where they can become linearly separable. Thus, choosing kernel parameters is crucial for obtaining good results. This method generally shows high prediction accuracy and can behave well with non-linear problems when using appropriate kernel methods. An exhaustive search must be conducted on the parameter space, thus complicating the task. Since the optimal problem solution by SVMs is convex, SVMs deliver a unique solution, unlike Neural Networks, which provide multiple solutions associated with local minima. SVMs may provide high performance for minor problems. However, their computation and storage requirements increase rapidly with the number of training vectors. SVMs are not scale invariant. Therefore, the data need to be re-scaled before being fed as input into SVM. An SVM classifier can be used for SS and decision-making [74, 103, 71, 29, 157].

Artificial Neural Networks

An artificial NN is a statistical learning model consisting of interconnected nodes, also called neurons [57]. A neuron gets information from all neighboring neurons and gives an output depending on its activation functions. Adaptive weights represent the connection strengths between neurons. During the learning process, the weights are adjusted until the network output is approximately equal to the desired output.

As a particular type of NN, CNN uses convolution operations with a set of kernels (filters) instead of employing total connections between layers of neurons [57]. Since convolution operations are invariant in translation, CNNs help analyze spatial data. Another type of NN, RNN, is designed for modeling sequential data, where sequential correlations exist between samples. RNN uses recurrent connections from

a neuron in one layer to neurons in previous layers. In training a traditional RNN, vanishing or exploding gradient problems frequently occur, making them hard to train. The Long Short-Term Memory (LSTM) is a particular kind of RNN that mitigates these issues by introducing a set of gates [51].

The connection pattern between different layers of neurons, the learning process for updating the weights of interconnections, and the activation function that converts a neuron's weighted input to its output activation are the most critical parameters to be trained. Neural networks may face slow training depending on the network size. They provide multiple solutions associated with local minima and, for this reason, may not be robust over different samples. NN can be used at the SU 's end in SS and adapting radio parameters in cognitive radio [162, 36, 92].

1.2.2 Unsupervised Learning

In unsupervised learning, only unlabeled data is provided, and the goal of a model is to find a pattern in data. The most common applications of unsupervised learning methods are clustering, dimensionality reduction, and anomaly detection.

Clustering

Clustering aims to identify data groups and build a representation of the input. Clustering methods can be classified as non-overlapping, hierarchical, and overlapping. Among non-overlapping methods, K-means clustering and Self-Organizing Map (SOM)s are most popular. K-means clustering aims to partition observations into clusters so that each observation belongs to a cluster with the nearest mean and within-cluster variance is minimized. The k-means applications for context awareness improvement mainly perform sensing [199, 103]. The most common algorithm uses an iterative refinement technique: k clusters are created by associating every observation with the nearest mean, and then the centroid of each of the k clusters becomes the new mean. In SOM, unlabeled data are fed into a neural network to produce a low-dimensional, discretized representation of the input space of training samples, called a map. In overlapping clustering, an observation can exist simultaneously in multiple clusters. Gaussian mixture models belong to this class of methods as well.

Dimensionality Reduction Algorithms

Dimensionality reduction methods produce lower-dimensional models of high-dimensional datasets. Principal Component Analysis (PCA) achieves this by creating new combinations of features, which project data onto a lower dimensional subspace by identifying correlated features in data distribution. The principal components (PCs) with the most significant variance are retained, and all others are discarded to preserve maximum information and retain minimal redundancy. PCA can be useful in modulation recognition [132, 46].

1.2.3 Reinforcement Learning

The purpose of Reinforcement Learning (RL) is to learn an optimal (or sub-optimal) policy maximizing the so-called reward function based on the observed immediate rewards through dedicated agents. The RL may be model-based (where

the dedicated model is generated) or model-free [160]. A basic reinforcement learning model consists of environment states, possible actions, rules for transition between states, rewards of transitions, and rules for observation. The learner, called an agent, interacts continuously with the environment by selecting actions. The environment changes by responding to these actions, and the agent receives numerical rewards as the environment responds. In RL, the agent tries to maximize rewards over time. Learning can be centralized in a single agent or distributed across multiple agents. Reinforcement learning is useful in sequential decision and control problems where it is impossible to provide explicit supervision, and only a reward function can be given. Using RL, each SU can sense the spectrum, perceive its current transmission parameters, and take necessary actions when a PU appears.

Q-Learning

There are many RL algorithms, and the review of all of them is out of the scope of this thesis. One of the popular methods is Q-learning, for example, used for discovering the optimal spectrum sensing policy [21] or interference control [49]. In Q-learning, the algorithm computes the expected rewards (Q) of an action taken in a given state, independent of the policy being followed [160]. Then, the next action is chosen using a policy derived from value Q, and the observed rewards are used to update Q with the weighted average of the old value and new information.

Supervised learning algorithms are the most developed of all machine learning methods and are most frequently used in ML applications for radio-context awareness considered in the literature. For example, supervised learning can be applied in SS and user classification. A possible drawback of supervised learning is the requirement for labeled data, which can be time-consuming to acquire. Unsupervised learning, employed less commonly than supervised learning, automatically finds patterns in large data. Reinforcement learning is primarily suitable for complex network-control problems.

1.3 Machine Learning for Spectrum Sensing

While discussing context awareness and the process of collecting information about the environment, a natural and logical association could be made with the contributions in the CR domain. In general, CR and Cognitive Radio Network (CRN)s are assumed to follow the well-known cognitive cycle to optimize a network's overall functioning and improve its performance. Within this cycle, CR and CRN tend to observe the environment, learn, and perform decisions based on collected data. It is evident that observation of the environment, also through SS or access to some databases, is a rudimentary requirement of the CR technology [116, 62]. As CR has been investigated for over two decades, the author starts analyzing AI-based context awareness by reviewing recent findings in this domain.

SS is, in principle, the process of observing a given spectrum portion and making a decision about the presence/absence of a licensed signal in the observed band at the given location. In the last twenty years in the literature, many schemes have been proposed for efficient SS. These methods may be blind or use some apriori knowledge about PU signals and noise variance [128, 198]. Well-known representatives of the first group of SS algorithms are eigenvalue-based detection, higher-order-statistics-based detection, or a solution focusing on the symmetry property of the cyclic

autocorrelation function. Next, the well-known semi-blind technique requiring just the knowledge of noise variance is energy detection, i.e., a simple scheme yet highly inaccurate in low SNR regions. On the other hand, a classic non-blind method is matched-filtering [140].

Irrespective of the selected SS technique, a sensing entity can make correct or incorrect SS decisions, and the spectrum band can be occupied or vacant, resulting in four cases constituting a well-known confusion matrix. When a node detects a signal and it is indeed present, one refers to the probability of detection P_{d}. In the case of signal absence and correct decision, the correct-negative scheme is considered. Next, two types of errors are known - false positive described by the probability of false alarm P_{fa} (the probability that the signal is absent but it is decided to be present), and false negative measured typically as the probability of misdetection P_{md} (signal is present but it is not detected). The decision process based on these estimated probabilities is referred to as the double-hypothesis statistic test [39]. Typically, the performance of any receiver is expressed through the so-called receiver operating characteristics (ROC), which is the plot of a true positive rate (TPR) (which is the estimation of P_{d}) against a false positive rate (FPR) (which is the estimation of P_{fa}) for various test-function threshold settings. The area under the Receiver Operating Characteristic (ROC) curve is widely used for sensing performance evaluation and is denoted as AUROC, i.e., the area under the receiver operating characteristics.

ML is becoming increasingly popular for improving or even replacing traditional SS methods. The use of artificial intelligence algorithms in spectrum sensing allows for not only deciding on the PU's transmission state [89, 74, 166, 195, 194, 4] but also enables the estimation of the number or localization of active PUs [70, 200, 9, 188]. ML can also be useful in SS with feature recognition, which can be used in PU and SU signal differentiation [157] or PU behavior recognition. Since the best chance to find an idle spectrum band is to search in a wide frequency range, ML has also been applied for wideband sensing in sparse signals [203, 158, 79, 120, 76, 34] which is another utilization of ML in SS. Also, as an indirect way of using ML for SS, one can distinguish applying ML algorithms for SS threshold adaptation and also using ML in the Fusion Center (FC) in Cooperative Sensing as a decision method instead of typical OR, AND or majority rules [31, 118, 107, 91, 163, 103]. Last, ML can be used for future channel state prediction based on SS data. However, in this thesis, the author splits the discussion into two dominant categories: A. Single node sensing enhancements and B. Cooperative sensing improvements, including advanced data fusion techniques. Within each part, the analysis of the existing solutions is arranged from two perspectives: the pure sensing process and the resultant decision-making.

1.3.1 Single Node Sensing Improvement

Various AI/ML techniques have been proposed to improve single-node SS, focusing on specific aspects of this procedure. A comprehensive comparison of numerous techniques has been presented in [89]. The authors have verified the performance of 13 detection methods (including 11 ML methods) for SPN-43 radar detection. Actual radar transmission data in the form of spectrograms were used as input data for ML. Three classical ML algorithms have been chosen, namely SVM, kNN, and a Gaussian Mixture Model (GMM). The remaining eight applied ML algorithms are deep learning methods, and more specifically CNN methods: VGG-16, VGG-19, ResNet-18, ResNet-50, the Inception-V1 network, DenseNet-121, and two

algorithms designed by the authors; one CNN algorithm, and LSTM. All ML algorithm performances were compared with classical SS methods: energy detection and sweep-integrated energy detection. ROC curves have been drawn to compare the performance of the used algorithms, and the speed of algorithms performances has been compared. The authors claimed that based on the evaluations of real-world data, the superiority of ML-based detection was proved when compared to the energy-based scheme.

A similar comparison of the performance of ML-based solutions and traditional SS schemes has been presented in [166]. The authors raise an essential question: when is it better to use ML techniques, and when is it more beneficial to rely on classical statistical signal processing methods? ML and signal processing methods were proposed for multiple transmitter detection and automatic modulation classification to investigate this issue. Two ML algorithms were proposed for multiple transmitter transmission detection: TxMiner based on a Rayleigh-Gaussian mixture model and a Log-Rayleigh mixture model. Both algorithms' performances have been compared with the signal processing method: multiple hypothesis testing based on normalized threshold binning. One ML algorithm, namely kNN, and one signal processing method (maximum likelihood) have been used for automatic modulation classification. However, the authors claimed a significant trade-off between accuracy and computation/implementation complexity exists. In particular, it has been shown that ML-based solutions offer better accuracy at the expense of significantly higher complexity.

Application of classifiers

As SS may easily be represented as a classification, various AI-based classification tools have been considered in numerous papers. For example, in [74], the SVM algorithm has been proposed to classify spectrum into free or occupied classes. The category of the free spectrum is further classified into a few subcategories that indicate what power SU can use to transmit. This approach is supposed to minimize the interference level in the case of misdetection. Other papers that consider the application of SVM-based solutions for single node SS are, e.g.,[63, 42, 144]. Also, [190] applies SVM to achieve better sensing performance while combining it with genetic algorithms and self-organizing maps. In [16], eigenvalue-based spectrum sensing with SVM in a multi-antenna cognitive radio was investigated. It has been further evaluated in [13], where SVM has been adopted for temporal, as well as joint spatiotemporal sensing, together with beamformer-aided feature extraction for enhancing the capability of SVM. The method allows for determining the number of active PUs and their locations in the network during the sensing interval. Least square regression, SVM, and manifold learning have been applied to classify features extracted by the energy detector, waveform-based sensing, and cyclostationarity-based sensing in [102].

Zhang et al. in [199] proposed a machine learning-based SS framework for a scenario where PU operates on multiple transmit power levels. The method does not require prior information on PU or the environment. Before sensing, SU undergoes a learning phase, where K-means clustering is applied to discover PU's transmission patterns and statistics. Then, SVM is implemented to train SU to distinguish PU's status based on energy feature vectors. A similar approach is proposed in [189] for SS using a sample covariance matrix of the received signal vector from

multiple antennas. K-means clustering is performed to discover the primary user's transmission patterns, and afterward, a decision is made using SVM. The feature vector used in learning is extracted from the covariance matrix. It consists of the maximum and minimum eigenvalues ratio and the ratio between the absolute sum of all matrix elements and diagonal elements.

In [154], the authors applied several ML methods for SS to data from the Global System for Mobile Communications (GSM) 850 MHz band on one day in March 2016. Records of the power of a radio channel have been obtained in 290 ms intervals throughout 15 h per day. kNN, SVM, Logistic Regression (LR), and DT classifiers have been investigated. The best results were obtained using the DT classifier. However, SVM, kNN, and LR classifiers took much less time than the DT.

Although sub-Nyquist SS in the narrowband case was researched in [120], the authors claim the proposed algorithm could also be used in wideband after some modifications. The algorithm uses a low sampling rate and a learned dictionary to recover the sampled signal. In the end, ML classifiers were used to enhance detection. Two ML algorithms are tested: SVM and deep NN. As feature vectors, absolute gradients are used. The ML algorithm significantly improved detection performance, which is shown using the probability of detection and false alarm for different SNR values. The algorithm has also been tested using lab measurements.

Finally, the Kalman filter-based channel estimation technique for tracking a temporally correlated slow-fading channel is worth mentioning, as presented in [14]. This technique adapts parametric classifiers to changing channel conditions. Moreover, [165] proposed SS in an OFDM system using an NB classifier. A class reduction-assisted NB method was used to train the model and reduce spectrum sensing time. The method has been tested using a second-generation terrestrial digital video broadcasting (DVB-T2) system simulation. It has been shown that compared with non-ML methods, the proposed method achieves higher spectrum sensing accuracy, particularly in critical areas of low SNRs.

Application of neural networks

Another big class of AI tools applied for SS are NN-based solutions. In [162], NN has been applied to predict the state of a radio channel using the results of energy detection and cyclic spectrum feature detection as input of the network. This approach improved the detection of PU at low SNR. NN was also proposed in [172] whereas input features, energy (from energy detection and Likelihood Ratio Test statistic) was used. The method has been shown to outperform the classical energy detection method.

In [123], the authors employed CNN for SS in design, experimental assessment, and Software-Defined Radio (SDR) implementation of the SU link. One-dimensional acCNN has also been applied for SS in [60]. As an input to CNN, a matrix composed of energy and cyclic spectrum features has been used; similar features have been employed in [162], where a Back Propagation Neural Network (BPNN) has been applied. Simulations show that the proposed algorithm has a higher detection probability than cyclostationary feature detection. Also, in [201], the authors proposed a method based on CNN for SS of the Orthogonal Frequency-Division Multiplexing (OFDM) signal. According to this method, a covariance matrix is normalized and transformed into a gray-level representation, which is classified using CNN. Simulation experiments were performed to examine the effectiveness of

the algorithm. In the simulation platform, the transmitter uses packaged OFDM signal frame data based on the 802.11a protocol, and the data stream is subject to Rayleigh fading and Gaussian white noise. It has been shown that the algorithm performs well in low SNR environments and can be rapidly trained.

Next, a mixture of CNN and LSTM was considered in [191], where a DL model for SS was investigated. The dataset for the experiments in this paper was obtained from a radio frequency signal sampled from digital radio. The experiments have shown that for in-band SNR in the range from $-9\,\mathrm{dB}$ to $-5\,\mathrm{dB}$, the proposed model can achieve a $25-38\%$ performance improvement over the energy detection method. This performance improvement does not require the introduction of any prior information on the signal of interest.

Finally, various papers investigated the usage of autoencoders. The so-called autoencoder-based SS has been proposed in [188], and the stacked autoencoder-based SS method with time-frequency domain signals has been used to detect the activity states of PU in OFDM signals. The methods allowed the authors to detect PU activity solely based on the received signals and proved robust to noise uncertainty, timing delay, and carrier frequency offset.

Application of Q-learning

Another big AI class of solutions proposed for signal detection was Q-learning and Gaussian mixtures. In particular, in [49], real-time multi-agent RL, known as decentralized Q-learning, has been proposed to manage the aggregated interference generated by multiple SUs. Similarly, in [21], a reinforcement learning algorithm that allows each autonomous SU to learn its own SS policy distributively has been developed, assuming that PU channel occupancy follows a Markovian evolution (mainly, a Q-learning algorithm with a decentralized, partially observable Markov decision process).

Prediction of the spectrum occupancy

AI tools are widely used for future prediction, and this application has also been evaluated in the context of SS and prospective spectrum occupancy identification. The prediction of signal presence exploits the temporal correlation of the collected historical signal. This approach can be very beneficial in terms of time and energy consumption.

In particular, in [195], the application of LSTM for spectrum state prediction was considered. Two spectrum datasets collected by measurements were used as input data for experiments: GSM1800 downlink and satellite signals. The temporal correlation of collected data was utilized to make a future spectrum state prediction. The Taguchi method has been introduced to determine the network's architecture. Next, in [194], an LSTM network was used to predict future channel state. As the signal was to be detected, a frequency hopping signal was used. As historical data was used to predict the future spectrum state, SS results for a specific frequency and timeslot were used. The authors in [4] employed two NN algorithms for spectrum occupancy prediction, mainly the multilayer perceptron and RNN, and also two SVM algorithms: SVM with Linear Kernel and SVM with Gaussian Kernel. Also, three different network traffic models were analyzed: Poisson, interrupted Poisson, and self-similar traffic.

Other AI tools

The variety of existing AI tools is very high and can be classified in various ways. Thus, this section shows a small glimpse of other AI tools that have not been assigned to the previous key groups. For example, in [200], the proposed method to improve SS performance for low SNR is a modified cyclostationarity feature detection algorithm based on softmax regression. As ML features, characteristic cyclic values are extracted from the spectrum when the signal is present and not. The softmax regression is trained using this feature dataset. The Hidden Markov model has also been considered in [37], where an algorithm for estimating channel parameters based on expectation maximization is proposed. Similarly, the authors in [109] investigated an expectation maximization-based SS algorithm. The number of active users in a given frequency band, the power received from each user, the occupied time slots, and the noise floor were estimated. The received estimated power was modeled as a Gaussian mixture; the Gaussian with the lowest mean is associated with the noise floor and used to estimate an adaptive threshold. The method has been validated in a Wi-Fi experimental setup, where real-world data have been acquired with a SDR.

To better classify various single-node spectrum sensing algorithms, a Tab. A.1 was dedicated, which concludes all ML for SS improvement papers.

Observed trends

Based on the analysis of Tab. A.1, one can conclude that the ultimate goal of applying AI/ML tools for single-node spectrum sensing is to improve the accuracy of PU detection. Numerous approaches have been considered, spreading all classes of AI/ML algorithms: supervised, unsupervised, and reinforced. However, from the gained context information, other aspects of the observed spectrum can be identified besides the accurate knowledge of PU presence. For example, by applying advanced ML methods, the knowledge of the PU traffic pattern or types of signal classes (in terms of detected modulation type, prospective SNR, etc.) can be obtained. Moreover, reliable prediction of PU behavior can also be guaranteed. Thus, the application of AI/ML tools not only improves the performance of single-node spectrum sensing but can also be the source of some additional context information.

1.3.2 Cooperative Spectrum Sensing Improvements

In Cooperative Spectrum Sensing (CSS), the final decision on the global state of the spectrum is made in an Fusion Center (FC), which collects data from collaborating SUs present in the network. In general, SU nodes may either deliver raw sensed data (e.g., the value of measured energy) or some local decisions, and the role of FC is to process the delivered data to make reliable decisions. In the final step, the decision on spectrum occupancy may be sent back to the interested SUs. In such an approach, the decision-making process is more robust against the negative impact of a wireless channel on the sensing process. Traditionally, an FC uses AND, OR, or k-over-n rules to decide whether the spectrum is occupied or free. In CSS, ML may be used independently in each node to improve the sensing or local decision-making process or in the FC to enhance the system's performance. As the single node case has been discussed in the previous subsection, the focus now will be on AI/ML applications at the FC node.

Classification methods applied at FC

Similar to the improvements in single-node sensing, various AI-based classification methods have been proposed in the vast literature in the collaborative case. Many papers propose using ML algorithms to improve FC performance. For example, in [91], the focus was on reducing the cooperative overhead, such as overlong sensing time and energy consumption, by introducing SUs grouping algorithms. The SVM algorithm has been implemented to achieve this goal. The proposed ML framework consists of four modules: an SVM training module, an SVM classification module, a user grouping module, and a group scheduling module. The ML algorithm is trained and tested using the input energy vectors dataset. The training module is responsible for training the ML algorithm. The classifier can determine whether a given energy vector implies an occupied or free spectrum. The user grouping module groups users into different subsets depending on the usefulness of information received from users. For example, redundant SUs, SUs that suffer from severe fading, malfunctions, etc., are not included in sensing. A similar approach has been investigated in [31], where the authors proposed two-hybrid adaptive boosting (AdaBoost) algorithms, i.e., the algorithms where the so-called weak learners deliver their outputs to one entity that combines them into a weighted sum and produces boosted classifier. The first method is a decision stump-based AdaBoost, whereas the second is an SVM-based AdaBoost algorithm. The results presented in the paper were compared with SVM, kNN, K-means, and OR and AND rules.

SVM classifier is also considered in [71], where the aim was to alleviate the noise uncertainty effect by applying a novel ML algorithm called Fuzzy SVM with a nonparallel hyperplane. The authors especially emphasize that the noise level is usually unknown to SU. The proposed algorithm reduces the effect of noise on feature data by introducing the probability of each data and double hyperplanes for representing value deviations.

A two-stage cooperative SS is presented in [32]. In the first stage, offline training is performed. In the second stage, online classification takes place. The main classification algorithm is K-means clustering, which groups feature data into occupied and free channel categories. PCA extracts the features. Another approach has been verified in [163], where a learning-based NB classifier tells whether the channel is occupied or free.

The performance of three kinds of classifiers in cooperative sensing is provided in [52], namely SVM, kNN, and NB. Energy levels are used as feature vectors. The ML task is determining whether a given feature vector means the spectrum is available. However, the final decision on channel availability is made by weighted voting using the results of all three classifiers. Combining results from many classifiers can compensate for each classifier's differences in performance, as each concentrates on different aspects of data. The proposed weighted voting method is a particle swarm optimization method, which determines the weights for each ML classification decision and combines the weighted decisions linearly. As a verification, the classification error rates of the proposed method and separate ML algorithms are compared.

Another comprehensive comparison is presented in [164], where various unsupervised and supervised ML techniques in CSS are evaluated for a fixed received SNR. As in the previously cited paper, the vector of energy levels estimated by the devices is treated as a feature vector and supplied as input to the classifier, determin-

ing whether the channel is occupied. The authors considered K-means clustering, GMM, SVM, and weighted kNN. The performance of each classification technique has been quantified in terms of the average training time, sample classification delay, and ROC curve. The authors found that spectrum sensing methods based on kNN and SVM are more adaptive to changing signal environments; SVM performed better than the kNN method, whereas K-means clustering performed better than the GMM. In [113], kNN, SVM, NB, and DT classifiers are trained over a set containing energy test statistics of PU channel frames. The simulation results show that the ML classifier-based fusion algorithm has the same accuracy as the conventional fusion rules with shorter sensing time, overheads, and extra operations.

kNN for cooperative spectrum sensing has also been employed as a counting mechanism in [115]. A global energy detection threshold for different rules of decision combinations in FC is proposed, which does not consider the weight of individual SUs and their performance history. In [161], kNN is used in building a TV white space database to reconstruct the missing spectrum sensing points. kNN determines a label based on the majority of labels of the neighboring data points.

In [147], a CSS scheme based on kNN is proposed. Each SU produces a sensing report in its training phase, and local decisions are combined by majority voting at FC. At each SU, the global decision is compared to the actual PU activity, which is ascertained by an acknowledgment signal. In the classification phase, the sensing reports are sorted into sensing classes using kNN. Smith-Waterman algorithm is used to accurately calculate the distance between the current sensing report and existing members of the sensing classes. Each SU is assigned a weight based on its effectiveness. The scheme performs well even at low SNR values in a fading environment.

An interesting approach was presented in [188] for the CSS framework with mobile SUs based on non-parametric Bayesian machine learning. The beta process sticky hidden Markov model is introduced there to capture the spatial-temporal correlation in the data collected at different times and locations by various SUs. Bayesian inference is then carried out to group sensing data into different classes in an unsupervised manner, where the spectrum data in each class share a common spectrum state. Based on the classification results, the locations of PUs, and their transmission ranges are inferred by the Levenberg-Marquardt algorithm.

Finally, the application of an SVM-based cooperative decision scheme for spectrum sensing in vehicular communication environment to mitigate shadowing and multipath fading, as discussed in [29]. In the proposed scheme, individual vehicles perform sensing using energy detection, and the local results are sent to a central node, which constructs vectors of energy levels for classification. The proposed SVM-based sensing performs better than the hard fusion combining rule in a low SNR region.

Application of artificial neural networks at FC

Apart from traditional classification methods, various artificial NNs types have been considered promising tools for improving CSS.

Three fusion methods were considered in [118]: a conventional CSS model with hard fusion rules, an ML-based fusion, and a cluster-based model. In the conventional fusion model, all SUs collect energy detection results and send them to FC to evaluate global results using one of the rules: AND rule, OR rule, or majority

rule. In ML-based fusion, NN is proposed as a decision-making algorithm. Energy detection decisions and SU locations are used as input features. In the clustering model, two different fusion models are considered. In the first model, called OR-OR fusion, OR decisions are made in a given cluster head and then globally in FC. Similarly, the second model employs NN at both cluster-fusion levels.

The Extreme Learning Machine (a type of NN) devoted to high learning speed applications has been discussed in [107]. The authors considered a CRN with multiple PUs, where each PU transmits in a separate channel. FC in the proposed system receives energy vectors of length N, consisting of energies calculated by N SUs. The FC's task is to match those vectors with sets of output values that give information on which channels are occupied by PUs. The extreme learning machine has been compared with traditional SVM results.

The Ensemble Learning framework for CSS has been adopted in an OFDM signal-based cognitive radio system in [93]. The spectral coherence density is provided as input and is classified locally by CNN at each SU. SUs are considered weak learners, and the stacking strategy in the Ensemble Learning is adopted in FC to integrate their results. For this task, another deep NN learner is used in FC.

Finally, [88] proposed a deep CNN for combining sensing results in CSS. The strategy for combining single-node sensing results of SUs is learned autonomously with CNN using training sensing samples. Single-node sensing results from different bands and SUs constitute two-dimensional input data for CNN. Thus, both spectral and spatial correlation of single-node sensing outcomes are considered. The proposed scheme can achieve higher sensing accuracy than the K-out-of-N scheme or a scheme based on SVM.

AI/ML algorithms complexity reduction

As the complexity of various AI/ML methods may be very high, especially with the significant number of data entries, there is a need to find ways to optimize them. The authors of [103] investigated a new method of using a low-dimensional probability vector in ML classification instead of an N-dimensional feature vector. This method is supposed to shorten the training and classification time. The probability vector is represented as a vector of two values of the probability density function of an energy vector under the condition of PU's signal present and not present, accordingly. Two ML algorithms use these feature vectors to classify the spectrum as occupied or free: K-Means clustering and SVM algorithms. Another method substantially reducing training time is presented in [56], where SVM-based FC soft decision algorithms are proposed. One of them keeps a constant P_{fa}, and the other enables adapting the value of P_{fa}. Both focus on redefining the problem of finding a decision boundary in the SVM algorithm to make the training process faster. The results are compared with the performance of the traditional SVM algorithm.

Next, the authors in [101] proposed a RL-based cooperative sensing method to address the cooperation overhead problem and to improve cooperative gain in CRNs. In the proposed algorithm, SU, acting as the fusion center, is represented as a decision-making agent that interacts with the environment of cooperating neighbors and their observations of PU activity. The authors utilize temporal-difference learning to address cooperation overhead issues and show that the optimal solution obtained by the algorithm improves the detection performance under correlated shadowing while minimizing the control channel bandwidth requirement. The pro-

posed algorithm converges asymptotically with the option of optimal stopping for fast response in a dynamic environment, mitigates the impact of control channel fading, improves the reliability of user and sensing data selection, and adapts to PU activity changes and the movement of SUs.

Other AI/ML tools considered

Similarly, as in single-node SS, the variety of AI/ML-based schemes applied to CSS is very high. For example, in [208], the authors investigated distributed algorithms using no-regret methods to detect malicious and incapable secondary users in collaborative spectrum sensing. In [38], a linear fusion rule for CSS is developed, and the Fisher linear discriminant analysis has been used to obtain linear coefficients. In [104], the authors proposed a distributed multi-agent, multiband reinforcement learning-based sensing policy. The sensing policy employs SU collaboration with neighbor SUs through local interactions. In [12], the expectation-maximization algorithm for detecting PU in multi-antenna cognitive radio networks was investigated. The PU signal is detected, and the unknown channel frequency responses and noise variances over multiple subbands are jointly estimated iteratively. A distributed implementation of the proposed scheme to reduce communication overhead is researched.

In order to concisely summarize the above-discussed papers, a dedicated table has been created - see Table A.2, which contains the summary of papers regarding decision-making in FC.

Observed trends

As an extension of single-node spectrum sensing, cooperative spectrum sensing benefits similarly from applying AI/ML tools. The ultimate goal of most solutions is to improve the performance of PU presence or absence detection. However, as in previous cases, other kinds of information can be fetched besides the knowledge of the PU activity. In particular, the presence of multiple sensing nodes allows for the deduction of the PU signal source location and the number of PU signals. Moreover, the observed signal quality can be estimated much better when AI/ML tools are applied to data collected from many sources.

1.4 Resource occupation pattern recognition

Occupied resources pattern recognition recognizes statistical properties or time, frequency, and dependencies in the received signal. By detecting those dependencies, it is easier to estimate the current spectrum state or predict the spectrum state in the near future. Predicting the future spectrum state translates into more efficient reuse of spectral resources, better spectrum management, shorter sensing time, and, therefore also, lower energy consumption. Other users' traffic patterns depend on the statistical distribution of PU activity, sensing area, time of day, telecommunication system, etc.

The other users' activity statistics are not always known, the transmitted signal is complex, and subtle transmission space-time-frequency dependencies and correlations are hard to see. ML algorithms are beneficial for this application.

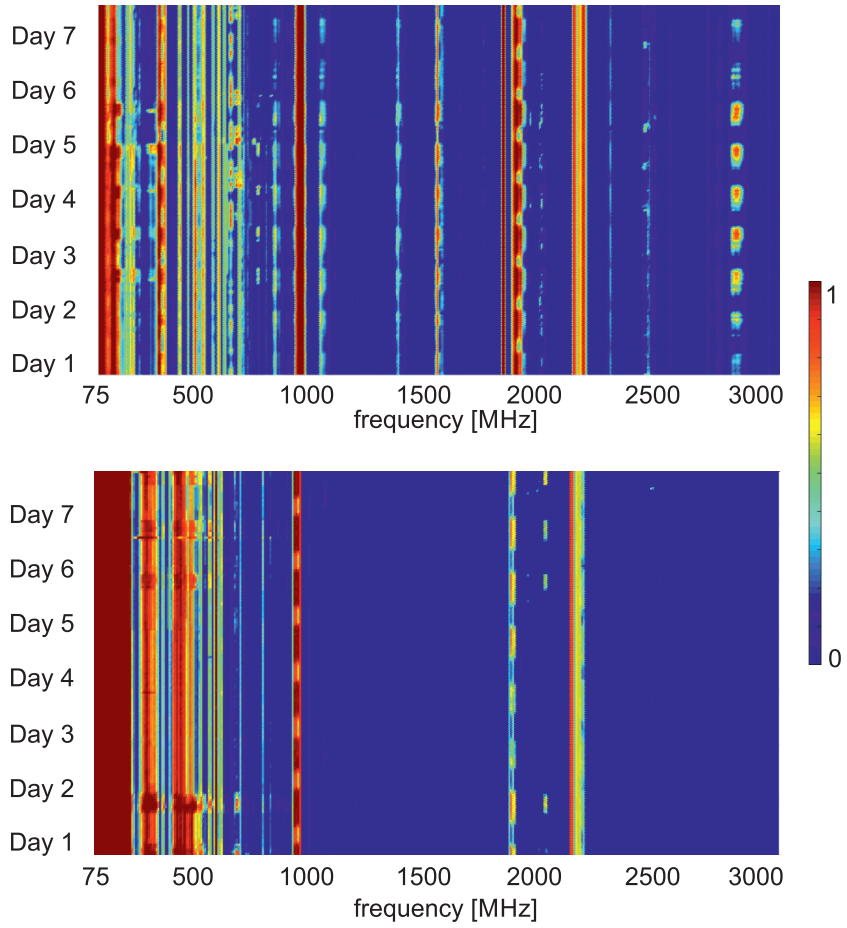


FIGURE 1.5: Examples of measured real-data time-frequency patterns of wireless signals: top figure - the measurements carried out outdoors (rooftop); bottom figure - the measurements carried out indoors. (With the permission of colleagues from the Institute of Radiocommunications, who carried the measurements at the Poznan University of Technology.)

The main pattern recognition methods employing ML methods can be categorized into the following groups: time pattern recognition, frequency pattern recognition, spatial pattern recognition, or a combination of any of those. The combined time and frequency patterns can be seen in Figure 1.5. This figure shows two scenarios: one outdoors and one indoors. One can observe that the location of measurements greatly impacts the results. Last but not least, the spatial pattern in the form of different SNR values in different locations in space is shown in Figure 1.6. The most common causes of traffic pattern occurrence are summarized in Figure 1.7. Usually, ML is not used to find the patterns explicitly but rather to predict the next signal occurrence or probability of its occurrence in the future.

Here below, the state-of-the-art in AI/ML for traffic pattern recognition is presented. All the papers considered below have been categorized, compared, and summarized in Appendix A in Table A.3.

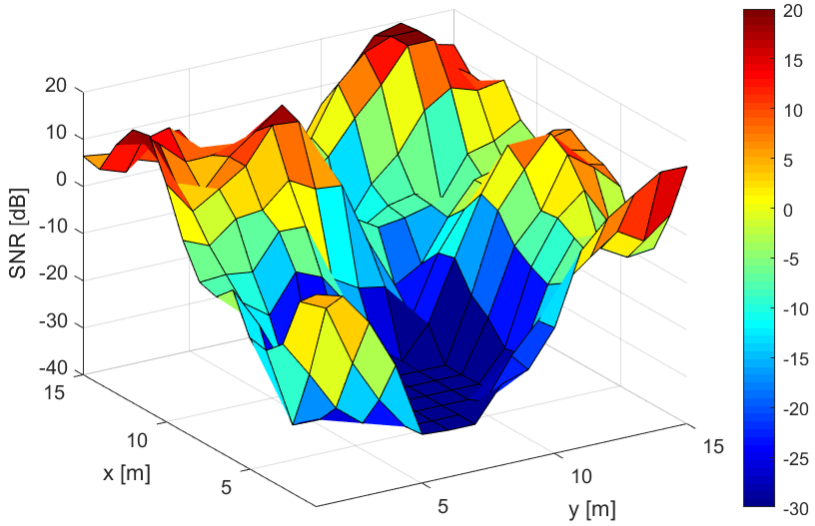


FIGURE 1.6: SNR values varying in space. Here, the AWGN and fading effect are causing variety of SNR values. The path loss is not taken into account.

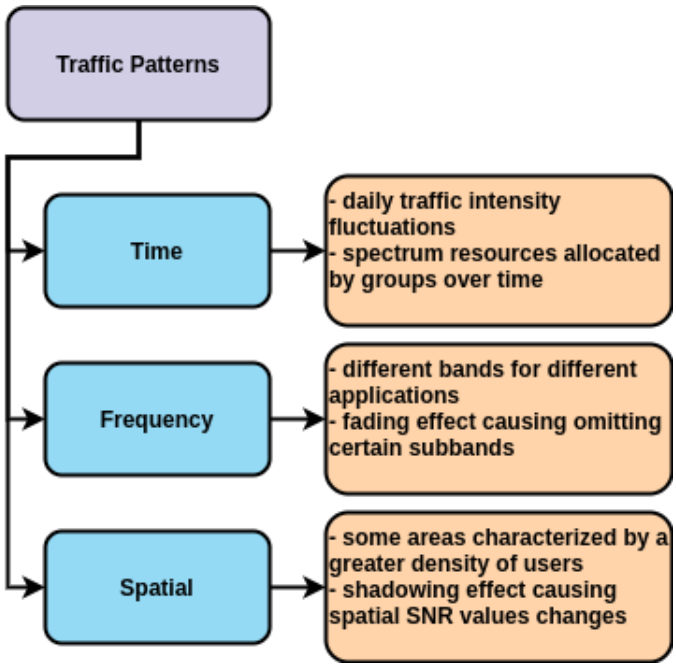


FIGURE 1.7: Types of traffic patterns occurring in telecommunication signals

1.4.1 Sensing and Prediction of Signals with Time Dependencies

The typical approach to finding sensed data patterns is looking for temporal dependencies and correlations. Knowing the history of the signal's occurrence in time can improve its sensing or predict its occurrence in the future.

One of the most popular ML algorithms for finding sequence dependencies is RNN. Rutagemwa *et al.* [138] employed RNN to learn the time-varying probability distribution of received power samples. RNN predicts the following samples and makes it possible to establish the suitability of sharing the channel with other users.

Roy et al. [137] also proposed using RNN to take advantage of the temporal statistical distribution of received signals. The received data samples are In-Phase and Quadrature (IQ) samples, and several samples are collected for each time interval. Therefore, the collected data is in the form of two-dimensional temporal data, where the first dimension is the time dimension and the second dimension is the multiple samples per time interval. This approach allows it to use CNN to take advantage of the two-dimensional form of the data. The authors propose a hybrid of RNN and CNN algorithms, the ConvLSTM algorithm, to improve prediction results. Another example of RNN usage for detecting signals correlated in time has been presented by Hamedani et al. [59], where a new class of RNN, namely, a delayed feedback reservoir, has been applied.

In older papers, one can find other, less complex ML algorithms employed for prediction. For example, Zhang *et al.* [203] proposed Suppor Vector Regression (SVR)-based online learning, where the past signal power measurements for a given frequency are used to establish the probability of the next spectrum occupancy state. SU then uses the probabilities to decide which channel to select for transmission.

1.4.2 Sensing and Prediction of Signals with Time and Frequency Dependencies

Another approach to predicting future spectrum states is combining received signals in time and frequency to create two-dimensional data matrices or spectrograms and use those to find combined temporal and frequency dependencies. The ML algorithm usually used for this application is CNN. CNN deep learning algorithms are usually used to analyze images, so their application in any two-dimensional dataset seems appropriate. Camelo *et al.* [26] used CNN on spectrum samples that are combined into a spectrogram image. The proposed algorithm can predict the following signals and detect transport protocols and transmission rates. Wasilewska *et al.* [179] used the temporal and frequency dependencies to predict the state of a few following LTE/5G time slots. The LTE/5G resources are allocated in bunches, so the probability of adjacent resource blocks being of the same allocated/idle state is high. Three deep learning algorithms are employed: deep NN, RNN, and CNN. RNN is trained to recognize the time dependencies for each frequency separately, and CNN is trained to treat the spectrum energy data as two-dimensional images.

In [195], the RNN algorithm has been used for each frequency separately to find idle spectrum in measured GSM and satellite data. In this paper, the main focus is finding the best architecture of the DL RNN by employing the Taguchi method. Compared to [195], where RNN works separately for different frequency channels, in [197], the proposed deep learning algorithm based on LSTM can work jointly for multiple channels at the same time and predict the following spectrum occupancy states for those channels. Joint time/frequency sensing can be applied in IoT systems as well. As shown in [67], IoT data is transmitted as frames that create rectangular shapes in the time and frequency domains. The clustering algorithm is applied to find data points closely located in the time and frequency domains as a transmitted signal while discarding the scattered points as falsely detected noise.

1.4.3 Sensing and Prediction of Signals with Time, Frequency and Spatial Dependencies

Last but not least, the most comprehensive solution is to combine spectrum data into three-dimensional datasets that contain information on signals in the time, frequency, and space domains. This approach enables obtaining the most comprehensive context information but also requires a lot of processing and calculations.

Liu *et al.* [98] proposed theoretical solutions for CSS where broadband big-spectrum data are collected by many users and analyzed in FC servers. This paper proposes that ML algorithms be used in two stages: in front-end ML and back-end ML. To extract relevant spectrum features, front-end ML is used, and after preprocessing the obtained feature data in order to generate time-frequency data maps, a back-end ML model obtains spectrum prediction information. K-means clustering is proposed in the front-end ML model, which groups received data into categories of different communication systems; then, for each category, different back-end ML can be used according to the classified data characteristics.

Although the most intricate deep learning algorithms are popular in processing data of this complexity, simple algorithms can be employed in finding space-, time-, and frequency dependencies as well. Wasilewska *et al.* [177, 178] focused on signal sensing for which there occur strong correlations in time, frequency and space. In [177] sensing is performed separately for different locations in space, but in [178] the learning is performed including localization information as ML input. In both of the papers, simple ML algorithms have been employed, namely, kNN and random forest.

As Table A.3 shows, ML for traffic pattern recognition usually employs those detected patterns to predict the future spectrum state. The input datasets vary from simple IQ samples to energy values to spectrum sensing decisions. As predicting future spectrum states requires analyzing signals as time sequences, DL methods, especially RNN algorithms, are prevalent. CNN algorithms are usually employed if the patterns are both in time and frequency.

1.5 Context-Awareness Design Trade-offs, and Recommendations

As discussed in the previous sections, the role of context information for the expected performance of future radio communication systems must be considered and has been emphasized in many recent papers. Thus, the definition of a context-information framework for its acquisition, representation, and distribution and the definition of the suitable architecture, either centralized, distributed, or mixed, are essential for future radio communications' broadly understood efficiency. This section summarizes the design trade-offs and recommendations for such an architecture.

1.5.1 Signalling Overhead vs. Reliability

Signaling overhead and information reliability are directly related to the key performance metrics of a radio network. Both reflect the methods and places of acquiring, representing, modifying, and disseminating context information. Signaling overhead (the cost) must be balanced with the performance improvement (the

profit) that comes with exploiting context information. This is one of the main challenges of deploying distributed databases for environmental information. The signaling overhead can be significantly reduced by designing an architecture that carefully considers the type and amount of information exchanged between different network layers and entities. Signaling overhead can also be reduced by a suitable choice of dissemination strategy.

An on-demand model is most appropriate when information is needed only rarely. A proactive model, on the other hand, may result in better performance for commonly needed and dynamically changing data. The AI/ML algorithms residing at the appropriate network point, e.g., at the network edge or dedicated subsystem, can reduce the signaling overhead by avoiding transmitting data that can be retrieved by learning.

The required information accuracy and reliability depend on context-information usage and timescale objectives. For instance, fast power control requires high precision, while dynamic spectrum allocation performs well with approximate or statistical information. The reliability also depends on time dynamics and regional characteristics of information aggregation. The choice of large regions over a long time to reduce signaling overhead would come at the expense of the model's accuracy or statistical characterization. Thus, whenever AI/ML methods are applied to enrich CA, they must converge at a required pace, responding to the system dynamics and with accuracy tailored to the application.

1.5.2 Context-information Acquisition, Storage and Distribution vs. Power Consumption

A popular concept of providing context information is to have it stored, dynamically updated, and made available in a centralized database (a storage unit) with an accompanying AI/ML engine to capture the dynamics and hidden dependencies of the information arriving from agents.

The opposite approach to acquiring, storing, and distributing context information relies on a fully distributed architecture and edge intelligence (AI in the end devices). Distributed architecture and the radio context information subsystem operation require significant additional traffic between the network devices, resulting in energy consumption. The *Smart Dust* project implemented in the University of Berkeley explored the limits on size and power consumption in autonomous sensor nodes [77]. This concept can be understood as the decentralized acquisition and distribution of pieces of information. It incorporates the requisite sensing, communication, and computing hardware, along with a power supply, in a few cubic millimeters volume while still achieving the required performance in terms of sensor functionality and communication capability. The networking nodes consume low power, communicate at bit rates measured in kbits per second, and potentially operate in high volumetric densities. However, they do not initially possess any AI.

Considering future networks with various degrees of node mobility and density, it is possible that the idea of a network of simple low-power sensors (or information points) exchanging pieces of information but applying edge AI/ML algorithms could augment the concept of highly reliable (but costly) centralized databases at low energy cost. A significant challenge is to define and incorporate the required functionalities of sensors/nodes for context-information acquisition, storage, and distribution while maintaining low power consumption.

1.5.3 Reduced vs. Incomplete Information

The required reliability and the amount of context information in a network can be generalized by models of complete information vs. incomplete information and full information vs. reduced information. In game theory, a metric describing the cost of not having complete information, the *Price of Ignorance*, is defined as a relative loss of common welfare (e.g., network performance) that results from incomplete information.

Network performance can be defined in several ways, e.g., as the total network energy saving, its spectral efficiency (sum throughput over available bandwidth), or sum-throughput net. Ignorance can be understood as either uncertainty of information or possessing complete (specific) information, which has a reduced representation of the information describing the players' environmental conditions and options in detail.

For instance, one can consider channel state information required for optimal resource allocation in a network. Providing complete information on all link qualities of all players to all other players in the considered network is associated with a considerable communication cost, making it impractical. The Bayesian game models, which require the fading statistics of all channels for all players to consider every player's behavior with a given probability, are even more impractical. In a dynamic radio environment, these statistics change with time. Moreover, it is impractical to consider the channel gains probability density functions with high granularity because it exponentially increases the computational complexity of calculating the equilibrium point. This example shows a fundamental trade-off between information availability, compactness, and network performance.

Again, AI/ML methods have considerable potential to uncover hidden elements of context information, transforming incomplete information into complete. However, it is impossible to infer full context information once it has been reduced.

1.5.4 Machine Learning Algorithms Design vs. Quality Datasets

Although the number of scientific papers describing the application of machine learning for context awareness is constantly increasing, most authors do not publish the datasets they used to generate results. This leads to a lack of possibilities for objectively comparing ML methods and architectures. The successful application of ML models requires high-quality datasets. Sufficient training data volume is essential, especially for larger NNs that have a large set of parameters. However, mobile network datasets are scarce. Mobile data collected by sensors or network equipment is frequently affected by loss, redundancy, and mislabeling, thus requiring cleaning before application for model training. In addition, mobile service providers and operators keep the collected data confidential and are reluctant to share them for research purposes.

Moreover, the absence of public mobile network datasets leads to another problem: many investigations are performed on private data. Without comparing the performance of various models on the same data, it is hard to design and select the approach that works best and decide in what aspect it could be improved.

1.6 Chapter summary

Above, the author of this thesis has surveyed the existing literature to address the following issues: (i) What is the role of context information, its availability, and representation in contemporary and future radio communication networks? (ii) What are the suitable AI/ML methods to enrich context awareness in these networks? (iii) What kind of ML algorithms and framework are considered for autonomous and cooperative SS? (iv) How the ML-based pattern recognition methods can be used for SS and prediction in time-, frequency-, and spacial dimension? (v) What are the design trade-offs and recommendations for intelligent context-aware radio communication? The author believes that answering these questions is of particular relevance for the efficiency of her original methods proposed in the next chapters.

Chapter 2

Autonomous Machine Learning-Based Spectrum Sensing

Cognitive Radio (CR) technology's essential element is detecting the spectrum holes (i.e., spectrum resources not occupied by a PU). This may be achieved by getting the relevant information from a dedicated database (if available in a given location), often called a radio environment map, or performing SS. As explained in the previous chapter, SS is a process intended to uncover spectrum occupation and holes. It allows for taking advantage of spectrum opportunities, dynamic spectrum access, resource management for anticipated traffic, etc.

Spectrum Sensing is based on choosing between two possible hypotheses. Hypothesis $\mathcal{H}_0$ assumes that the received signal consists of noise only, while hypothesis $\mathcal{H}_1$ means that the received signal is a sum of noise and the PU's transmitted signal distorted by the channel, i.e.,

$$\begin{aligned} &\mathcal{H}_0 : y(t) = n(t), \\ &\mathcal{H}_1 : y(t) = h(t) * s(t) + n(t), \end{aligned} \tag{2.1}$$

where $y(t)$ is a received signal, $s(t)$ is the transmitted signal, $h(t)$ is the channel impulse response, $n(t)$ is the noise, and $*$ denotes linear convolution.

The goal of SS is to determine which hypothesis is more probable. A good sensing algorithm should maximize the probability of correct detection (P_{d}) while maintaining the value of the probability of false alarm (P_{fa}) close to some assumed level. Probability P_{d} is defined as the probability of (correctly) deciding through spectrum sensing that hypothesis $\mathcal{H}_1$ is true. Probability P_{fa} is the probability of making a wrong decision that hypothesis $\mathcal{H}_1$ is actual, while in fact hypothesis $\mathcal{H}_0$ is true. Essentially, SS is a binary classification problem, where P_{d} is a probability of achieving accurate positive results on signal's presence. At the same time, P_{fa} is a probability of achieving false positive results on signal's presence. Alternatively, one could test a SS method by calculating true negative and false negative probabilities, which is equivalent to measuring the accuracy of spectrum hole detection instead of signal presence detection. Therefore, the true negative probability (P_{tn}) is the probability of correctly detecting a spectrum unused resource, and the false negative probability (called the probability of misdetection: P_{md}, as already mentioned in 1.3) is the probability of not detecting a present signal, or detecting a spectrum hole, when in fact the resource is occupied. All mentioned probability measures are presented in table 2.1 for easier comprehension.

TABLE 2.1: SS performance probabilities binary classification

	signal detected	signal not detected
$\mathcal{H}_1$ is true (present signal)	true positive signal detection: P_{d}	false negative signal detection: P_{md}
$\mathcal{H}_0$ is true (absent signal)	false positive signal detection: P_{fa}	true negative signal detection: P_{tn}

The probabilities P_{md} and P_{tn} can be defined using P_{d} and P_{fa}, i.e.,

$$\begin{aligned} P_{\mathrm{md}} &= 1 - P_{\mathrm{d}}, \\ P_{\mathrm{tn}} &= 1 - P_{\mathrm{fa}}. \end{aligned} \tag{2.2}$$

Therefore, to fully measure SS method performance, only two probabilities of those four are needed, either P_{d} and P_{fa} or P_{tn} and P_{md}. In this thesis, the author decided to use the first set of probabilities, namely P_{d} and P_{fa}, that measure signal presence detection instead of free resource detection performance.

In this chapter, the author of this thesis presents her original research and results supported by simulation experiments on autonomous ML-based SS. First, in Section 2.1, ED and the ML-supported ED in the 5G system is considered and some standard methods are discussed. The details of the time and frequency patterns occurring in the signal are discussed in more detail in Section 2.2. Moreover, a new approach to improve the autonomous SS utilizing these patterns in applied ML, and based on measured EV rather than ED decisions is discussed. The simulation experiments of this approach with the application of kNN, RF, SVM, and Gaussian NB methods are presented in Section 2.3. They are compared against the standard ED-based SS. The key findings of this chapter are summarized in Section 2.4.

2.1 Basic Concept of ML-supported Energy Detection for Spectrum Sensing

2.1.1 Energy Detection

There are some conventional SS methods, such as ED, Matched Filtering, and Cyclostationarity Detection [198]. ED is the simplest method, in which the received signal power is compared with a specific threshold to determine the presence of a (noisy) PU's signal or just the noise. In contrast to Cyclostationarity Detection [47], Matched Filtering [78], and most other methods, it does not require any specific knowledge of the signal that is to be detected; however, it does require the knowledge of the channel noise power. ED methods are considered in [80, 45, 85, 8]. Kim et al. [80] propose a histogram-based method to determine the energy detection threshold. Digham et al.'s study [45] concerns ED over fading channels, and the signal of unknown characteristics is also under consideration, the same as in [85] and [8]. The mentioned articles discuss the ED method with a single decision threshold. A double-threshold ED has been discussed in [134, 171, 184]. Although, in many papers, the noise power is assumed to be known, some works present ways of noise estimation for ED purposes. Farag et al. [48] propose dynamic threshold evaluation based on noise estimation. Furthermore, papers [69, 146, 110] present other methods of noise-power estimation for energy detection.

As mentioned, ED requires knowledge of the noise power level and works poorly in low SNR regions. In the ED method, the energy of the received signal is calculated in some given time and frequency range. The calculated energy of signal $y(n)$ is a so-called test function $T(y)$. For N samples of the received complex signal $y(n)$, the test function is defined as:

$$T(y) = \frac{1}{N} \sum_{n=1}^{N} |y(n)|^2. \tag{2.3}$$

To decide whether the spectrum is occupied or not, the test function value is compared with the threshold given by the equation:

$$\lambda = \sigma_{\mathrm{n}}^2 \left(Q^{-}1(\bar{P}_{\mathrm{fa}}) \sqrt{\frac{1}{N}} + 1 \right), \tag{2.4}$$

where σ_{n}^2 is the noise power (which must be estimated), $\bar{P}_{\mathrm{fa}}$ is the assumed level of the probability of false alarm, and $Q^{-}1(\cdot)$ is the inverse Q function, which is given by:

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} \left(-\frac{u^2}{2} \right) du, \tag{2.5}$$

where variable u is a variable of integration. If the value of $T(y)$ is higher than threshold λ, the decision that the spectrum is occupied is made. Otherwise, the spectrum is considered unoccupied ("free"). Thus, the probabilities of detection P_{d} and P_{fa} are defined as:

$$\begin{aligned} P_{\mathrm{d}} &= \Pr\{T(y) > \lambda | \mathcal{H}_1\}, \\ P_{\mathrm{fa}} &= \Pr\{T(y) > \lambda | \mathcal{H}_0\}. \end{aligned} \tag{2.6}$$

Note the difference between $\bar{P}_{\mathrm{fa}}$ and P_{fa}: the first is the theoretical probability of a false alarm assumed for the calculation of the decision threshold λ, while the second is the probability of false alarm resulting from the application of this threshold. In the ideal case, i.e., when $N \to \infty$, these values should be equal. However, in this thesis, the author calculates $T(y)$ for finite N. Moreover, in the computer simulation experiments, described in this and the following chapters, both P_{d} and P_{fa} are estimated based on the finite number of repeated simulations.

2.1.2 ML-supported Decision Making for Spectrum Sensing

A promising approach to improving spectrum sensing efficiency is to predict white spaces based on detected traffic trends and the functioning of the PU. Currently, in most telecommunication traffic patterns, intensity fluctuations can be observed due to communication demand variations, e.g., daily or weekly variations [64]. Besides the time domain, dependencies and patterns can be observed in frequency and space due to, e.g., applied frequency planning and scheduling or shadowing occurring in some location areas, etc. ML techniques have been proposed as methods of SS performance improvement. ML methods can be used directly as detection algorithms. In 1.3.1, the author has already mentioned several papers regarding ML-based autonomous SS, and in 1.3.2 examples of cooperative ML-based SS have been included. ML can also be used with one of the SS methods, for example, with

the already mentioned ED. ED-based ML for SS is presented in [74], which employs an SVM-based classifier for SS enhancement. ED and ML are also a popular combination used in cooperative sensing [163, 107, 118, 114]. Tavares et al. [163] uses three Bayesian SS estimators and compares them with typical cooperative SS methods, such as Maximum Ratio Combining, AND, and OR rules. Ma et al. [107] propose using the Extreme Learning Machine for channel pattern classification in cognitive radio networks with multiple PUs. The proposed method uses an Extreme Learning Machine in the fusion center to correctly classify the channel state. Mustafa et al. [118] propose using a NN as a decision fusion scheme in the fusion center. Mikaeil et al. [114] analyze the kNN, DT, SVM, and NB ML algorithms as decision-making methods in the fusion center. ML using calculated signal energy values as energy vectors are presented in [199, 15, 52]. K-means clustering and SVM algorithms are used in [199] for discovering PU's transmission patterns and statistics and for SS, respectively. Variational Bayesian learning for the Gaussian mixture model is used in [15] as an SS method for a multi-antenna CR network. Noncooperative SS is considered in both papers. Cooperative SS is considered in [52], where SVM, kNN, and NB algorithms are used for signal classification.

The work presented in the following sections aims to show how much the probability of detection can be increased by using ML methods as a supporting component in autonomous (noncooperative) spectrum sensing. Below, SS is used to detect occupied 5G downlink signal RBs utilized by a base station (eNodeB). The ultimate goal is to create a possibility of transmitting the CR signal in the 5G New Radio standard with a flexible choice of RBs.

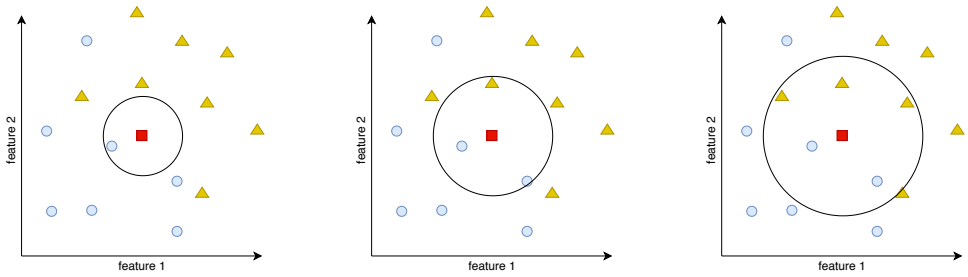
Thus, here, ED is used as the first 5G downlink signal-presence detection stage in a given area. A specific daily intensity distribution in the considered network characterizes the communication traffic. Moreover, due to 5G frequency planning and channel characteristics in the considered geographical area, some frequency resources are used with location-dependent probability. Shadowing characteristics also reflect spatial correlation. There are patterns and dependencies of RB utilization in the time-, frequency- and spatial domains. Hence, in the second stage of SS, ML is used to discover these dependencies and increase P_{d} while decreasing the P_{fa}. Two ML methods are considered: kNN and Random Forest (RF). Both algorithms are studied and compared in terms of SS performance improvement. As for the ML input data, both the ED output decision data and the detected spectrum energy values are considered, and the advantages and disadvantages of using them in a real-world scenario are discussed.

Typically, ML methods are used to achieve a near-optimum solution of the optimization problem, when it is too complex for the conventional optimality analysis and when the function mapping the input data to the output (the solution of the problem) is unknown. In radio communication systems, this kind of uncertainty is caused by noise (thermal, colored, impulse, or other) or other types of channel distortions and interference. In this chapter, ML is applied as a way of fine-sensing decision enhancement. Based on the specific 5G-RBs -occupancy decision from the first stage (ED decision, $T(y)$ value, range of the 5G subcarriers, time slot, adjacent 5G-RBs occupancy, etc.). The ML algorithm should be able to deduce whether currently examined 5G-RBs are busy or idle. Therefore, the author shall use the *classification* algorithms that classify the input data for the final binary decision. Let us consider two ML classification algorithms: the k-Nearest Neighbors and Random Forest. The reasoning behind the selection of these two is provided in the

subsections below.

k-Nearest Neighbors

kNN-based classification is considered one of the simplest ML algorithms [148]. It is a supervised algorithm, which means that, in the training phase, complete knowledge of the output data corresponding to the training input data is required. Based on the training data set, new input data can be classified into one of the output categories by calculating the distances (usually the Euclidean distance) to k closest neighbors in the used features space [40]. For example, in the case of parameter $k = 1$, only one closest data point is considered, and its output value is assigned to the label of the data point that is supposed to be classified. In the case of $k > 1$, the most numerous group of the neighboring points of one category determines the result. Figure 2.1 shows the example training data points of two categories: blue circles and yellow triangles arranged in the space of two features. When a new point (red square) from outside of the training set occurs, and an output value is to be determined, the algorithms look for k closest neighboring data points. Figure 2.1a shows that the closest neighbor of the red square is a blue circle, so, for $k = 1$, the red square is classified as a blue circle. In addition, for $k = 3$ (Figure 2.1b), two of the three closest points belong to the blue circle category. However, for $k = 5$ shown in Figure 2.1c, there are three yellow triangles and only two circles closest to the input data point, so the square is classified as a yellow triangle.



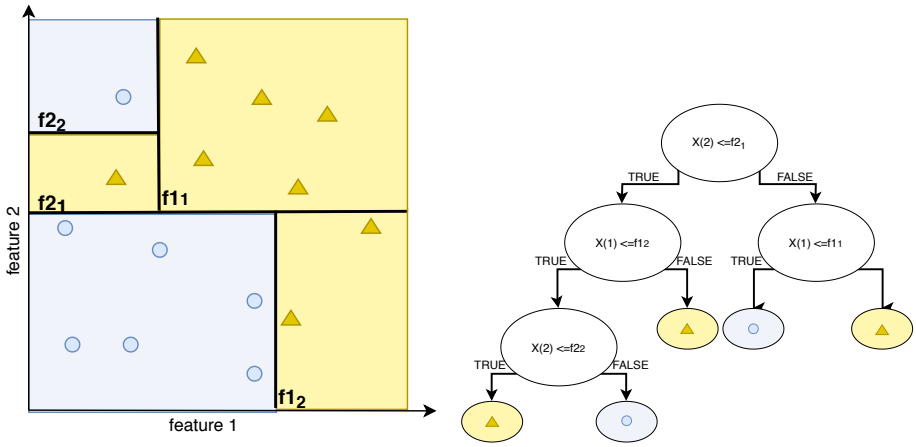
(A) $k = 1$ (B) $k = 3$ (C) $k = 5$

FIGURE 2.1: k-Nearest Neighbors (kNN)–visualization of the closest data points for different k values.

In typical 5G downlink transmission (as in the considered scenario), RBs are assigned to users in groups of several adjacent RBs, depending on a user's demand associated with a specific service. This is because the 5G scheduler tries to avoid fragmentation of resources for a user. This means the probability that a given RB is occupied is higher if the adjacent RBs (in the time or frequency domain) are also occupied. The tangential RBs occupation is not as probable. Therefore, classifying RBs based on closest neighbors, namely kNN, is an appropriate method to improve detection. However, in this case, *neighborhood* and distance should be carefully re-defined. These issues are considered in the following section. Finally, note that the kNN algorithm requires storing all the training data, and the prediction can be very slow for large sizes of training datasets and high k parameter values.

Random Forest

Random Forest (RF) is in an expanded version of the Decision Tree (DT) algorithm [119]. This work uses the classifying version of RF. RF consists of several single DTs. The DT algorithm divides the input data set into subsets, each of which means a different output category (for example, blue circles or yellow triangles, as in the example above). A DT algorithm of full depth iteratively divides a feature space into subspaces, as shown in Figure 2.2a, so that every training data point belongs to a subspace of its category. Figure 2.2b presents a decision-making algorithm for a new input data point of two features $\mathbf{X} = [X(1), X(2)]$. Thanks to feature subspaces' boundaries specified by the DT algorithm (values $f1_1, f1_2, f2_1, f2_2$), a new data point can be assigned to an output label of a blue circle or a yellow triangle.v



(A) decision boundary (B) decision tree

FIGURE 2.2: Decision tree—tree with depth 3.

The main drawbacks of DTs are that the algorithm becomes very complex for big data sets and tends to overfit. The tree that correctly assigns all training data points is most likely of considerable depth and highly overfits, i.e., results in too sharp boundaries for different data sets. One way to prevent this is to use trees of smaller depths that may not work with 100% accuracy on the training data but perform better on new input data and are less complex. Another way is to use the RF algorithm, which creates several DTs with slightly different boundaries. Single DTs used in RF might overfit on the training data, but each of them does it in a different way, which creates an averaged result after combining separate trees and prevents overfitting [65].

The occupied and unoccupied RBs will likely create regions on the time-frequency plane (also in the location domain). Therefore, the RF algorithm is proposed as an alternative to the kNN algorithm. RF does not require storing as much data as kNN, so it works faster but is still easy to implement and analyze. Algorithms such as kNN and RF are appropriate for the considered problem, with simple dependencies between data, which are too complex to analyze with standard optimization and other, even heuristic, non-ML methods.

2.2 New Algorithm for Improved ED

In this section, the author proposes the 5G-downlink signal-presence detection algorithm based on the considerations of the applicability of kNN and RF methods for improved SS. The algorithm's first step is either calculating the energies of subsequent blocks of N received signal samples (forming a vector of EV for a given data set) in a given frequency range or making ED-based hard decisions on the PU signal presence based on these energy values (forming a vector of binary values). Then, one of the considered ML algorithms is applied, i.e., either kNN or RF. The elements of the dataset, being the observed (or recorded) RBs, are characterized by properties called *features*, which can be individual, derived, or combined attributes constructed from underlying data elements. Feature definition or calculation is necessary to properly implement the proposed ML algorithm stage. Every feature is calculated for every RB in every time slot equal to the duration of a single RB. The following features are used in the proposed algorithm:

1. the index of a time slot (the smallest 5G-RB dimension in the time domain),
2. the index of the 5G basic subcarriers set (the smallest 5G-RB dimension in the frequency domain; in 5G, consisting of 12 OFDM (Orthogonal Frequency Division Multiplexing) subcarriers),
3. ED hard decision—values 0 or 1, or, alternatively, the energy value—a real number,
4. the number of diagonal (tangential) neighboring 5G-RBs detected as busy or the sum of energies of diagonal (tangential) neighboring 5G-RBs,
5. the number of adjacent neighboring 5G-RBs detected as busy or the sum of energies of adjacent neighboring 5G-RBs.
6. history coefficient with forgetting factor

The listed features are intended to improve the ML algorithm's performance by feeding it enough information about the RB occupation that ML prediction applies to and about the state of the closest neighboring RBs.

When ED hard decisions are used in the algorithm, the history coefficient with the forgetting factor is calculated as follows:

$$\phi_{ED}(m,l) = ED(m,l) + \alpha \cdot \phi_{ED}(m-1,l), \tag{2.7}$$

where $ED(m,l)$ is the energy detection decision for RB for which m is a time slot index, l is the index of the 5G subcarriers set, and α is the forgetting factor in the range of $[0,...,1]$.

In the case of using EV as inputs to the ML stage, the history coefficient is calculated as:

$$\phi_{E}(m,l) = E(m,l) + \alpha \cdot \phi_{E}(m-1,l), \tag{2.8}$$

where $E(m,l)$ is the energy value for the RB of time slot m, and frequency index l.

To illustrate data flow in the algorithm, Figure 2.3 is presented which shows the system model on the receiver side.

To illustrate the calculation of the features, Figure 2.4 is presented. It shows RBs in the time- and frequency domains as yellow and blue squares, representing

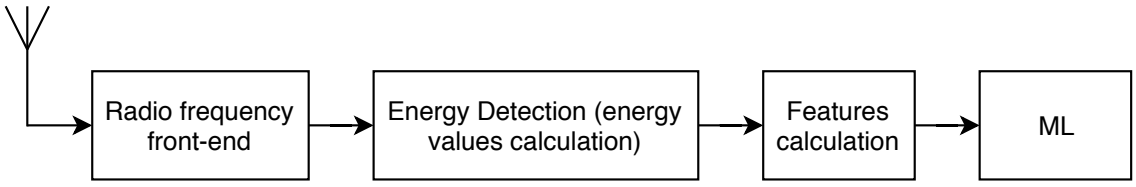


FIGURE 2.3: System model.

resources detected as occupied and free, respectively. For example, for the RB marked as **A**, the index of the time feature is m_3, and the frequency index is l_5. The ED algorithm decided that RB **A** is occupied so that the third feature can be represented by value 1. There is also one tangential neighboring RB marked as occupied by ED and one adjacent occupied RB. Assuming that m_1 is the first time slot index, the history coefficient for **A** equals 0. Thus, the feature set for RB **A** can be presented as a feature vector: $[m_3, l_5, 1, 1, 1, 0]$. Similarly for RB marked as **B**, a feature vector can be presented: $[m_5, l_4, 1, 2, 2, \phi_{ED}(m_5, l_4)]$, and for RB **C**: $[m_2, l_2, 0, 0, 1, \phi_{ED}(m_2, l_2)]$.

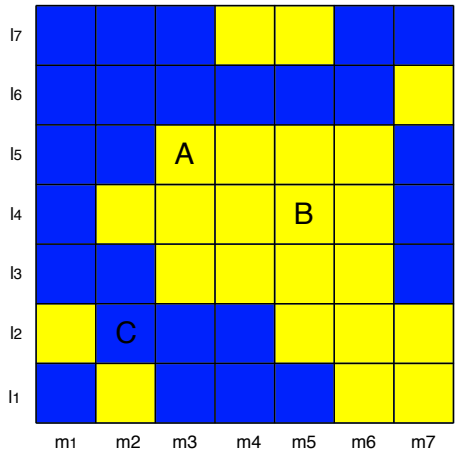


FIGURE 2.4: Example 5G Resource Blocks features.

Similar features can be calculated using energy values instead of ED decisions. Then, most of the features will no longer be represented as discrete numbers but as real numbers.

The proposed ML detection methods need to be learned separately for different SNR values. This requires knowledge of the SNR level at the specific moment. The noise power level has to be estimated for ML using ED hard decisions as the input dataset features. (Note that ML using EVs as the dataset features should not require estimation of noise power.) In order to solve this problem, the ML algorithm can learn separately for different location points, assuming that the shadowing variations in the channel are much slower than the Rayleigh fading. The idea is to train multiple separate models, where each model is trained with the assumption of previous knowledge of the SNR values in the considered area. This approach might be considered blind as it does not require noise-level estimation, but at the same time, it does require additional preceding knowledge. As long as the SNR values stay approximately the same at the given location, the model assigned to this SNR level can be used to perform SS. In addition, there is no need to train one model per

location. As long as the SNR space map remains approximately constant, only one model per a given SNR value is sufficient for multiple locations characterized by the same SNR value. The SNR values were rounded to the closest integer value in dB. In order to minimize the number of created ML models, they could be generated for some intervals of SNR values instead of specific SNR values or even rounded values. It is worth noting, though, that the wider the range of the SNR interval, the more general and less specialized ML models, and therefore their accuracy may decrease.

2.3 Simulation Experiment

2.3.1 Simulation Setup

A downlink 5G signal (OFDM signal) was generated for the bandwidth 10 MHz to test the proposed ML-improved RB energy detection algorithm. This bandwidth is divided into 50 RBs. Each RB has a standard bandwidth of 180kHz and is transmitted over 12 OFDM subcarriers. The RB time slot lasts 0.5ms. The order 1024 Inverse Fast Fourier Transform (IFFT) algorithm generates an OFDM symbol over random Binary Phase Shift Keying (BPSK)-modulated data symbols. There are seven OFDM symbols per each RB. The OFDM cyclic prefix has a length of 144 samples.

One of the reasons for using ML in SS in the proposed solution is to discover and use the time intervals when the communication traffic is low and minimize the P_{fa} in these intervals that would enable SUs to transmit. Some random periodicity has been introduced in the generated signal to reflect communication traffic variations. It has been assumed that the transmission probability is higher for some periodically occurring time slots than others. It is assumed that RB occupation follows the normal distribution for these time intervals. Thus, the probability of transmitting by PU is much lower for time slots of the lowest communication traffic.

Moreover, in the frequency domain, some regularities have also been introduced. They reflect typical 5G frequency planning, where, in a given area, some of the frequency bands are less likely to be used due to specific (adverse) channel conditions and Inter-Cell Interference Coordination (ICIC) schemes. Here, for the simplicity of implementation, it has been assumed that the probability of RB occupation is higher in the bandwidth's central frequency range. It is justified by assuming that the RB scheduler reduces interference from the neighboring frequency ranges. Figure 2.5 presents an example of the generated busy and idle RBs in the time and frequency domains. The yellow represents occupied RBs, and the blue represents idle ones.

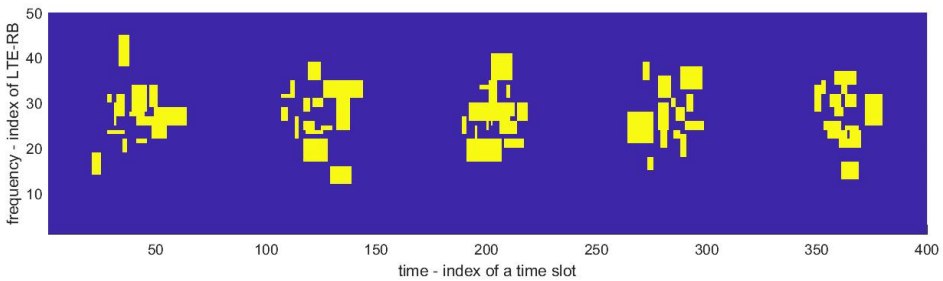


FIGURE 2.5: Example of 5G system RBs occupancy.

The generated transmission signal is sent over a multi-path Rayleigh fading radio channel with the shadowing effect and Additive White Gaussian Noise (AWGN). The Extended Pedestrian A Model (EPA) [153] has been chosen as a multi-path channel model. The shadowing effect has been simulated using log-normal distribution in the considered area. The SNR is calculated as a mean value of the signal power in a one-time slot to the noise power observed in this time slot.

The received 5G-downlink signal is the subject of the SS method described in Section 2.2. Four basic versions with various parameters have been examined: combinations of EV or ED hard decisions as the ML input dataset features, and kNN or RF as the actual algorithm for improving detection. The kNN has been tested for the parameter k values equal to 1, 3, 5, and 9. RF has been tested for one tree, 10, 50, and 100 trees. Those parameter values were chosen heuristically as it was observed that, for higher parameter values, the results were getting worse. The number of samples used in training and testing was equal to 328,000, which corresponds to RBs transmitted in 6560 time slots. A set of features has been generated for these 328,000 RBs. During testing, it was essential to maintain groups of samples together, as the data is correlated and in time in frequency. For validation purposes, the group k-fold cross-validation was performed. Training sets were composed of 292,000 samples, while testing sets were composed of 36,000 samples. Finally, based on the conducted trials, the forgetting factor α equal to 0.9 has been selected.

For the simulation of the 5G signal, channel, ED stage, and all of the calculation of the features, Matlab software (R2018a version 9.4, MathWorks, Natick, MA, USA) has been used, along with the Communications System Toolbox and Statistics and Machine Learning Toolbox. ML algorithms have been implemented using the scikit-learn library in Python [125].

2.3.2 Simulation Results

In this section, the author presents and discusses the results of the simulation experiment. The algorithms are evaluated based on the *estimated* P_{d} and P_{fa}, i.e., the number of true positive and false positive decisions of the RBs occupancy respectively, over the total number of decisions (considered RBs) taken in the simulation experiment. The word *estimated* is omitted for simplicity of description.

First, ED has been performed for different assumed values of $\bar{P}_{\mathrm{fa}}$ without using any ML algorithm. To perform ED, the threshold λ had to be calculated, requiring the noise level estimation. In the case of an OFDM signal, σ_{n}^2 is estimated by taking advantage of the fact that in the frequency domain, there are unused frequency bands around the signal transmitted within an RB. Therefore, samples from these bands can be used to calculate the noise level. The noise level has been assumed to be constant in the entire frequency range. The results are shown in Figure 2.6. For each $\bar{P}_{\mathrm{fa}}$, it can be seen that P_{d} is always higher than P_{fa}. As expected, P_{fa} remains constant and approximates the $\bar{P}_{\mathrm{fa}}$ value for each P_{d} plot for every SNR value from the examined SNR range.

Then, two ML algorithms have been evaluated: the kNN algorithm and the RF algorithm. They were examined in two cases: ED-based and EV-based. ED-based ML means that ML features were calculated based on ED results (hard, binary ED decisions on the presence or absence of the PU signal in a given RB). In the case of EV-based ML, features were calculated using detected energy values. Figure

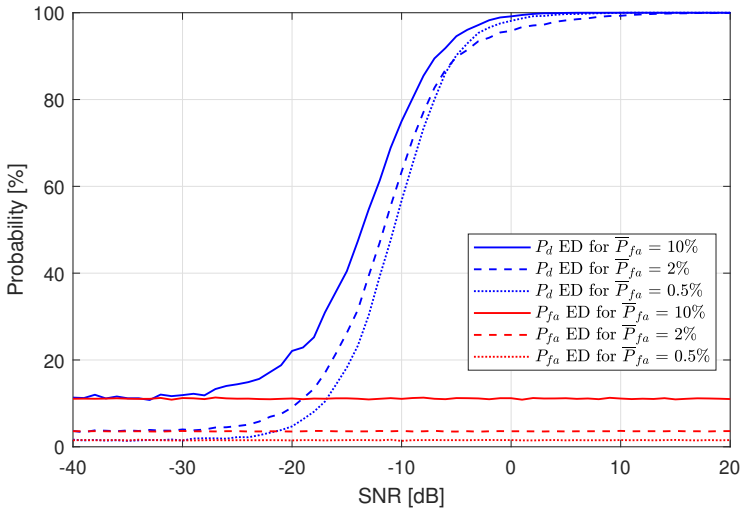


FIGURE 2.6: Probability of detection P_{d} and of false alarm P_{fa} for the Energy Detection stage for $\bar{P}_{\mathrm{fa}} = 10\%$, $\bar{P}_{\mathrm{fa}} = 2\%$ and $\bar{P}_{\mathrm{fa}} = 0.5\%$.

2.7 presents the results of the ED-based kNN algorithm for the threshold λ set for $\bar{P}_{\mathrm{fa}} = 10\%$. This value of $\bar{P}_{\mathrm{fa}}$ has been chosen because the differences between the simulation results are most visible in that case. Other values of $\bar{P}_{\mathrm{fa}}$ are analyzed later in this section. In the mentioned figure, the probability of detection P_{d} and the probability of false alarm P_{fa} at the output of the first (denoted as ED in the legend) and the second stage of the algorithm are presented. Note that the $\bar{P}_{\mathrm{fa}}$ assumed for the first stage of the algorithm (ED) may differ from the resulting value of P_{d}. It can be observed that the best performance in terms of P_{d} has been achieved for $k = 1$, which means that the simplest version of the kNN algorithm works best. In the case of assumed $\bar{P}_{\mathrm{fa}} = 10\%$ in the ED algorithm, kNN improves SS performance for lower values in the considered SNR range. For SNR values higher than -15 dB, the kNN results are worse than the ED results.

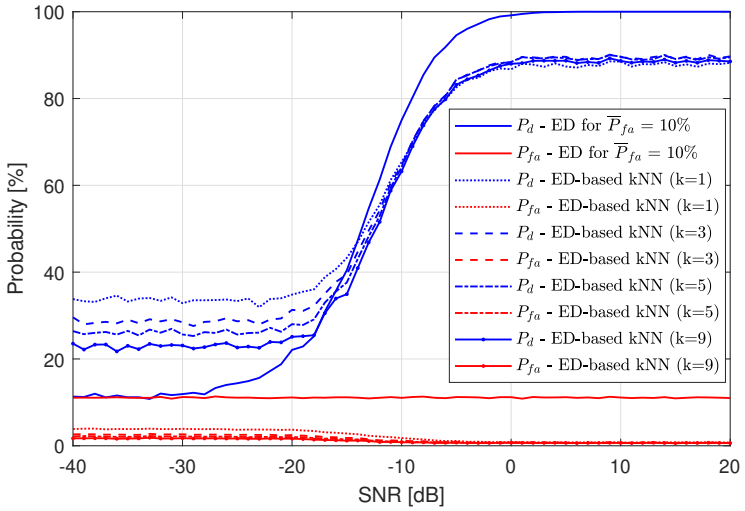


FIGURE 2.7: Resulting probability of detection P_{d} and probability of false alarm P_{fa} of the Energy Detection-based k-Nearest Neighbors method for $\bar{P}_{\mathrm{fa}} = 10\%$.

For the RF algorithm applied, the results are the best when one tree is used.

Figure 2.8 presents the RF application (after the first ED stage) performance compared to single-stage ED for $\bar{P}_{\text{fa}} = 10\%$. As in the case of using the kNN algorithm, RF also improves the SS probability for low SNR values but performs worse than just ED for high SNRs. The low performance for high SNR is caused by P_{fa}, which is the cause of ML algorithm fallibility. For high SNRs, all RBs wrongly recognized as busy by ED are gathered around occupied RBs. Since ML is trained to recognize grouped RBs marked as busy by ED as more probable to contain a signal, decreasing P_{fa} is the cause of the increasing number of detection errors. The higher the value of P_{fa}, the lower the maximum value of P_{d}. In Figures 2.7 and 2.8, one can see that the maximum achieved value of P_{d} by both ML algorithms is approximately equal to 90%, which means that the maximum P_{d} approximately equals $100\% - \bar{P}_{\text{fa}}$.

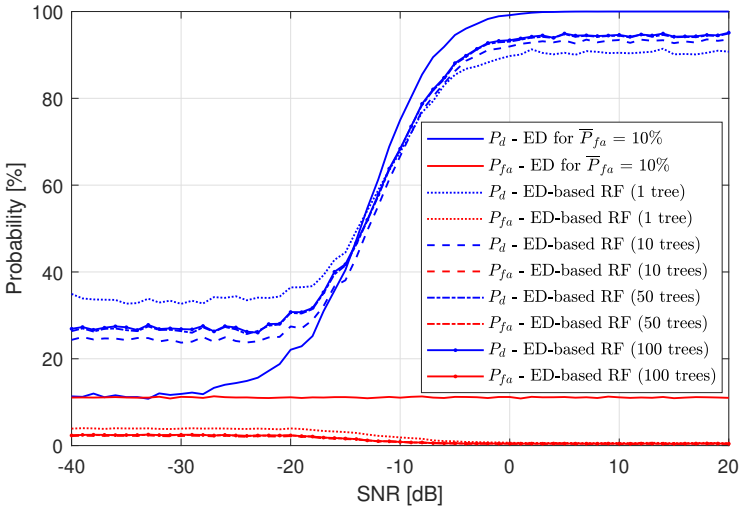


FIGURE 2.8: Resulting probability of detection P_{d} and probability of false alarm P_{fa} of the Energy Detection-based Random Forest method for $\bar{P}_{\text{fa}} = 10\%$.

With the second feature set used in the proposed detection algorithm, energy values calculated for separate RBs and all other features calculated based on them have been used directly in ML. Figure 2.9 shows the results of the kNN algorithm (again applied as the second stage of RB detection) compared to single-stage ED results with $\bar{P}_{\text{fa}} = 10\%$. It should be noted that P_{fa} does not affect the ML results in this case. Again, the best results have been obtained for $k = 1$. This time, the application of ML achieves a detection probability equal to 100% for high SNR values. In the whole SNR range, applied ML improves the performance of ED.

The same analysis was performed on the RF algorithm. Again, one tree is the best choice for the SS scenario. The results are presented in Figure 2.10.

In order to compare the best results of the ML application to the considered SS scenario, Figure 2.11 is presented. Here, ED results for the assumed $\bar{P}_{\text{fa}} = 10\%$ are compared with the kNN and RF algorithm using the ED-based feature set (red plots) and with kNN and RF using the EV-based feature set (green plots). The observation can be made that ML algorithms improve the SS performance in terms of P_{d} approximately the same in all considered cases for low SNR values (below -13 dB. However, the difference is significant for high SNR values and shows the superiority of EV-based ML over ED hard-decision-based ML.

Similar figures have been obtained for lower values of $\bar{P}_{\text{fa}}$ used in ED, namely $\bar{P}_{\text{fa}} = 2\%$ (Figure 2.12a) and $\bar{P}_{\text{fa}} = 0.5\%$ (Figure 2.12b). For smaller $\bar{P}_{\text{fa}}$, it is visible

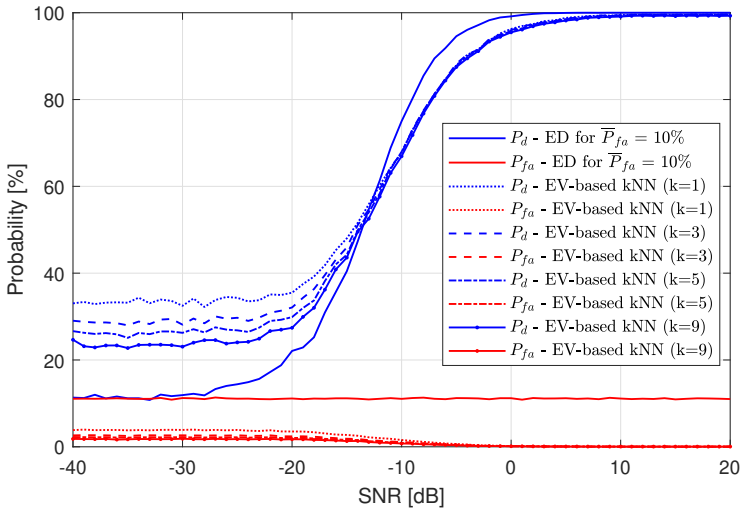


FIGURE 2.9: Resulting probability of detection P_{d} and probability of false alarm P_{fa} of the Energy Vector-based k-Nearest Neighbors method for $\bar{P}_{\mathrm{fa}} = 10\%$.

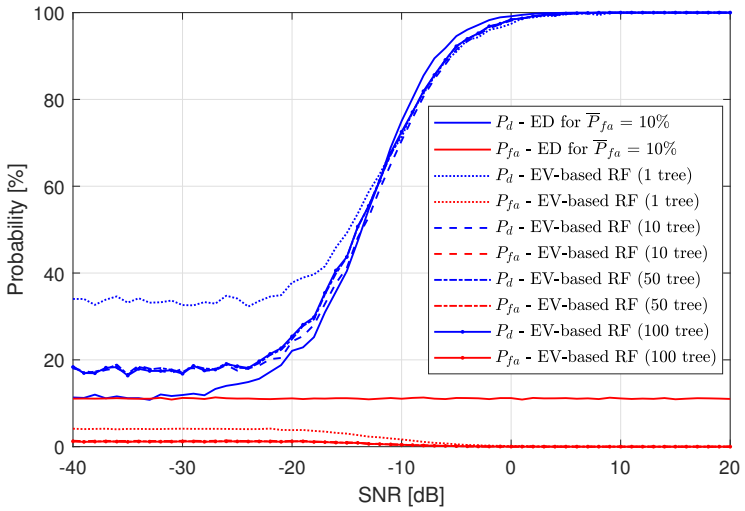


FIGURE 2.10: Resulting probability of detection P_{d} and probability of false alarm P_{fa} of the Energy Vector-based Random Forest method for $\bar{P}_{\mathrm{fa}} = 10\%$.

that the results for ED-based ML and EV-based ML become similar, although EV-based ML performance remains better. For ED-based ML, the impact of $\bar{P}_{\mathrm{fa}}$ still appears. In Figure 2.12a, the maximum $\bar{P}_{\mathrm{d}}$ value of the red plots is close to 97%, and, in Figure 2.12b P_{d}, the results are similar. The results of EV-based ML all reach 100%.

Having compared the results, one can conclude that EV-based ML methods applied in SS, in the examined scenario, perform better than ED-hard-decision-based ML methods, so there is no reason to use ED hard decisions in the feature set. Although using energy directly in ML has significant advantages, it also has disadvantages. For example, storing and processing energy values requires more memory than storing binary numbers, as in ED-decision-based ML. For EVs, the more accurate the energy values, the more precise the results. ED can be implemented using simple analog components with a comparator to compare the measured energy with a given voltage set as a threshold. An analog-to-digital converter is needed at the

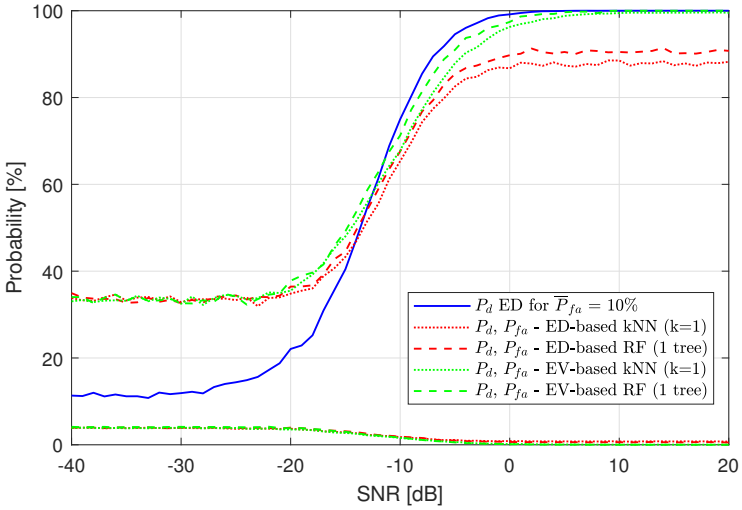
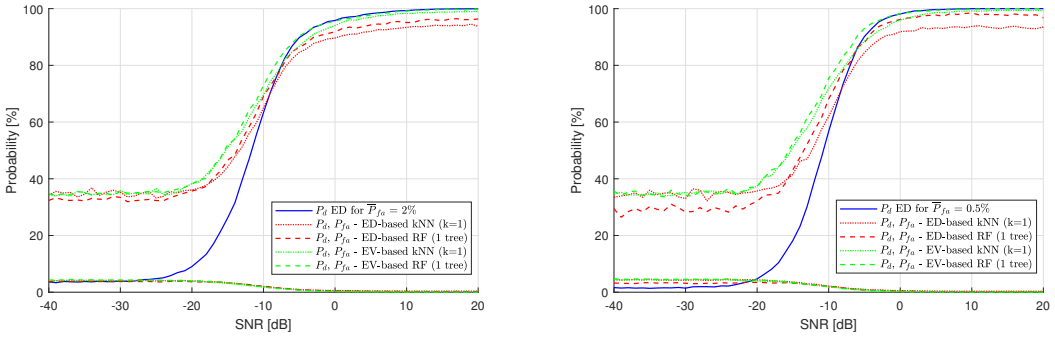


FIGURE 2.11: Probability of detection P_d comparison of the Energy Detection-based and Energy Vector-based k-Nearest Neighbors and Random Forest methods for $\bar{P}_\mathrm{fa} = 10\%$.



(A) $\bar{P}_\mathrm{fa} = 2\%$ (B) $\bar{P}_\mathrm{fa} = 0.5\%$

FIGURE 2.12: Probability of detection P_d comparison of the Energy Detection-based and Energy Vector-based k-Nearest Neighbors and Random Forest methods for different assumed $\bar{P}_\mathrm{fa}$.

output of a similar EV setup, which would also introduce a quantization error to the energy value.

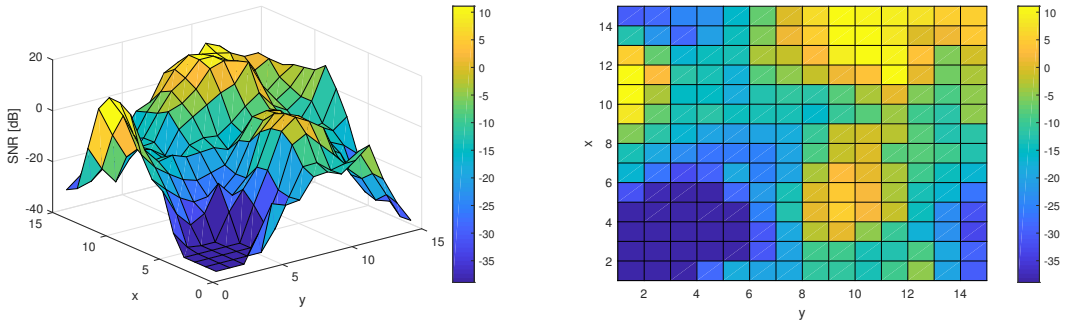
Finally, note that in the case of ED-based ML, the results may be poorer (in terms of the lower probability of detection) for high SNR values than for the ED algorithm applied alone. This could be solved by adaptively using the ML stage. In that case, ML could be used only for low SNR values; the ED performance should be sufficient for high SNRs. This requires the implementation of some adaptation algorithm or components but also reduces algorithm complexity for high SNRs. Hence, it shortens the processing time.

Table 2.2 compares the advantages and disadvantages of using ML based on ED- and EV dataset features respectively.

In the simulation experiment, a shadowing channel model was generated to check this method. Figure 2.13 shows the SNR values for the generated example of a shadowing channel characteristic in different locations. It consists of values ranging from around -40 dB to 10 dB, which covers the whole range of SNR values for which P_d

TABLE 2.2: Comparison of Energy Detection-based Machine Learning and Energy Vector-based Machine Learning.

Energy Detection	**Energy Vectors**
Advantages	
Can be used adaptively—for low SNR Machine Learning is also used, for high SNR just Energy Detection results are sufficient.	Can be used for every SNR value—results are always better or close to Energy Detection results.
Requires much less memory.	Does not require knowledge of noise or noise estimation.
Disadvantages	
The chosen Energy Detection threshold has a big impact on detection probability.	High computational complexity—Machine Learning is used in all transmission conditions.
Knowledge on noise is needed.	Memorization of real numbers required.



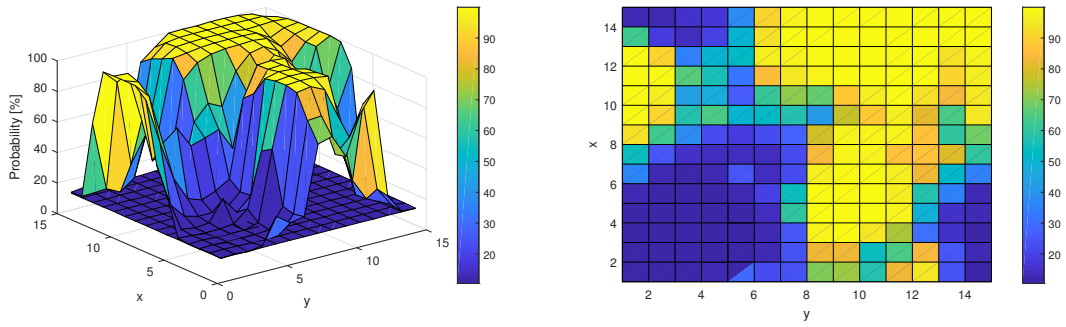
(A) SNR 3D surface plot. (B) SNR 2D plot.

FIGURE 2.13: SNR values resulting from the shadowing effect in the considered area.

ranges from its minimum value to 100%. For this simulation experiment, 5G signals resulting in RBs occupation data of the same parameters as used in the previous experiments have been generated and collected at every point of the 15-by-15 location area. Then, the ED-based sensing and ML methods were applied separately at every one of the 225 location points to evaluate the applied ML algorithms aiming at improving the spectrum detection at different locations.

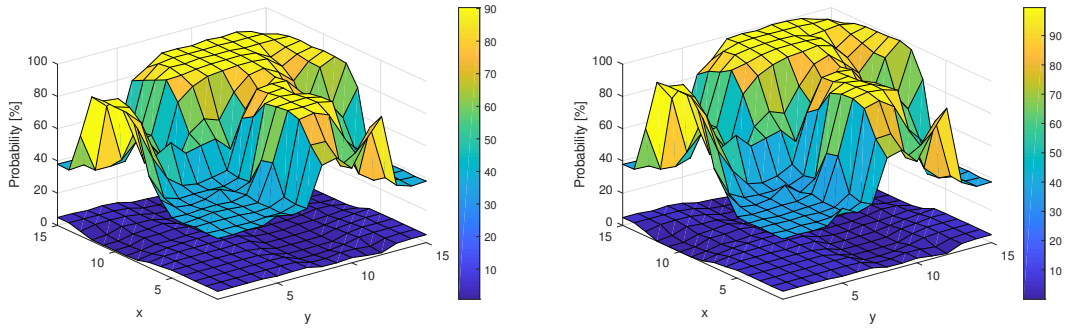
Probabilities P_{d} and P_{fa} of the ED method applied alone (basic ED hard decision method) are presented in Figure 2.14. These results have been obtained for $\bar{P}_{\mathrm{fa}} = 10\%$. As expected, P_{d} values range from 10% to 100%.

Moreover, algorithms kNN and RF have also been examined to compare ML results when ED decisions and measured EVs are used as dataset features. Figure 2.15 shows P_{d} and P_{fa} surfaces for ED-based input (Figure 2.15a) and EV-based one (Figure 2.15b) for the applied kNN algorithm with parameter $k = 1$. The probability of detection P_{d} for kNN based on ED hard-decision input ranges from 10% to 90%, while P_{d} for kNN based on EV input ranges from around 37% to 100%,



(A) P_{d} and P_{fa} 3D surface plot (B) P_{d} 2D plot

FIGURE 2.14: P_{d} and P_{fa} for different locations in case of basic hard-decision ED.



(A) P_{d} and P_{fa} surfaces for Energy Detection-based k-Nearest Neighbors

(B) P_{d} and P_{fa} surfaces for Energy Vector-based k-Nearest Neighbors

FIGURE 2.15: P_{d} and P_{fa} for the k-Nearest Neighbors method applied in different locations.

so the improvement resulting from using EVs as dataset features is visible. This is even more clear when P_{d} is plotted versus SNR, as in Figure 2.16.

The results presented in Figure 2.16 have been obtained by assuming previous knowledge of the SNR values for every location point. The results are similar to those achieved without shadowing effect (assuming that the SNR variations of slow-fading can be corrected by adaptive gain control) or even slightly better, as, for low SNR values, P_{d} is equal to 37%. In contrast, P_{d} was equal to approximately 34% in the previous experiments. Similar results have been obtained for the RF algorithm using one decision tree. Figure 2.17 shows P_{d} and P_{fa} for this algorithm. Moreover, Figure 2.18 presents P_{d} and P_{fa} vs. SNR in the shadowing scenario considered. In this case, ED improvement can also be observed with the applied RF method, especially for low SNR values.

Finally, the comparison was made between the author's proposed kNN and RF algorithms and the algorithms most commonly found in the literature, namely SVM and NB algorithms. Figure 2.19a presents results of ED-based algorithms: the kNN, the RF, the SVM classifier (with radial basis function kernel), and the Gaussian NB. Similarly, Figure 2.19b shows results for the same ML methods but EV-based. It can be observed that SVM and Gaussian NB methods do not improve detection performance very much for low SNR values. The exception is the EV-based Gaussian NB method, which significantly improves P_{d} for low SNR values but unfortunately also introduces a large increase in P_{fa}.

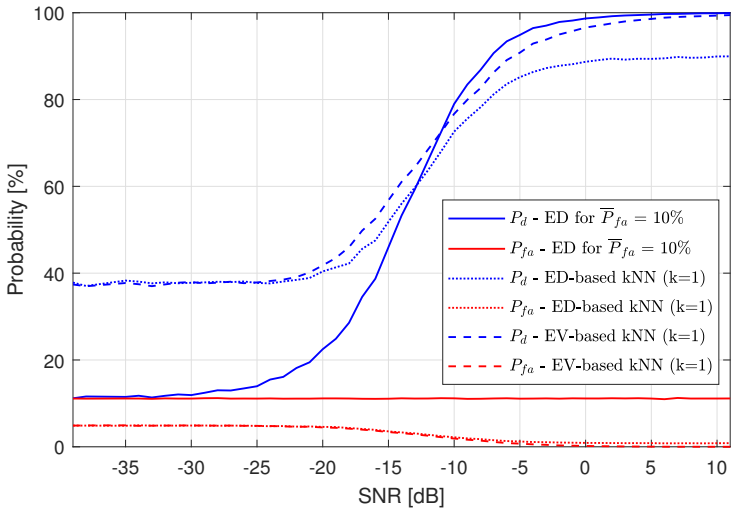
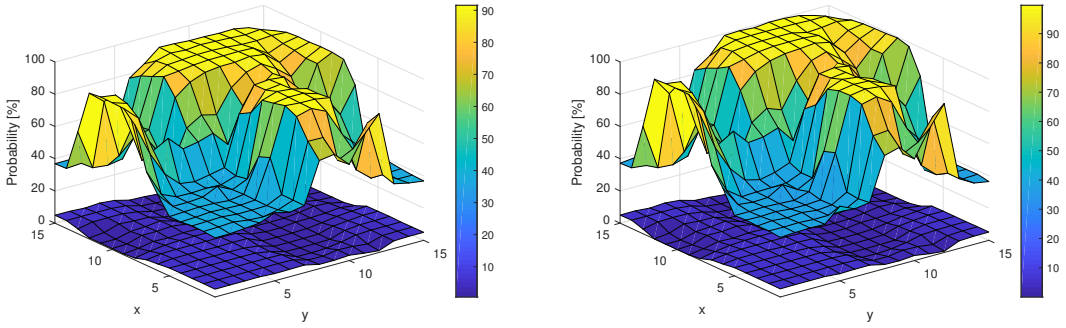


FIGURE 2.16: Resulting probabilities P_{d} and P_{fa} of Energy Detection-based k-Nearest Neighbors compared with Energy Vector-based k-Nearest Neighbors for $\bar{P}_{\mathrm{fa}} = 10\%$ for different SNR values with a shadowing channel.



(A) P_{d} and P_{fa} surfaces for Energy-Detection-based Random Forest

(B) P_{d} and P_{fa} surfaces for Energy-Vector-based Random Forest

FIGURE 2.17: Probabilities P_{d} and P_{fa} in different locations for the applied Random Forest method.

2.4 Chapter summary

In this chapter, ML algorithms have been considered to increase the quality of energy-measurement-based SS in the presence of 5G downlink transmission. The idea behind applying ML algorithms was to take advantage of the dependencies existing in the RBs occupation in the time and frequency domain. By learning some signal properties and traffic intensity correlation in time and frequency, it is possible to increase the detection probability of the mentioned 5G PU transmission. Both the ED hard decisions and the EVs, obtained at the first phase of the SS method, have been considered as ML input dataset features. Moreover, several classifiers have been considered as the second phase of SS, namely kNN, RF, SVM, and Gaussian NB.

It has been shown that, for high SNR values, kNN and RF based on EVs as the input dataset features perform better (particularly in terms of the probability of detection P_{d}) than those based on ED hard decisions. Furthermore, when compared

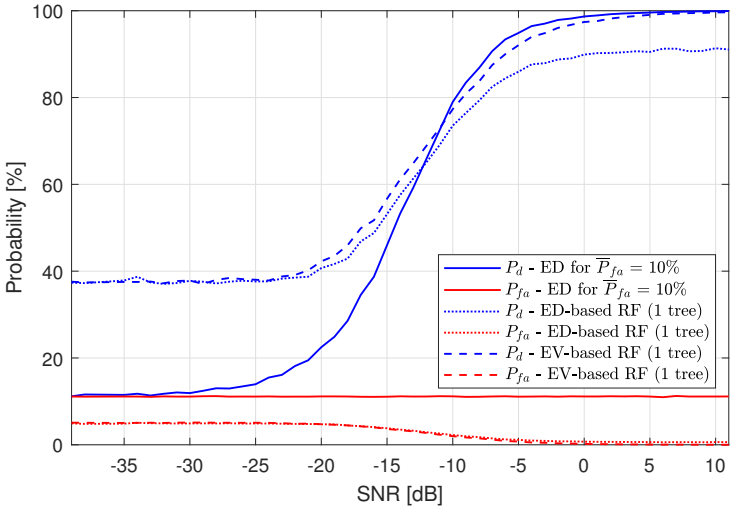
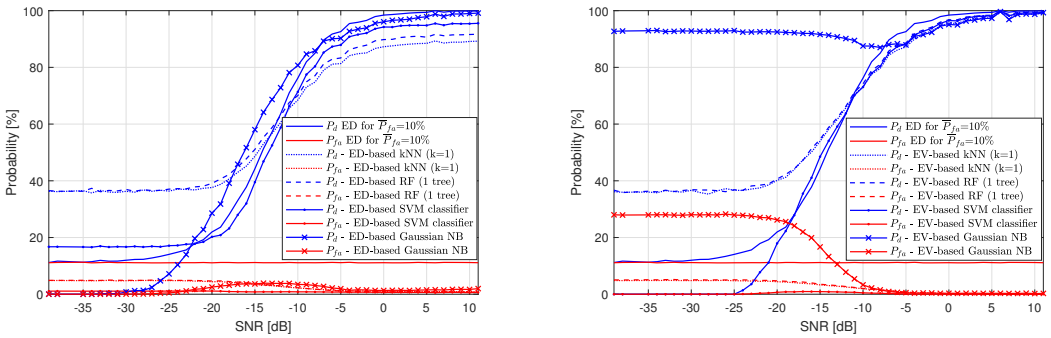


FIGURE 2.18: Resulting probabilities P_{d} and P_{fa} of applied Energy Detection-based Random Forest compared with Energy Vector-based Random Forest for $\bar{P}_{\mathrm{fa}} = 10\%$ for different SNR values with a shadowing channel.



(A) P_{d} and P_{fa} surfaces for Energy Detection-based Machine Learning algorithms

(B) P_{d} and P_{fa} surfaces for Energy Vector-based Machine Learning algorithms

FIGURE 2.19: Probabilities P_{d} and P_{fa} in different locations for the applied k-Nearest Neighbors, Random Forest, Gaussian Naive Bayes, and Support Vector Machine classifier methods.

with the standard ED method, application of ML (both kNN and RF) increases P_{d} in lower SNRs region.

The proposed ML algorithms have also been implemented in different locations of the considered area to suppress the need for multiple SNR estimations when choosing the training dataset features. Taking advantage of the spatial characteristics of the shadowing effect, further improvement in the performance of the proposed algorithm can be observed; that is, the probability of detection P_{d} increases with respect to standard methods. This eliminates the need for continuous noise power estimation.

Chapter 3

Autonomous Deep Learning-Based Spectrum Sensing

This chapter extends the analysis of the previous chapter of traditional ML algorithms by considering the application of neural networks to improve SS, and SP, and fading level estimation based on calculation of energy values. In the first part of this chapter, the author examines different DL methods, namely Neural Network (NN), Recurrent Neural Network (RNN), and Convolutional Neural Network (CNN) algorithms. Their application and performance in SS and SP are compared and advantages and disadvantages are noted. A simple baseline method, used for results comparison of SP is proposed. In the second part of this chapter the author focuses on one chosen DL method, namely CNN applied for both SS and SP. The addition introduced in the second part of the chapter is a fading level estimation at the receiver, also based on a CNN algorithm designed by the author. Moreover, optimization of the fading level threshold for the spectrum occupancy decision making is discussed. Its impact on the SS and SP results is analyzed.

The author's original contribution and research results are structured in this chapter as follows. First, in Section 3.1, the concept of DL is explained, and as examples of DL algorithms, NN, RNN, and CNN algorithms are discussed. In Section 3.2 one of the two DL-based approaches to SS and SP is described. It focuses on the design and evaluation of NN, RNN, and CNN applied for SS and SP. In Section 3.3, the author of this thesis presents a new algorithm applying CNNs that not only performs SS and SP but also estimates the current and predicts the future impact of the fast-fading channel on the received PU signal. As a result, this novel algorithm further improves SS and SP accuracy. The key findings of this chapter are summarized in Section 3.4.

3.1 The Concept of Deep Learning for Spectrum Occupancy Detection and Prediction

DL is a category of ML algorithms, which refers to NN algorithms with multiple neural layers and complex structure. NNs are considered deep, when they have multiple hidden layers of neurons. The hidden layers are the layers that are neither used as an input layer nor output layer of the NN. Therefore, the NN in order to be considered a deep NN must have a total number of layers at least equal 4. Each layer consists of elementary, computational units called neurons. A neuron

is a non-linear function that calculates output value from one, or multiple inputs. The operation of neurons is as follows: each input value is multiplied by their corresponding weight and then summed. At this point the result is a simple linear combination of the inputs. In order to introduce non-linearity an activation function is introduced to the neuron. The results of the linear combination passes trough the activation function and the final output value of the neuron is obtained. The activation functions can be tailored to the considered problem. The most popular are sigmoid type of functions. A NN consists of multiple layers in which there are multiple neurons. In the training process, the weights of the neurons are the values that are gradually adjusted in order to achieve a results at the output of NN as close to the expected value as possible. It is easy to notice that since an NN, especially a deep NN consists of many neurons, the training process that requires adjusting all of the weights is very computationally demanding. However, thanks to the complex structure the DL algorithms are able to solve very complex problems, recognize intricate patterns, and generate detailed and very specific data.

DL algorithms are known for finding complicated dependencies in input data [57]. In the prediction problem, it is crucial to recognize any patterns that may occur in the receiving signal, and DL algorithms should be a good choice. In this chapter, the considered ML algorithms involve a NN with dense layers, a RNN structure with LSTM layers, and a CNN. The RNNs are usually used in language and audio signal processing to predict sequences [43]. Their particular usefulness in this field is because basic elements of RNN layers called cells feedback their output as additional input information, which makes it possible for RNNs to notice intricate patterns occurring in input data in time. [151] is a work considering RNNs for spectrum state prediction in the form of a one-dimensional time series.

CNNs are broadly used for image recognition, processing, and classification [129], while in this chapter, the novelty is to consider them for spectrum sensing and prediction using two-dimensional images formed of energy values and additional features in time and frequency dimensions. As far as it is known to author, the CNN algorithm for spectrum occupancy prediction is usually used as a sensing or prediction tool for two-dimensional data in cooperative sensing [196, 94, 88], where data collected from each of sensing SUs is merged into a set of input information for CNN algorithm. In [122] CNN is also applied to predict the type, form, and several transmitting users in a given frequency band. In the proposed solution, data for detection and prediction is a one-dimensional time series. Time series has also been used in [5], where time-series modeling approaches have been compared with recurrent neural network SS performances.

In [122] also NN, RNN and CNN are applied to predict the type, form, and number of transmitting users in a frequency band, but data used for detection and prediction is a one-dimensional time-series data.

The long-term prediction based on spatial-spectral-temporal data has been addressed in [149], where a hybrid convolutional long short-term memory has been proposed for future spectrum state prediction. Another interesting hybrid DL approach has been presented in [137], which also exploits CNN and LSTM combination. Here, as input data, a set of IQ samples is considered for each moment in time.

3.2 New Deep Learning Algorithms for Spectrum Sensing and Prediction

Three DL algorithms have been implemented for SS and SP. First, the NN algorithm has been implemented as an example of the simplest algorithm. The second algorithm is an RNN algorithm. The last algorithm implemented is a CNN.

All of the proposed algorithms are supervised classification algorithms. Each of them is trying to establish an occupancy status of current or future RBs, indicating a binary classification problem. The proposed NN-based method is the simplest and requires the least calculations but can perform only a single RB classification. Based on the current input data, the NN classifies one current or one of the future RBs' as free or occupied. On the other hand, the RNN, as a more complicated method, can detect and predict several following RBs but only for a single frequency range. The CNN can perform the most complex calculations and classify multiple RBs in time and frequency.

Each proposed algorithm receives slightly different input data based on RB energy values. A single input dataset for the NN algorithm consists of four values that characterize a single RB: frequency index (values from range 0 – 49), time slot index (values from range 0 – 79), energy value for the considered RB, and sum of energies of neighboring RBs. Each element of an input sequence of RNN consists of 3 values: time slot index, energy value, and the sum of neighboring RBs' energy values. In the case of the CNN algorithm, the input data is constructed as a 2D image of which first dimension is frequency, the second is time, and pixels contain RB energy values. Input images have three layers, similar to color images with three RGB components. The first layer consists of the RBs' energy values; as mentioned earlier, the second layer contains frequency index values, and the third layer contains time slot indexes.

The example of a CNN input data image is presented in Figure 3.1, where three layers are combined into one CNN input picture.

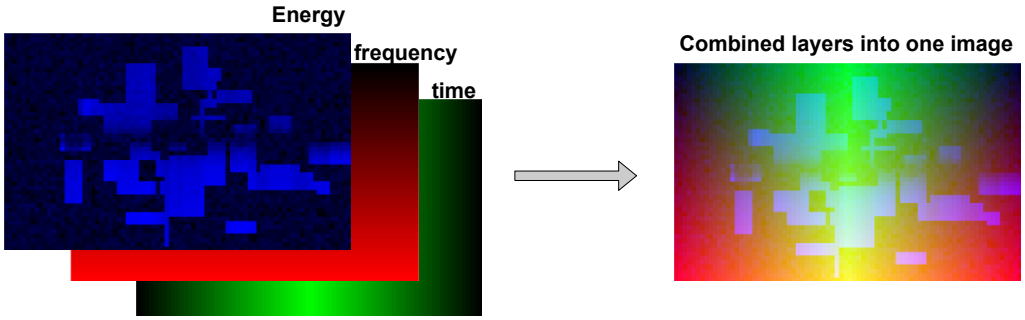


FIGURE 3.1: The CNN input data. The input image consists of three layers: energy values per RB, frequency indicator, and time indicator. The layers can be treated as RGB components.

3.2.1 Proposed DL algorithms and data sets

Transmission detection is the decision regarding the present state of spectrum occupation, i.e., at time moment t, while SU may be interested in predicting the future states. For the prediction, the main issue is to decide on a future spectrum

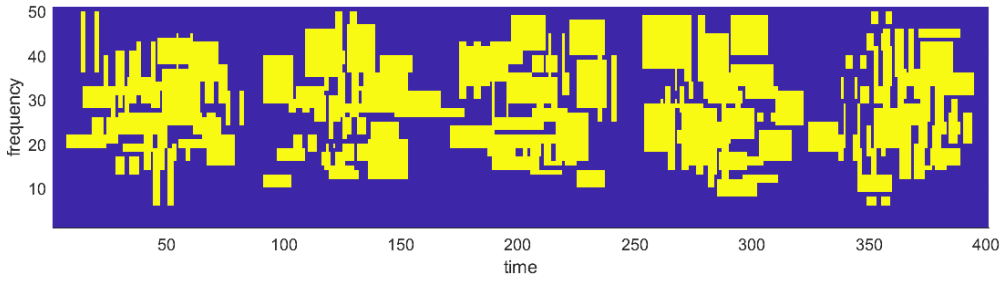


FIGURE 3.2: First dataset - time- and frequency-correlated RBs occupancy.

state, i.e., at the one following or several following time moments in the time interval $[t, t+\tau]$, where $\tau > 0$. This decision, however, is based on the current signal data, i.e., collected at time moment t.

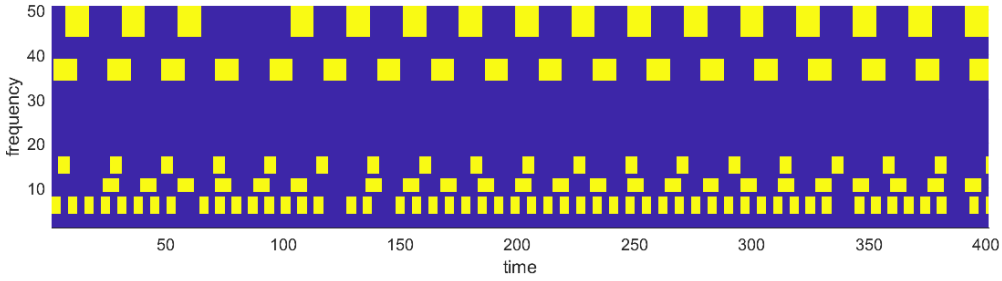
Although multiple well-known spectrum detection methods exist, ML algorithms have proven useful in the SS area. Future spectrum state prediction is another area in which machine learning performs well, thanks to its adaptability and ability to find patterns in input data.

In the considered scenario, PU is a 5G Base Station (BS). SU detects this signal and tries to decide (by employing ML techniques) on RBs occupancy in the current moment and in the subsequent time slots. To this end, SU collects the signal samples and calculates the energy for every RB in every first OFDM symbol in a given time slot. With this information, SU tries to decide whether a slot is occupied. By calculating the energy in a single OFDM symbol, SU has time to decide and transmit in the remaining part of the time interval. However, it would be beneficial to simultaneously gain knowledge of the occupancy of the same RBs in the future time slots to prepare for longer transmission. The ML algorithms proposed in the following aim at the RBs occupancy decision for the current and six next time slots. The ML input data is calculated based on the energy values per RB. The ML algorithms are trained separately for different SNR values.

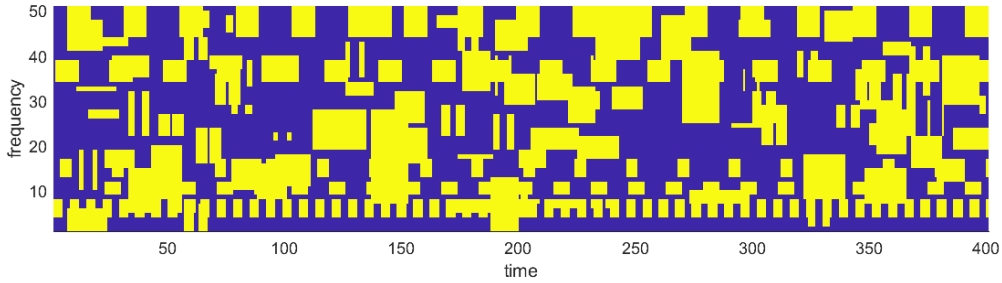
Two cases of PU signals and resulting RBs occupancy are considered. The first signal is a symbolic representation of the daily fluctuations in traffic intensity typical to radio communication systems. Figure 3.2 presents a signal for 400 slots as RBs in frequency and time. The yellow areas indicate occupied RBs, while the blue ones - free (unoccupied) RBs. One can see that the spectrum occupancy is correlated with time. Moreover, RBs occupancy in frequency is not uniform, which is dictated by the fact that the channel may prevent effective transmission on specific frequencies to some users. In the example presented in Figure 3.2, it is assumed that the PU signal is more likely to appear in the middle frequencies and less probable in the marginal frequencies.

The second case considered concerns a system in which data of the IoT devices (e.g., telemetric sensors) are transmitted in the form of short packages and with high periodicity. In such a case, the PU signal occurs in every cycle with high probability, although devices hold back the transmission from time to time. Figure 3.3a presents PU RBs occupancy in such a case. In addition to the IoT-device signals with a deterministic RBs occupancy, another PU signal presence is considered, characterized by a random RB occupancy and specific correlation over time. The RB occupation of two such PUs signals is presented in Figure 3.3b.

As mentioned before, three DL algorithms have been implemented. Figure 3.4



(A) Deterministic RBs occupancy by IoT-devices signals.



(B) Random RBs occupancy by IoT-devices and random time-correlated PU signals.

FIGURE 3.3: Second dataset - RB occupancy by IoT devices and time-correlated PU transmission.

shows the structure of the NN model applied as the first. It consists of three dense layers, preceded by a Softmax layer [20], which converts received data into probability values. The first two dense layers consist of 10 neurons. The last dense consists of two neurons.

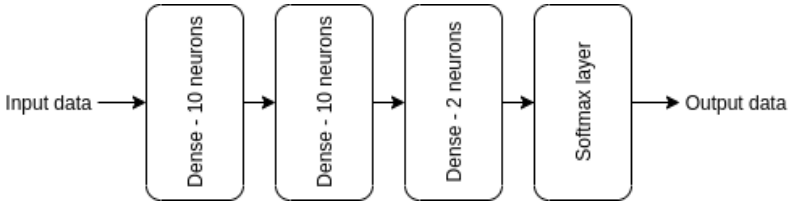


FIGURE 3.4: NN algorithm model

The NN is used here for classification problems, so as a loss function, Sparse Categorical Cross Entropy [57] has been used. The output data consists of two probabilities – the probability of a considered RB belonging to the 'occupied' category and the probability of belonging to the category 'free' (unoccupied). Those two probability values sum up to one. To achieve a classification result for current RB occupancy detection or for further RB occupancy prediction, separate NNs must be trained. In the experiments shown in this section, detection and prediction from the first to the sixth next time slot are performed. This requires creation of seven NNs, one for each application. The training process has used the Stochastic Gradient Descent optimizer [155].

The next proposed RNN model consists of three LSTM layers [66]. A dropout of 0.5, 0.3, and 0.2 is applied after each LSTM layer, respectively, to prevent overfitting. The last layer is a time-distributed dense layer consisting of 4 neurons. A sequence consisting of 100 feature sets is provided as input. The output consists of

a sequence of probabilities. The first probability value concerns the probability of the current RB being occupied. The following three probabilities are used to predict the following RBs occupancy for the same frequency but in future time slots. Since the RNN accepts one-dimensional data as the input, there is a need to train separate RNNs for each frequency separately. The Adam optimizer [155] has been used, with a learning rate of 0.001. As a loss function, binary cross-entropy has been implemented. Figure 3.5 presents the RNN model.

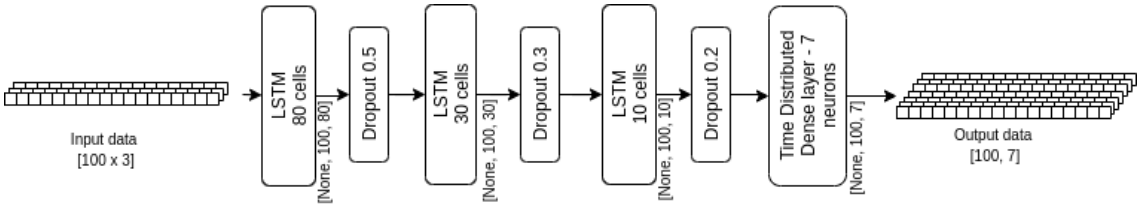


FIGURE 3.5: RNN algorithm model

The last algorithm implemented is based on CNN. Figure 3.6 shows the model of the proposed algorithm. This method accepts a spectrogram-like image of energy values and other features per RB as input. The input data is padded with zero values on the top, bottom, and right sides of the spectrogram. This network's output results are two two-dimensional layers of 50-by-7 pixels. This means the results are detection and prediction for each of the 50 frequencies for the current and next 7 time slots. One pixel represents one RB.

The created CNN model consists of four convolutional layers. The first one has 8 kernels (filters), 9-by-80 pixels in size. This layer returns an image of the same size as the input. The second layer uses 16 kernels of size 5-by-50, and the third one uses 32 kernels of size 3-by-25. The growing number of kernels in each layer ensures better recognition of any more abstract features of input data. The output layer has only two kernels, each for every RB's occupancy category - free or occupied. The filters are 1-by-27 to ensure a proper output image size. Each layer uses the rectifier function (ReLU) as an activation function, except for the last layer, which uses the Softmax function. The Adam algorithm is implemented as an optimizer with a learning rate equal to 0.0001. Because the output consists of two categorical probabilities, the Sparse Categorical Cross entropy is used as a loss function.

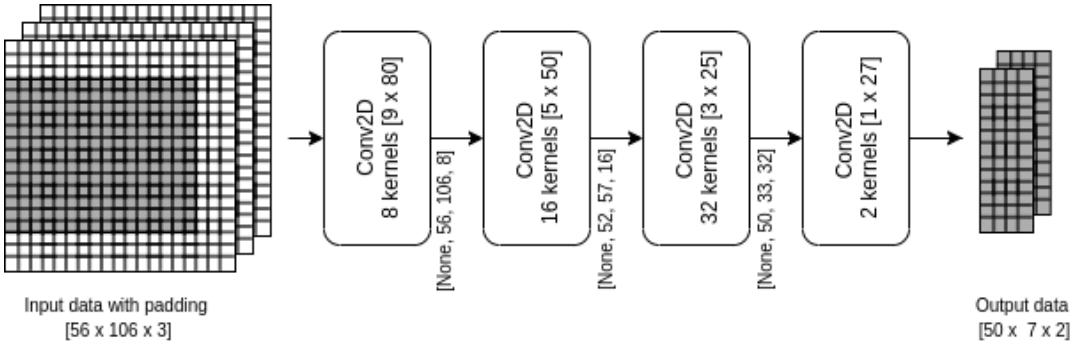


FIGURE 3.6: CNN algorithm model

Additionally, a baseline algorithm called PA for prediction is proposed to evaluate whether DL algorithms introduce any improvement in prediction. This primitive method uses detection results of DL. It assumes that all RBs for a given frequency

in every future time slot will have the same occupancy state as those that have just been detected. This simple (primitive) prediction method can provide good results if the spectrum is occupied most of the time or in continuous time groups of RB. The latter case is valid for both of the considered datasets.

3.2.2 Simulation Experiment

Simulation Setup

Below, a particular scenario is considered in which an unlicensed user SU aims to detect and predict 5G transmission activity of the licensed 5G users PUs in time and frequency. The PU's transmission occurs over a 10 MHz band comprising 50 RBs. Every RB consists of 7 OFDM symbols transmitted in a 0.5 ms slot over 12 subcarriers. The energy values (which are used for DL input) are calculated only for the first OFDM symbol in each RB. Two cases of PU transmission have been simulated as described in 3.2.1. The PU's signal received by SU is distorted by the fading channel modeled according to 3GPP *Extended Pedestrian A model* [153] model for which Doppler frequency is equal to 0 Hz.

In the experiments, SU trains its DL models specialized for a given SNR value, similarly to the algorithms presented in Chapter 2. Experiments have been conducted for different SNR values to test the performance of the algorithms on SS and SP.

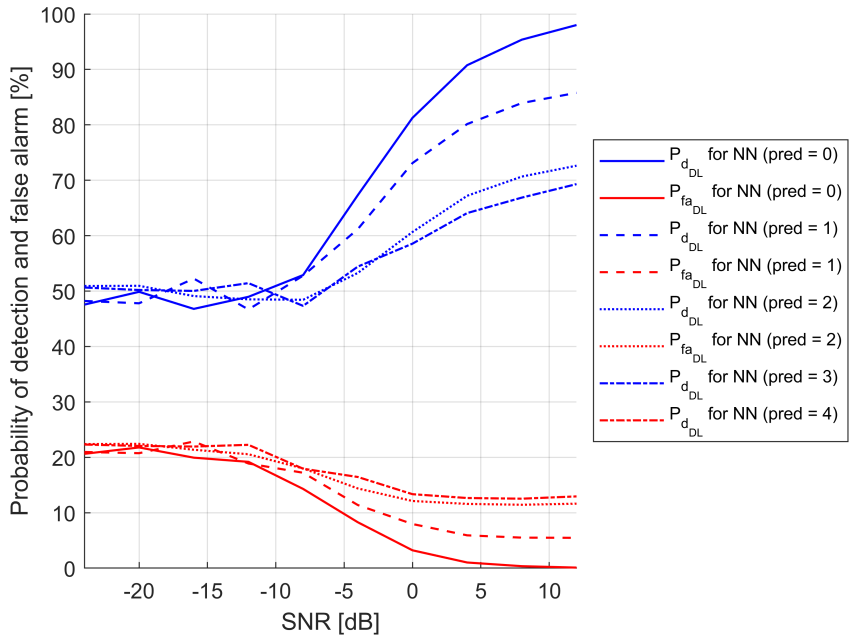
For all of the considered DL algorithms and their corresponding PA results, plots of P_{d} and P_{fa} for a range of SNR values have been obtained. To facilitate the interpretation of the results, the bar charts have been drawn of the probabilities P_{d} and P_{fa} for each of the prediction steps. These charts are created for the highest SNR value, with the most significant difference between DL and PA. Additionally, an overall measure of improvement - a total difference between both probabilities has been determined according to the formula:

$$D_{\mathrm{total}} = P_{\mathrm{d_{DL}}} - P_{\mathrm{d_{PA}}} + P_{\mathrm{fa_{PA}}} - P_{\mathrm{fa_{DL}}}, \tag{3.1}$$

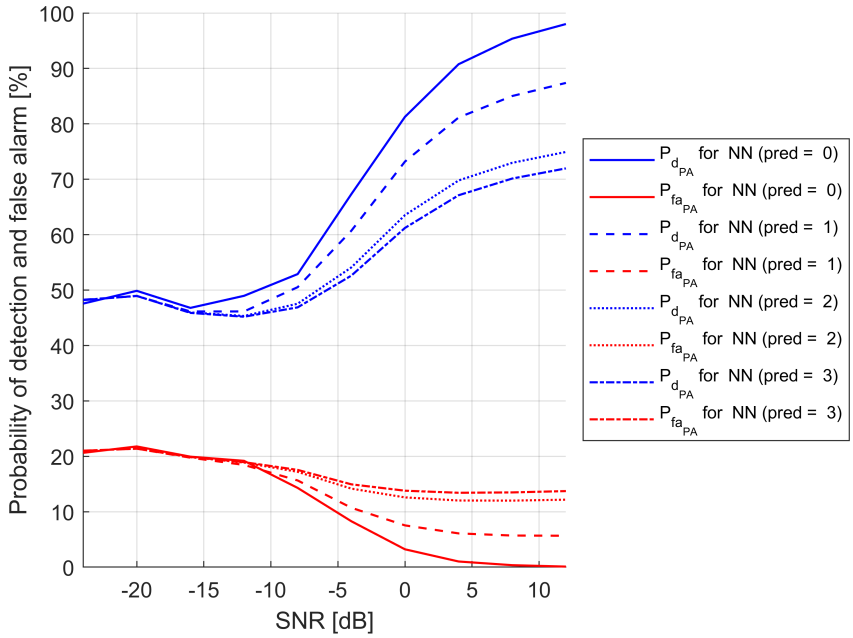
where D_{total} is a total evaluation measure, $P_{\mathrm{d_{DL}}}$, $P_{\mathrm{fa_{DL}}}$, $P_{\mathrm{d_{PA}}}$, $P_{\mathrm{fa_{PA}}}$ are probabilities of detection and false alarm for a DL algorithm applied to SS and SP and respectively for DL-based SS and PA-based SP.

Simulation Results

First Dataset. Results for the first dataset have been achieved by applying all of the proposed DL algorithms. Figure 3.7 shows plots of probabilities of detection and false alarm for detection and prediction for the succeeding time slots. In this and the following figures, 'pred' in a legend means the number of the succeeding time slots subject to prediction. It is also called the *prediction horizon* or *prediction step*. Figure 3.7a shows the results for the proposed NN algorithm, i.e., for NN-based SS and NN-based SP, while Figure 3.7b contains results for NN-based SS and simple (primitive) SP based on PA. It can be observed that for low SNR values, the NN algorithm can achieve a probability of detection equal to around 50% while keeping lower values of probability of false alarm around 20 %. Typical detection algorithms like the ED method usually achieve the same values of the probability of detection P_{d} and the probability of false alarm P_{fa} for low SNRs, so the results



(A) NN-based SS and SP.



(B) NN-based SS and PA-based SP.

FIGURE 3.7: Probability of detection and false alarm vs. SNR for NN-based SS and prediction for the first dataset.

achieved by NN are beneficial. However, $P_{\text{fa}_{\text{DL}}}$ and $P_{\text{fa}_{\text{PA}}}$ could still be considered high depending on the detection and prediction requirements for transmission. The results for spectrum detection and prediction for both NN-based SP and PA are the same for low SNR. This is a common rule for all results presented in this section. For low SNRs, the noise prevents time-dependent sequences from being found. Hence, there are no differences in the detection and prediction results. Because there are some dependencies on resource allocation in time, a significant difference between

$P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{fa_{DL}}}$ and between $P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$ can be observed. The results of NN-based and PA-based SP for high SNRs are similar, although PA shows a slight advantage.

The differences in the results for NN-based SS and two types of prediction (NN-based and PA-based) for SNR=12 dB are easier to spot in Figure 3.10. There, the blue bars correspond to $P_{\mathrm{d_{DL}}}$ (in the upper plot of this figure) and $P_{\mathrm{fa_{DL}}}$ (in the lower plot) when both SS and SP are based on NN. The grey bars overlapping the blue ones represent results ($P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$ respectively) obtained for NN-based SS and PA-based SP. It can be observed that both $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{fa_{DL}}}$ for all predictions are lower than $P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$. Figure 3.11 presents the overall prediction evaluation measure D_{total} as defined by equation (3.1) as well as another measure D'_{total} defined later in this section by equation (3.2).

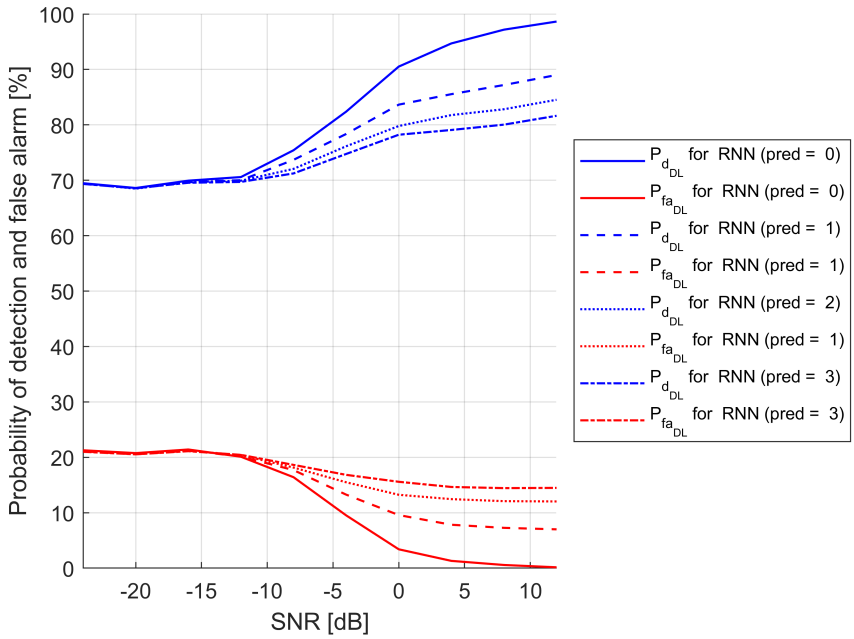
The results of RNN-based SS and SP are presented in Figure 3.8. Here, the differences between RNN-based and PA-based SP are more visible than in the case of the application of NNs. Especially $P_{\mathrm{d_{DL}}}$ is significantly higher for RNN-based prediction than $P_{\mathrm{d_{PA}}}$ (for PA-based prediction) in the next second and third slots. The gap between $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{fa_{DL}}}$ and between $P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$ for low SNRs is even more substantial than for the applied NN algorithm. Here, $P_{\mathrm{fa_{DL}}}$ and $P_{\mathrm{fa_{PA}}}$ remain at the 20% level, while $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{d_{PA}}}$ reach 70%. For high SNR values and a larger number of prediction steps, RNN-based and PA-based prediction results are also compared in more detail in Figure 3.10.

The results of the application of CNN for SS and SP are presented in Figure 3.9. It could be expected that CNN would work as well or even better than the RNN algorithm. The achieved results prove it is true. $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{d_{PA}}}$ for high SNRs are almost as high as in the case of RNN application, and $P_{\mathrm{fa_{DL}}}$ and $P_{\mathrm{fa_{PA}}}$ values are even lower. Although the $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{d_{PA}}}$ values for low SNRs are not as high as for RNN-based SS and SP, the $P_{\mathrm{fa_{DL}}}$ and $P_{\mathrm{fa_{PA}}}$ values are lower, and the differences between $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{fa_{DL}}}$ and between $P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$ remain equal to around 50%.

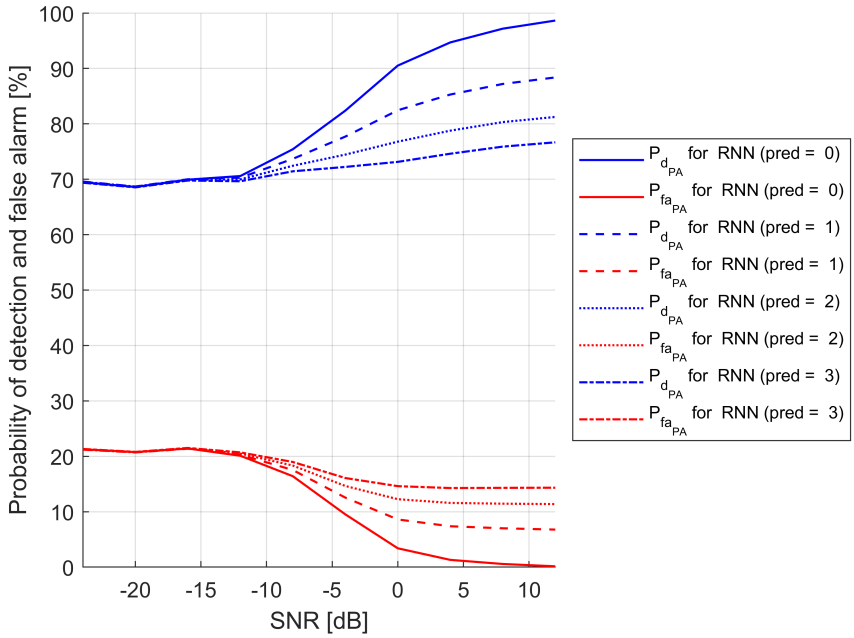
As mentioned above, all the detection and prediction results for high SNR are compared collectively to evaluate which algorithms work best on the first dataset. In Figure 3.10, one can observe $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{d_{PA}}}$ (upper plot) as well as $P_{\mathrm{fa_{DL}}}$ and $P_{\mathrm{fa_{PA}}}$ (lower plot) for each of DL algorithms and their corresponding PA (for spectrum prediction) at SNR=12 dB. Each of the DL bars has a corresponding overlapping grey PA bar. It can be observed that PA-based SP results are dependent on the DL algorithm used for spectrum detection (SS). For instance, PA-based SP is significantly worse when NN is used for spectrum detection than in the case when RNN and CNN are used for SS, although all three DL algorithms have very similar results for spectrum detection alone (prediction step equals zero). The probability of false alarm for NN-based SS and PA-based SP ($P_{\mathrm{fa_{PA}}}$) grows much faster with each prediction step. In fact, for the sixth predicted slot, the values of $P_{\mathrm{fa_{PA}}}$ and $P_{\mathrm{d_{PA}}}$ are only 15% apart.

Note that NN P_{d} results are usually a few percent worse than PA results, but since $P_{\mathrm{fa_{PA}}}$ values grow fast with each prediction step, $P_{\mathrm{fa_{NN}}}$ reach lower values, although it is still higher than P_{fa} of other DL algorithms. It is also quite clear that RNN achieves the best P_{d} results, but the best P_{fa} results belong to the CNN algorithm. Either way, both methods work comparably for the first dataset.

Figure 3.11a shows how each of the DL-based SS and SP algorithms perform compared to the same DL-based SS and PA-based SP. These results were obtained using equation (3.1). The negative values of the bars indicate that the PA-based SS



(A) RNN-based SS and SP.

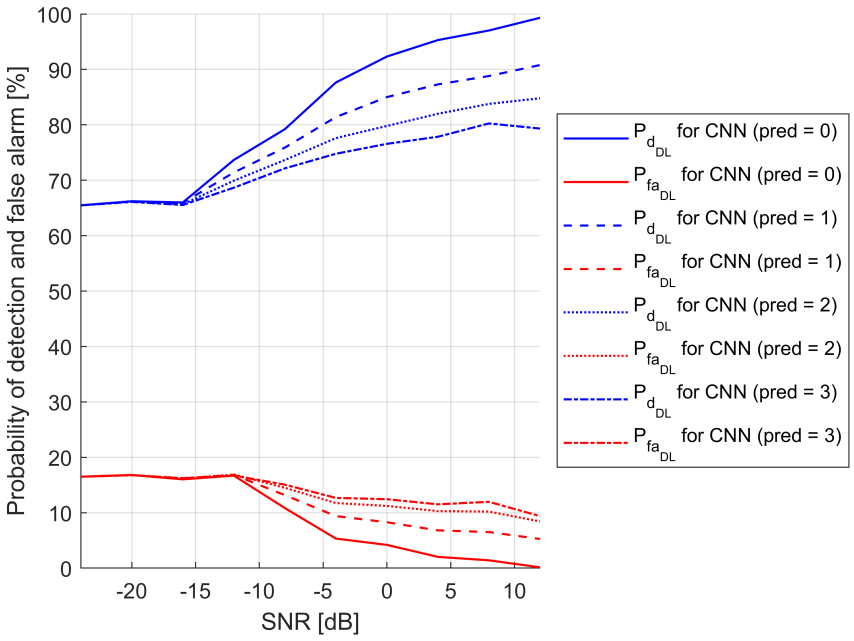


(B) RNN-based SS and PA-based SP.

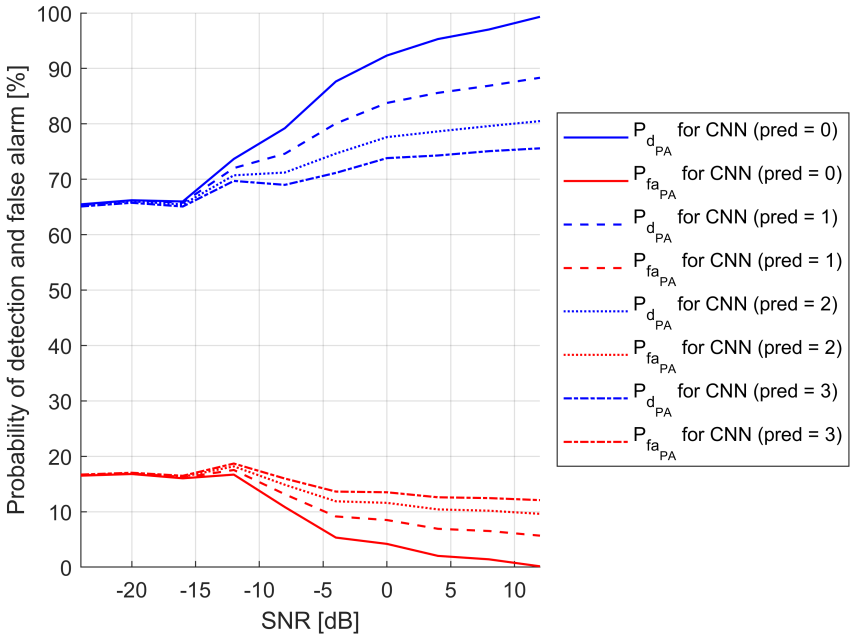
FIGURE 3.8: Probability of detection and false alarm vs. SNR for RNN-based SS and prediction for the first dataset.

is better than DL-based one for a given prediction step. Figure 3.11b also compares DL-based SP with PA-based SP, however here, the latter is always assumed to have the input from CNN-based SS. This is because out of all PA-based prediction algorithms, this one has the best performance in terms of the difference between $P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$. In other words, Figure 3.11b presents the following evaluation measure D'_{total}:

$$D'_{\mathrm{total}} = P_{\mathrm{d_{DL}}} - P'_{\mathrm{d_{PA}}} + P_{\mathrm{fa_{PA}}} - P'_{\mathrm{fa_{DL}}}, \tag{3.2}$$



(A) CNN-based SS and SP.



(B) CNN-based SS and PA-based SP.

FIGURE 3.9: Probability of detection and false alarm vs. SNR for CNN-based SS and prediction for the first dataset.

where $P'_{\mathrm{d_{PA}}}$, $P'_{\mathrm{fa_{PA}}}$ are probabilities of detection and false alarm respectively for a CNN algorithm applied to SS and PA-based SP. It is visible now that NN-based SS and SP is the worst-performing algorithm when D'_{total} metric is considered.

Second Dataset. As a second dataset, signals from IoT devices have been considered. Unlike the results of the first dataset, this time, the gap between $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{fa_{DL}}}$ and between $P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$ for low SNRs for all considered DL algorithms is

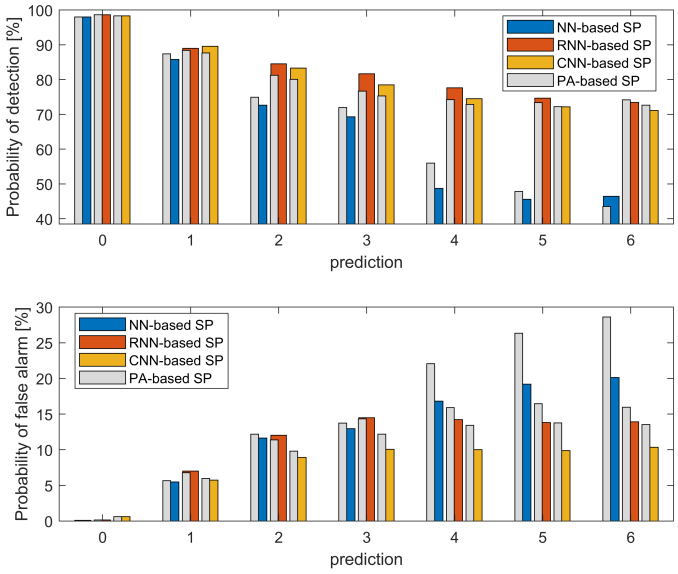
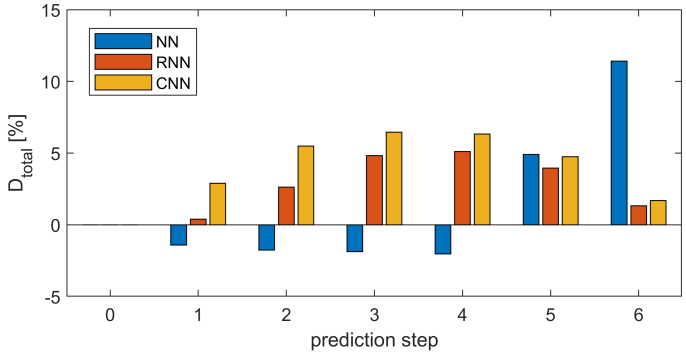
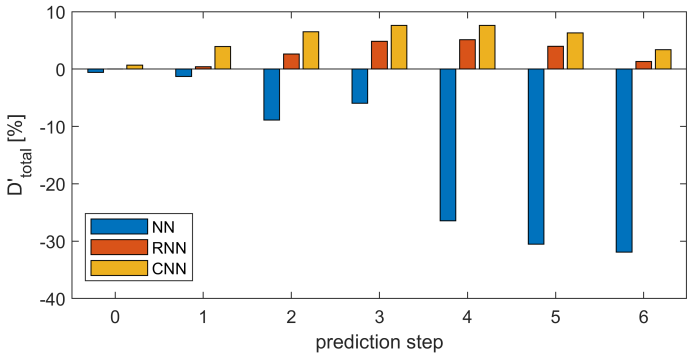


FIGURE 3.10: Probability of detection and probability of false alarm vs. the prediction horizon (prediction step) for SNR = 12 dB and the first dataset.



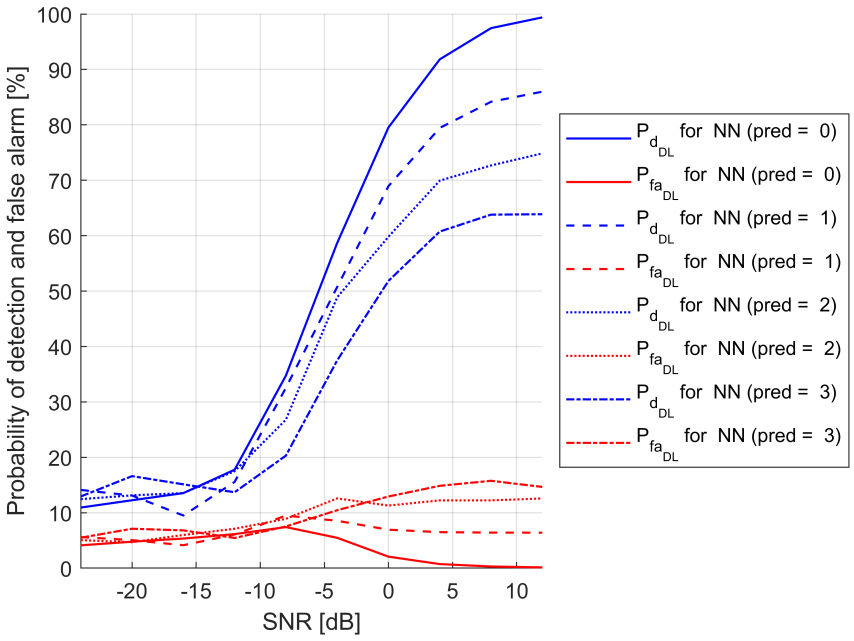
(A) Evaluation measure D_{total} vs. the number of steps in prediction for SNR = 12 dB (first dataset)



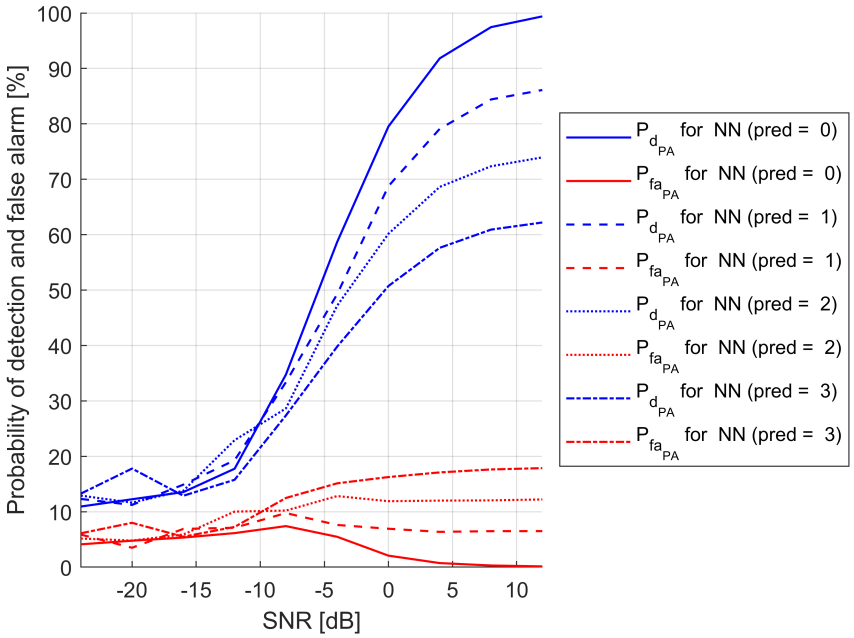
(B) Evaluation measures D'_{total} vs. the number of steps in prediction for SNR = 12 dB (first dataset)

FIGURE 3.11: Evaluation measure D_{total} and D'_{total} (first dataset)

small. Figure 3.12 presents NN-based SS and spectrum prediction results based on NN and PA. For low SNRs $P_{\text{d}_{\text{DL}}}$ and $P_{\text{d}_{\text{PA}}}$ equal only around 12%, while $P_{\text{fa}_{\text{DL}}}$ and



(A) NN-based SS and SP.

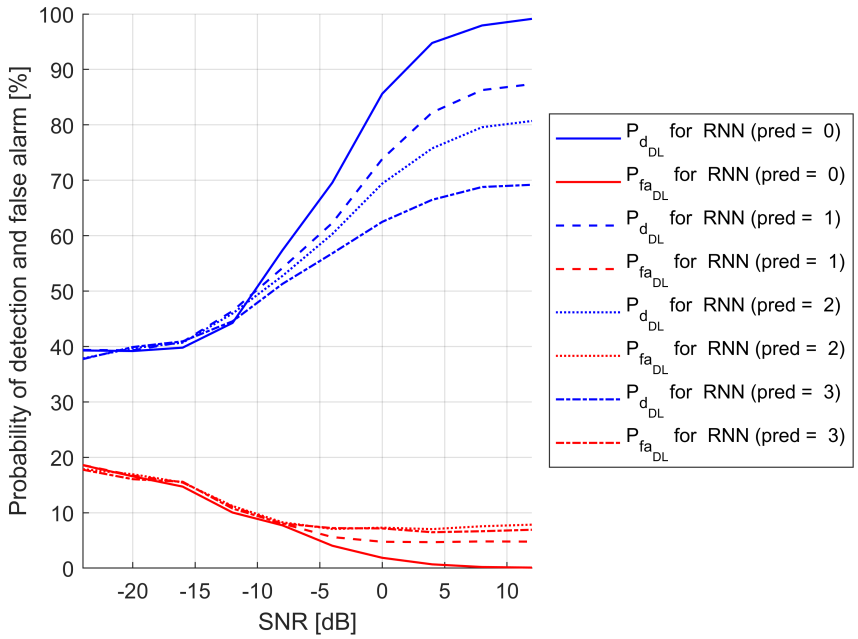


(B) NN-based SS and PA-based SP.

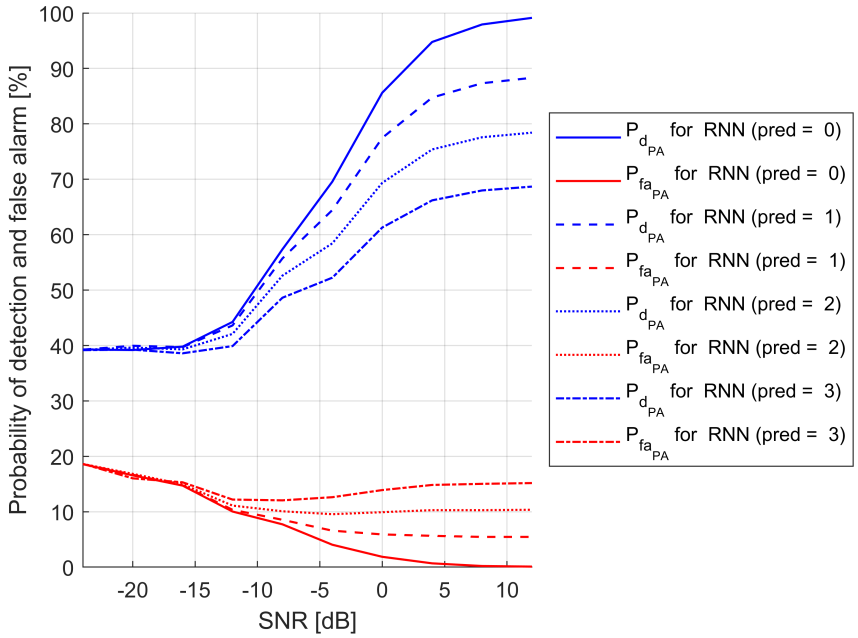
FIGURE 3.12: Probability of detection and false alarm vs. SNR for NN-based SS and prediction for the second dataset.

$P_{\mathrm{fa_{PA}}}$ equal 5%.

Analogously, Figure 3.13 presents results for the RNN-based SS and prediction based on RNN algorithm or PA. Similarly, as in the case of the first dataset, for the second dataset, RNN-based algorithms perform better in terms of the achieved values of $P_{\mathrm{d_{DL}}}$, $P_{\mathrm{d_{PA}}}$, $P_{\mathrm{fa_{DL}}}$ and $P_{\mathrm{fa_{PA}}}$ than NN-based ones. The RNN is better suited to recognize signal patterns despite noise and random signals. For low SNR values, it is not possible to associate a specific time with a higher or lower intensity of



(A) RNN-based SS and SP.



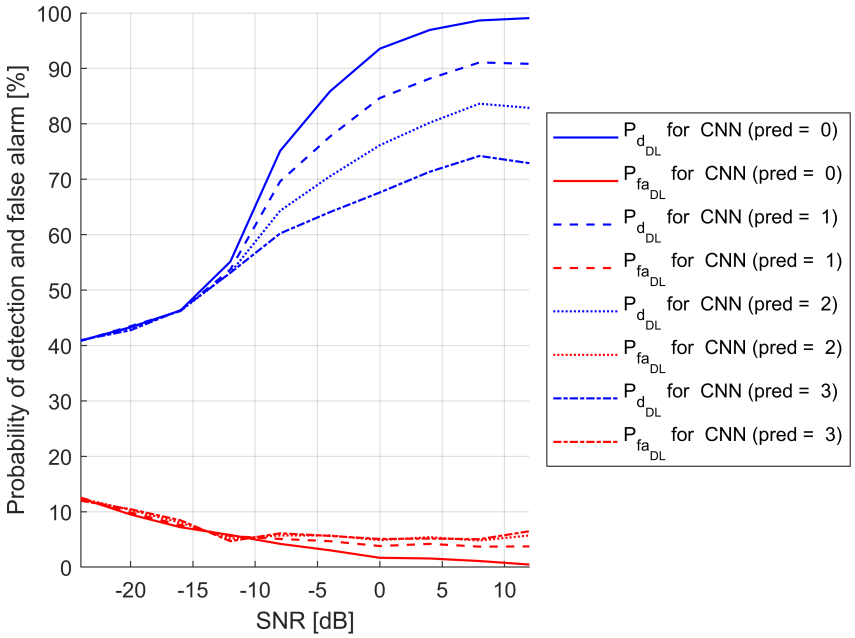
(B) RNN-based SS and PA-based SP.

FIGURE 3.13: Probability of detection and false alarm vs. SNR for RNN-based SS and prediction for the second dataset.

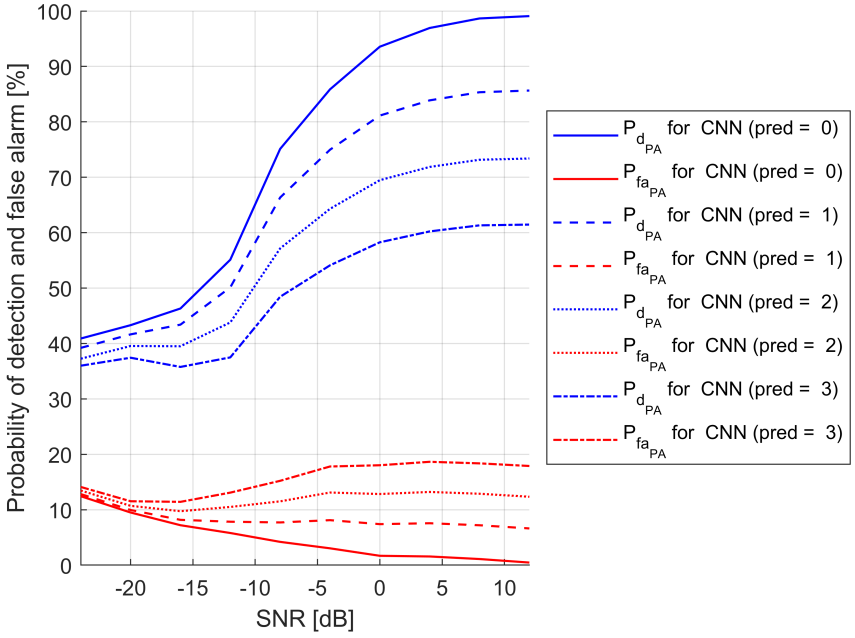
the communication traffic. However, predicting the deterministic and periodic RBs occupancy is still possible, hence the gap between $P_{\mathrm{d_{DL}}}$ and $P_{\mathrm{fa_{DL}}}$ and between $P_{\mathrm{d_{PA}}}$ and $P_{\mathrm{fa_{PA}}}$ is around 20%. It should be noted that for high SNRs, and for prediction steps from 1 to 3, the values of $P_{\mathrm{fa_{DL}}}$ are lower than $P_{\mathrm{fa_{PA}}}$.

Figure 3.14 shows analogous results achieved with the application of CNN. In this case, the differences between the CNN-based and PA-based SP results can be seen more clearly than before. The advantage of the CNN algorithm for spectrum

prediction over PA is visible in the high SNR region, i.e., $P_{\mathrm{d_{DL}}}$ relative to $P_{\mathrm{d_{PA}}}$ grows with each prediction step and $P_{\mathrm{fa_{DL}}}$ is much lower than $P_{\mathrm{fa_{PA}}}$.



(A) CNN-based SS and SP.



(B) CNN-based SS and PA-based SP.

FIGURE 3.14: Probability of detection and false alarm vs. SNR for CNN-based SS and prediction for the second dataset.

Similarly, as in the case of the first dataset, the comparison of the considered algorithms for SS and SP in terms of the probability of detection and false alarm for SNR=12 dB and for multiple prediction steps has been collectively presented in Figure 3.15. As previously, the upper graph presents $P_{\mathrm{d_{DL}}}$ for each DL-based SP and the corresponding $P_{\mathrm{d_{PA}}}$ for PA-based SP. The lower graph shows $P_{\mathrm{fa_{DL}}}$ for each

DL-based SP and the corresponding $P_{\mathrm{fa_{PA}}}$ for PA-based SP. The CNN algorithm appears to be the best choice in terms of the probability of detection and false alarm performance in comparison to the other two algorithms, which is also visible in Figure 3.16 showing the evaluation measure D_{total} defined by equation (3.1) and D'_{total} defined by equation (3.2). Moreover, the NN-based SP does not outperform RNN-based PA for any prediction step higher than 0, although it does outperform PA-based prediction results based on the input from NN-based SS.

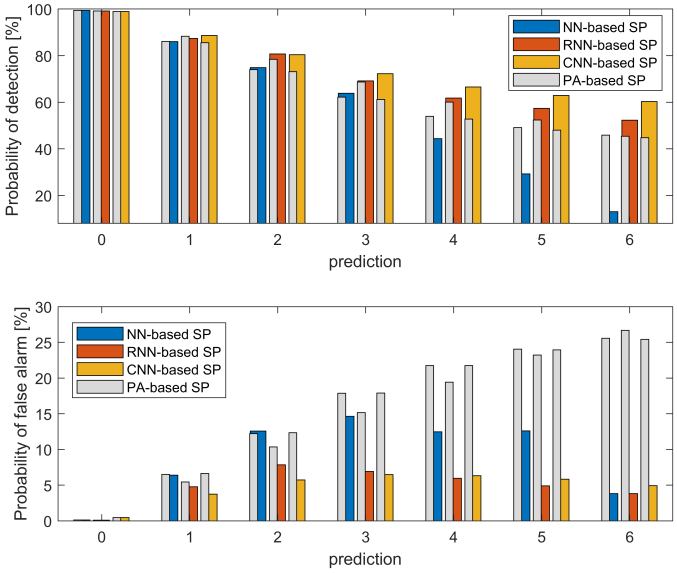
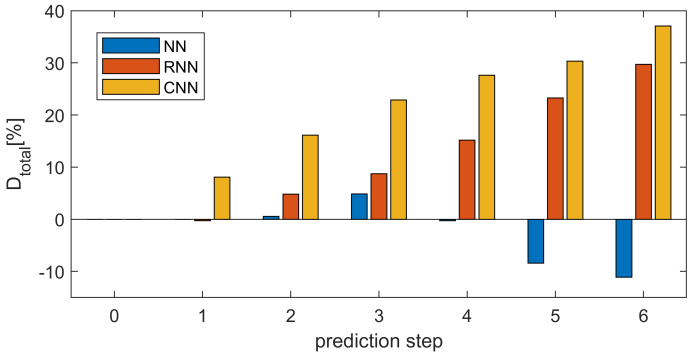


FIGURE 3.15: Probability of detection and probability of false alarm vs. the prediction horizon (prediction step) for SNR = 12 dB and the second dataset.

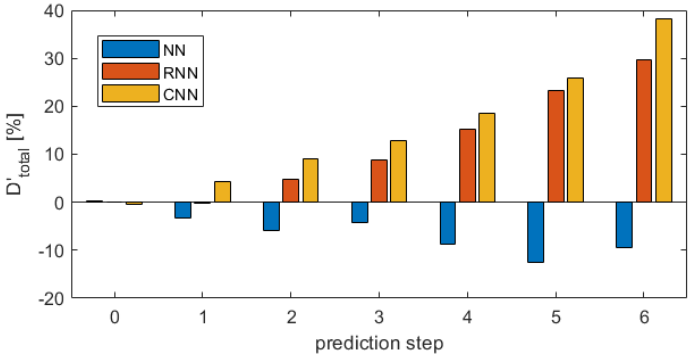
3.3 New DL Algorithm for Spectrum Occupancy Prediction Combined with Fading Level Estimation

In this section, the author focuses on SS and future spectrum occupancy prediction, or simply: SP. At the same time, to improve SS/SP results, the fading channel distortions are estimated by the ML and incorporated in the overall new proposed algorithm. SS/SP and channel fading estimation are assigned to separate dedicated modules using CNN. (Note that CNN proved to be the best fitting method for SS and SP in the considered 5G scenarios.) For this purpose, two-dimensional matrices (which can be interpreted as images) of energy values observed in time and frequency dimensions are formed.

In the proposed approach of the author, the SU first calculates the signal energy. Noise estimation is not mandatory, as the calculated energy is not compared with any threshold while deciding on the availability of the spectrum. The SU can only use free spectrum resources when PU is not transmitting, as the protection of PU's transmission is a priority in the considered approach. However, signal fading in the channel between PU and SU may mislead the SS algorithm and result in known *hidden node effect*, i.e., result in faulty detection of the spectrum availability in a



(A) Evaluation measure D_{total} vs. the number of steps in prediction for SNR = 12 dB (second dataset)



(B) Evaluation measure D'_{total} vs. the number of steps in prediction for SNR = 12 dB (second dataset)

FIGURE 3.16: Evaluation measure D_{total} and D'_{total}

given band, reuse of this band by SU and generated interference corrupting the PU transmission. To prevent this effect, the author proposed a novel algorithm that combines SS and SP with fading channel evaluation. Knowledge of fading level assesses the quality of signal reception. If a given set of spectrum resources is strongly faded, it should not be decided as free (unoccupied). The proposed SS algorithm forces SUs to avoid transmission in a given frequency band if the channel is predicted to be in a state of strong fading. Thus, it protects the PU transmission more effectively than standard SS/SP. To perform both SS/SP and fading evaluation, the author designed an algorithm that uses two CNNs for both problems. Additionally, a considered scenario includes more than one PU (which can be a more typical situation, although, in most papers, it is not considered). Experiencing different channels (and fading effects) with different PUs, an SU aims at protecting all PUs transmissions and simultaneously finding free resources for its use.

3.3.1 Proposed CNNs-based algorithm and data set

As in 5G, high flexibility in RBs assignment is guaranteed due to the flexible numerology; the goal is to opportunistically make use (by SU) of the spectrum gaps (created by PUs). Thus, let us now consider the scenario that includes SU located in an area where it can simultaneously receive multiple PUs' 5G signals experiencing different channel conditions (e.g., transmitted in the uplinks (ULs)), and narrow

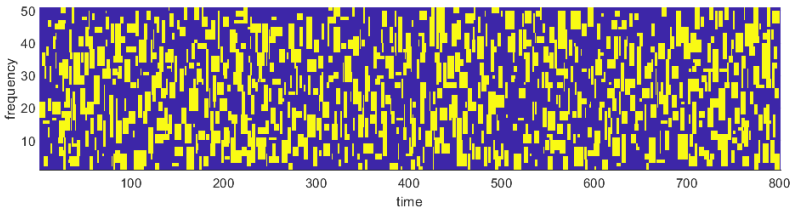


FIGURE 3.17: Generated examples of one of the PU's RBs occupation in time and frequency domain.

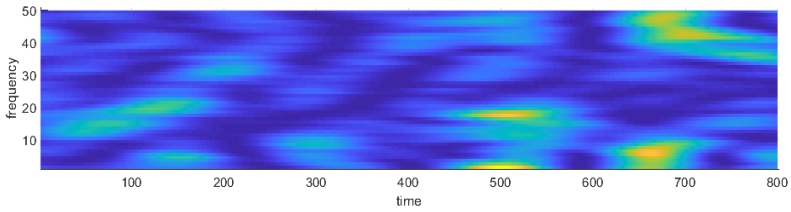


FIGURE 3.18: Visualization of fading channel (between the SU and one of the PUs) effect on all RBs.

the considerations to two such PUs. It is assumed that both PUs' signals are of the same priority and SU has to protect their transmission, so when the prediction indicates that the spectrum is not (or will not be) available, the SU has to stop its transmission or select another predicted time-frequency gap. However, once the spectrum has been detected/predicted as available, the SU may start allocating and using resources in the identified time-frequency gaps. It is also assumed that both PUs use the same spectrum band, and their resource allocation can overlap. The example of the traffic simulated during the experiments and generated by a single 5G PU is shown in Fig. 3.17, where the yellow color indicates the presence of PU data at a given RB, and the blue – the absence of PU signals. In the considered scenario, the RBs used for transmission occur in groups, a common way of allocating resources in 5G/6G. This means there is a higher probability of occupied RBs adjacent in time and frequency domains. The ML algorithm is supposed to recognize this dependency and determine the time and frequency probability distribution of RBs occurrence.

As mentioned above, the resource allocation methods introduce patterns into the signal. The considered signal is transmitted through a multipath wireless channel that introduces effects like shadowing and fading. The channel's influence on each RB (again for a single exemplary PU) is illustrated in Fig. 3.18. The darker-colored areas indicate a stronger fading. The multiple (two) PUs, and their channels' impact on the signal received by SU is presented in Fig. 3.19. Here, one can observe summed energy in RBs of two PUs signals distorted by their corresponding channels and received by SU.

In the proposed scenario, SU is moving in an area with constant mean SNR values of both PUs' signals. The instantaneous SNR signal value for a given RB depends on the fading effect. Additionally, the shadowing effect impacts the value of SNR.

In the following, the author proposes an algorithm capable of detecting and predicting signals in the presence of fading. The author employs one CNN (CNN_1) to detect (SS) the presence of the signal in a given moment and also to predict (SP) the possibility of its appearance in the upcoming RBs. The level of fading

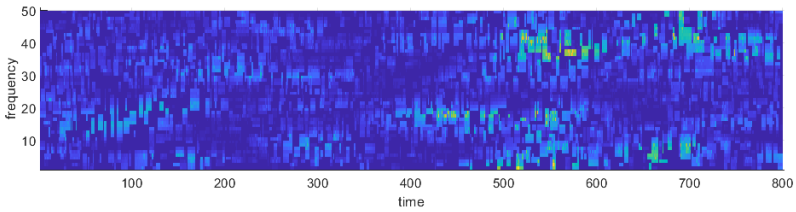


FIGURE 3.19: Received signal energy originating from two combined PUs' transmissions affected by their fading channels (energy values per RB).

is also estimated at a given RB (and the upcoming RBs) by using another CNN algorithm (CNN_2). Both models accept the same type of input data in the form of spectrogram-like images in which pixels contain energy values (EV). The CNN models are presented in Fig. 3.20. CNN_1 model (left) makes separate decisions on the occupancy of all frequencies for the current RB and the three upcoming RBs. CNN_2 (right) evaluates the fading level in the same time and frequency band for the same RBs as CNN_1. The training labels for CNN_1 contain binary values that represent the occupancy status of the resource. For calculating the CNN_2 labels, a signal with all RBs occupied has been generated with the same channel as the signal used as input in both CNN_1 and CNN_2. The training labels for CNN_2 contain values in the range $< 0, 1 >$. They were generated as normalized energy values calculated per RB.

In order to combine the output decisions of CNN_1 and CNN_2 into one improved SS/SP result, it is crucial to determine what fading limit value should indicate automatic protection of the potential PU transmission that could not be detected due to strong fading. The best limit value, the fading threshold (thr), is chosen to ensure the maximum possible probability of PU protection and minimum possible loss in SU opportunities to transmit. This threshold value is chosen separately during the learning process and during the testing phase, it stays constant.

Combining both CNN results, a joint occupancy decision was created that was equivalent to deciding whether SU can transmit on specific spectrum resources. The algorithm scheme is presented in Fig. 3.21.

3.3.2 Simulation Experiment

Simulation Setup

To perform experiments, the author has simulated 5G downlink signals as PUs' transmissions. Transmission occurs over a 10 MHz band comprising 50 RBs. Every RB consists of 7 OFDM symbols transmitted in a 0.5 ms slot over 12 subcarriers. The energy values (which are used as CNN input spectrogram data points) are calculated only for the first OFDM symbol in each RB. This approach leaves time for decision-making and potential transmission during the time of the rest of the OFDM symbols. It is worth adding that the performance of SS using energy values collected over more OFDM symbols has been tested, and the benefits of this approach were negligible. The considered downlink signal appears randomly in the time and the frequency domain, although the occupied RBs appear grouped, which is typical for the 5G systems to avoid spectrum fragmentation. The signal received by SU is distorted by shadowing and multi-path fading. The Extended Vehicular A model [153] with a Doppler frequency equal to 50 Hz is used as the fading channel model

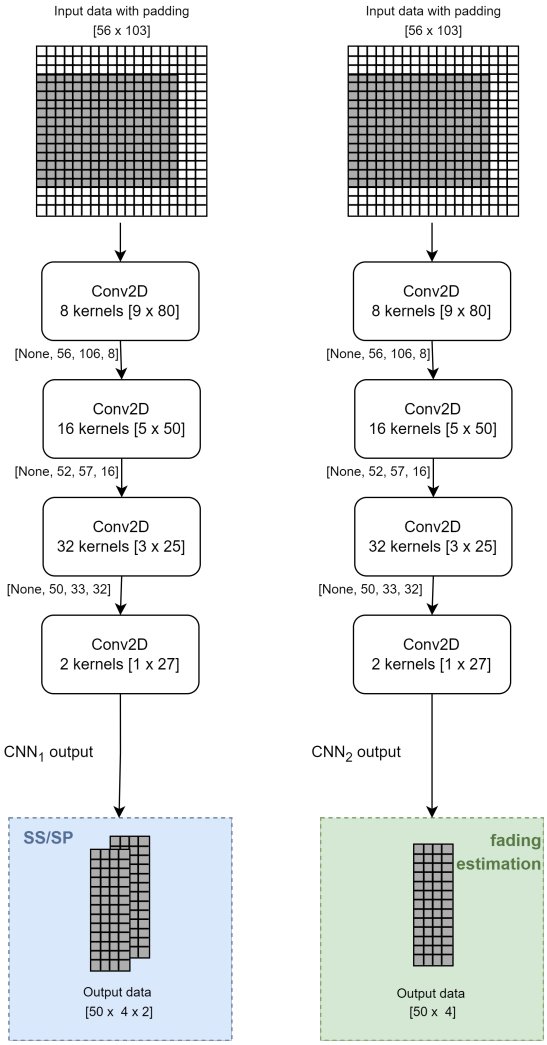


FIGURE 3.20: CNN models for spectrum sensing, prediction (left), and fading level estimation (right).

experienced by SU. Matlab software has generated the signals and the channel effects affecting those signals.

It is assumed that SU can train its own CNN models, or it can download them from an external source. A given CNN model has been trained for a specific range of SNR values.

The CNN_1 model uses the rectified linear unit function as the activation function for all layers except for the last layer, where the softmax function is used. Since detection is the categorization problem, the Sparse Categorical function [41] is used for the loss calculation of this model. The Adam optimizer [82] with a learning rate of 0.0001 is used in the learning process. This optimizer with that learning rate has yielded the best training results during tests run to find the best training setup. As the result of the CNN_1 model, the occupancy probabilities of each considered RB are calculated, which then are discretized to 0 and 1 values, where 0 means that a given RB is not occupied and 1 means that it is occupied. It is assumed that SU can train its own CNN models, or it can download them from an external source. A given CNN model specializes in a combination of one set of SNR values.

CNN_2 model uses sigmoid functions as activation functions in all layers. Similarly, as in the CNN_1, the Adam optimizer is used, but with a larger learning rate

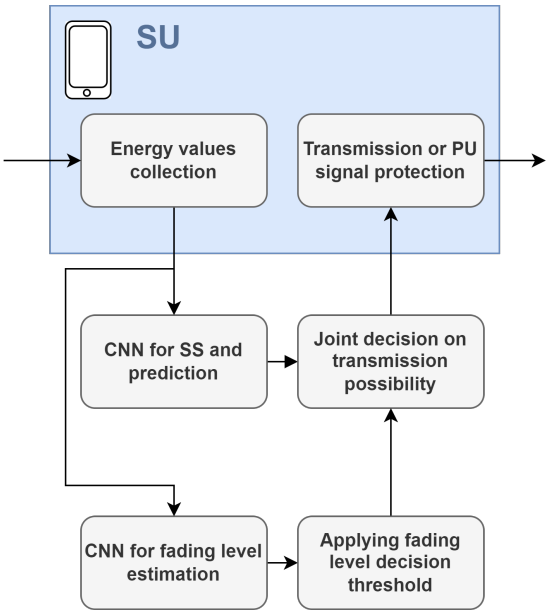


FIGURE 3.21: Joint detection, prediction, and fading level estimation algorithm for enabling SU to reuse the spectrum with simultaneous protection of the PU signal.

equal to 0.001. As a loss function, the mean squared error is calculated.

The fading level is represented by a value scaled to the $< 0; 1 >$ range. Value 0 denotes the lowest possible fading level observed, and 1 represents the best PU-SU link conditions. The method of finding the best *thr* value is described later in the chapter.

To obtain representative results, the 45 locations of SU have been picked. Each location is characterized by different mean SNR (resulting from the shadowing) values of PUs signals combination and the fading channel. The SNR values used in the simulation range from -20 dB up to 20 dB with a granularity of 5 dB.

Evaluation methods

The probability of correct detection and the probability of incorrect positive detection (probability of false alarm) are used as metrics for the evaluation of the proposed algorithm. The goal is to maximize the overall value P_{d}, which measures how well the transmission of PU is protected. At the same time, P_{fa} should be minimized because the small value of this metric ensures the most SU transmission opportunities.

Thus, to find and evaluate the best fading level threshold *thr*, the author proposes to maximize the weighted increase in the probability of detection and decrease in the probability of false alarm, i.e., to find *thr*, such that:

$$\arg\max_{thr}[a \cdot (P_{\mathrm{d}}^{thr} - P_{\mathrm{d}}^{\mathrm{CNN}_1}) + b \cdot (P_{\mathrm{fa}}^{\mathrm{CNN}_1} - P_{\mathrm{fa}}^{thr})], \tag{3.3}$$

where $P_{\mathrm{d}}^{\mathrm{CNN}_1}$ and $P_{\mathrm{fa}}^{\mathrm{CNN}_1}$ are the probability of detection and the probability of false alarm which would make up a result from using only the CNN_1 outcome as a final decision. P_{d}^{thr} and P_{fa}^{thr} are the probability of detection and the probability of false alarm, which would make up a result if a fading level threshold value would be equal to *thr*, and the CNN_1 and the CNN_2 decisions would be combined according to the proposed algorithm. The coefficients a and b are used to emphasize either

the subtraction of the probabilities of detection or the subtraction of probabilities of false alarm. If the first subtraction is set to be more important ($\frac{a}{b} > 1$), it means that the protection of the PU's signal is more important. If, on the other hand, $\frac{b}{a} > 1$, more emphasis is put on maximizing SU's transmission opportunities. In the experiments, factor b has been set to a constant value of 1. The author has considered factor a (further referred to as *PU-protection parameter*) to be either equal to 1 or given by the so-called *logistic function* dependent on PU's SNR values and detection/prediction stage:

$$a = f(s_1, s_2, pred) = \gamma + \frac{1}{pred + 1} \cdot \frac{1}{1 + e^{k(\theta - \sqrt{s_1 \cdot s_2})}}. \tag{3.4}$$

The logistic function given by equation: $\frac{1}{1+e^{k(\theta-\sqrt{s_1 \cdot s_2})}}$ is shifted by γ and scaled by $\frac{1}{pred+1}$. The logistic function was chosen for a parameter in order to put more emphasis on PU's protection for higher SNR values. The parameters used in equation (3.4) are defined as follows:

- s_1, s_2 are SNR of one of the PUs (PU_1) and the other PU (PU_2) accordingly. SNR ([dB]) values from are from the considered range $< -20dB, 20dB >$ and are scaled into a linear range $< 0, 40 >$
- $pred$ indicates detection ($pred = 0$) or spectrum occupancy prediction horizon, also called the prediction step ($pred = 1, 2$ or 3 for next RB, RB after that, and RB after that, accordingly).
- γ is a parameter assumed to be equal to 1.1 in the simulation. The value γ is used for regulating parameter a influence in equation (3.3).
- θ sets a function midpoint. It regulates the influence of factor a depending on SNR value. In the considered scenario, $\theta = 20$.
- k is a growth rate of the steepness of the logistic function curve. The assumed value $k = 0.25$

The above parameters were determined experimentally. Finding the optimal parameters is a separate issue, an interesting challenge for future work.

The purpose of function $f(s_1, s_2, pred)$ is to keep parameter a value regulated to increase PU's transmission safety, with minimum costs in P_{fa}. Figure 3.22 presents this function versus $s_1 = s_2$ and for $pred = 0, 1, 2$ and 3.

Simulation Results

First, SS results have been generated only by CNN_1. Figure 3.23 shows $P_{\mathrm{d}}^{\mathrm{CNN_1}}$ and $P_{\mathrm{fa}}^{\mathrm{CNN_1}}$ results for SS (pred = 0), and prediction for pred = 1, pred = 2, and pred = 3. The received signal combines two PU signals. The PU_1's signal SNR is constant and equal to 20 dB, and PU_2 SNR varies in range from -20 dB up to 20 dB. Results for a combination of signals, which always have the same mean values of SNR, are presented in Figure 3.24. When comparing these two figures, one can observe that the results presented in Figure 3.23 are better from the lowest SNR up to -5 dB. This is because a much stronger signal sometimes overlaps with a weaker signal in resource allocation. The much easier detection of the stronger signal makes detecting occupied resources possible partially, as well as for the weaker signal, even

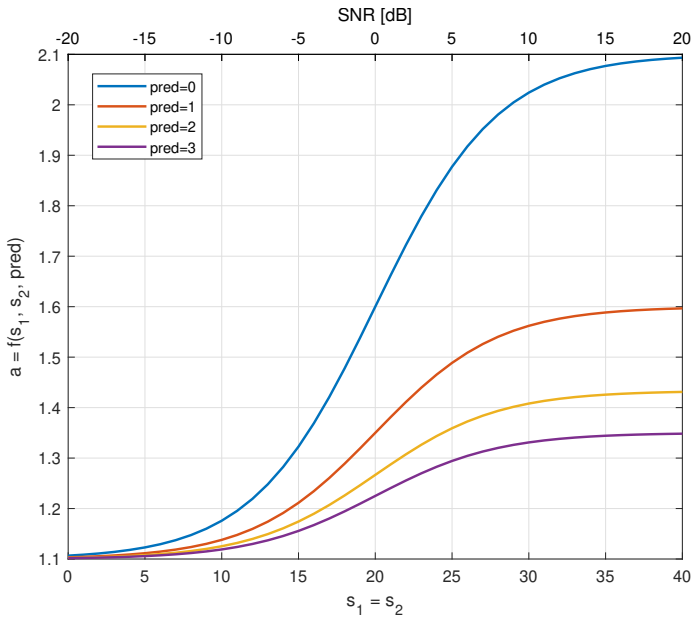


FIGURE 3.22: PU-protection parameter a as function of PUs-SU channels SNRs s_1, s_2 and the prediction perspective parameter $pred$: $f(s_1, s_2, pred)$.

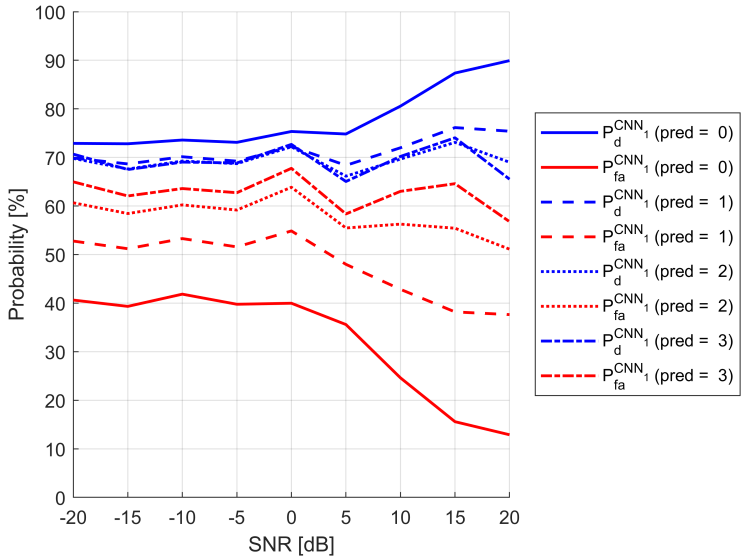


FIGURE 3.23: Probability of detection, correct prediction, and false alarm at the output of CNN_1 ($P_{\text{d}}^{\text{CNN}_1}$, $P_{\text{fa}}^{\text{CNN}_1}$) for SNR of PU_1 equal to 20 dB and SNR of PU_2 varying in range from -20 dB to 20 dB.

though it is purely by chance. The results of $P_{\text{d}}^{\text{CNN}_1}$ and $P_{\text{fa}}^{\text{CNN}_1}$ in both figures reach the same values for SNR equal to 15 dB and higher. This is because in Figure 3.23 PU_2 has a mean SNR value similar to PU_1's 20 dB.

Figure 3.25 has been created for equal SNR values of PU_1 and PU_2 by combining CNN_1 and CNN_2 decisions and applying chosen thr. The threshold has been calculated using equation (3.3), where $a = 1$ (recall that $b = 1$ is set for all experiments). The first noticeable thing that can be observed is significant growth in P_{d} and P_{fa} for the lowest SNR values. This happens because for low SNR, focusing on PU transmission protection is beneficial, especially since the chances of detecting free

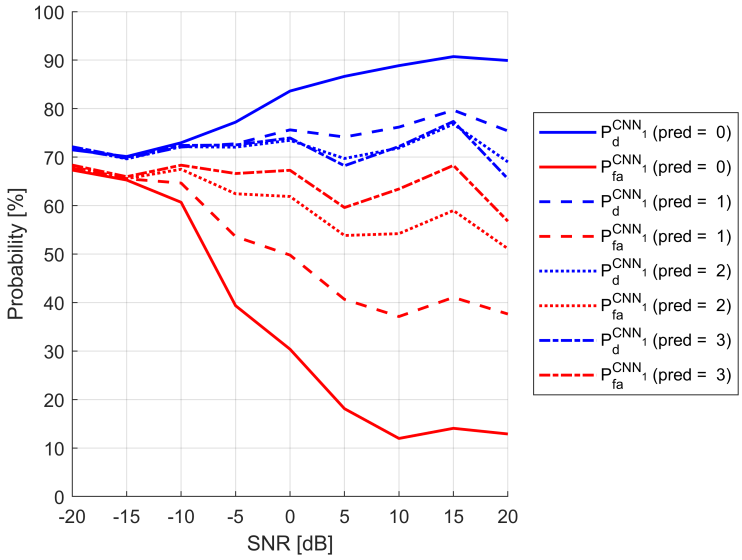


FIGURE 3.24: Probability of detection, correct prediction, and false alarm at the output of CNN_1 ($P_{\text{d}}^{\text{CNN}_1}$, $P_{\text{fa}}^{\text{CNN}_1}$) for equal SNR of PU_1 and PU_2 varying in range from -20 dB to 20 dB.

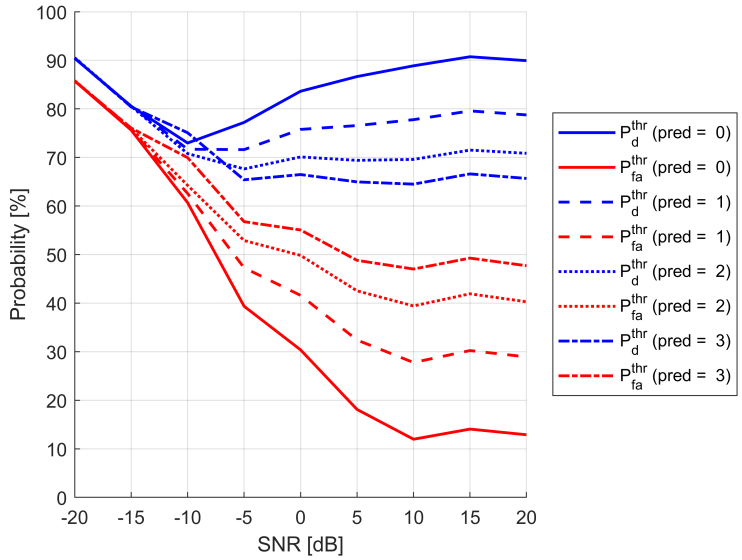


FIGURE 3.25: Probability of detection, correct prediction and false alarm at the output of the algorithm combining CNN_1 and CNN_2 decisions and applying threshold thr (P_{d}^{thr}, P_{fa}^{thr}) for equal SNR of PU_1 and PU_2 varying in range from -20 dB to 20 dB; $a = 1$.

resources are low. The proposed algorithm enables that. Another noticeable thing is improvement in overall performance, especially for higher SNR, mainly for P_{fa}, which has been lowered. This indicates more transmission opportunities for SU.

To compare the author's proposed algorithm with channel fading estimation with the application of CNN_1 alone, Figure 3.26 is presented. The values presented on this bar graph represent the gain of applying the proposed algorithm defined as follows:

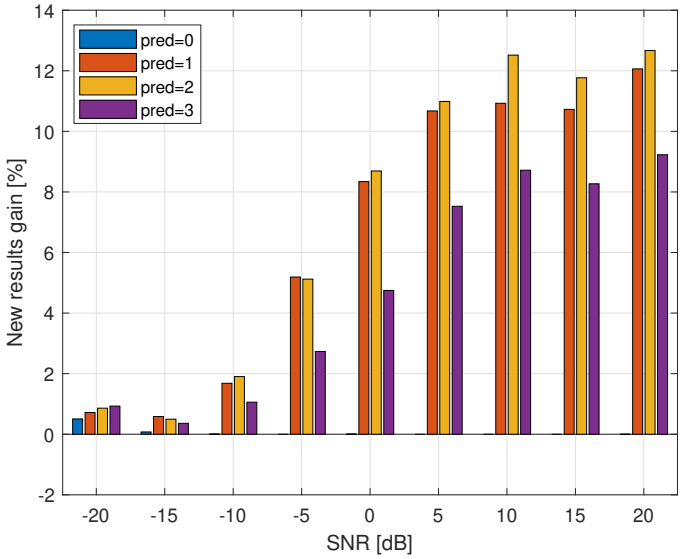


FIGURE 3.26: Proposed algorithm gain over CNN_1-only SS/SP (SNR of both PUs is equal); $a = 1$.

$$gain = P_{\mathrm{d}}^{\mathrm{thr}} - P_{\mathrm{d}}^{\mathrm{CNN_1}} + P_{\mathrm{fa}}^{\mathrm{CNN_1}} - P_{\mathrm{fa}}^{\mathrm{thr}}, \tag{3.5}$$

where $P_{\mathrm{d}}^{\mathrm{CNN_1}}$ and $P_{\mathrm{fa}}^{\mathrm{CNN_1}}$ are resulting of SS performed by CNN_1 and $P_{\mathrm{d}}^{\mathrm{thr}}$ and $P_{\mathrm{fa}}^{\mathrm{thr}}$ are calculated as a result of the proposed algorithm performance. The gain is the lowest for the lowest SNR and grows with SNR. For $pred = 0$, little gain has been observed for the lowest SNR values. The best improvement in SP (by applying the author's proposed method) is observed for $pred = 2$. This is because of a relatively high probability of false alarm in the case of SP based on CNN_1 alone that has been reduced by up to 10% in the case of the proposed algorithm with fading estimation and prediction. The proposed approach improved the detection and prediction by up to 13% for the highest considered SNR.

Finally, the last set of results shows how the the value of a affects P_{d}^{thr} and P_{fa}^{thr}. Figure 3.27 presents results for equal SNR of PU_1 and PU_2. However, this time, the parameter a has been established using equation (3.4) (again recall that $b = 1$ is set for all experiments). It can be seen that for the lower SNR region P_{d}^{thr} and P_{fa}^{thr} remain high in the wider SNR range and for higher $pred$ values. This results in higher PU protection in the case of worse signal detection conditions. Figures 3.25 and 3.26 show how different approaches to thr determination can affect either higher PU transmission protection or higher SU transmission possibilities.

3.4 Chapter summary

In this chapter, the DL algorithms have been considered to improve autonomous SS above the performance achieved by classification methods discussed in Chapter 2. Moreover, DL has been used for prediction future spectrum occupancy, and for fading level estimation.

In the first part of the chapter, three DL algorithms have been designed and examined, namely NN, RNN, and CNN. Two data sets have been tested, chosen to represent the time- and frequency-correlated traffic with random RBs occupancy

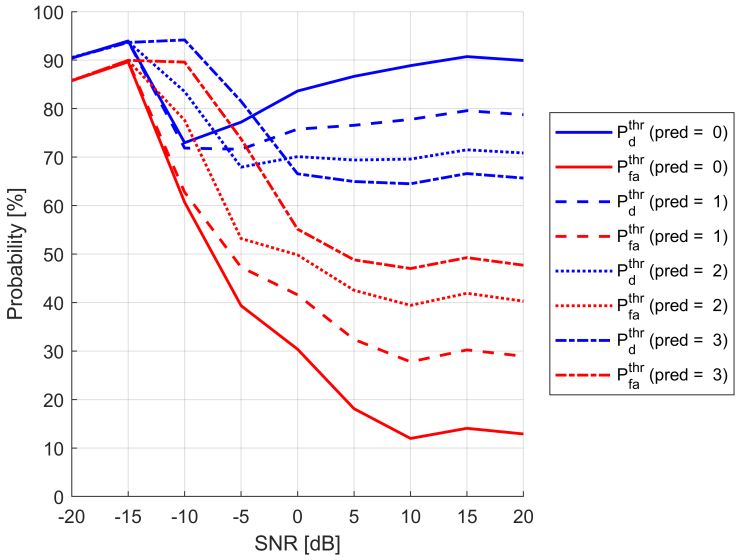


FIGURE 3.27: Probability of detection, correct prediction, and false alarm at the output of the algorithm combining CNN_1 and CNN_2 decisions and applying threshold *thr* (P_{d}^{thr}, P_{fa}^{thr}) for equal SNR of PU_1 and PU_2 after fading level estimation. Parameter a established using logistic function (3.4).

and the IoT-device traffic with periodic RBs occupancy on top of the time-correlated RBs occupancy. All three algorithms have been applied not only to spectrum detection but also to spectrum occupancy prediction. They have been compared to the simple prediction algorithm (called the Primitive Algorithm (PA)) based on the output from the respective DL method and simple repetition of these decisions for the succeeding time slots. The evaluation of all proposed algorithms leads to the conclusion that CNN-based SS and SP is the best-fitting method in all experiment scenarios. It results in the highest probability of detection and the lowest probability of false alarm for all respective prediction horizons and SNR regions. Thus, further experiments on CNN application in SS and SP were performed and examined in the second part of the chapter.

In the second part of the chapter, the author has proposed and examined a new CNN-based algorithm for improving the spectrum sensing and prediction of spectrum occupation in a 5G downlink transmission scenario. Additionally, she combined CNN-based SS/SP with CNN-based fading level estimation and applied an optimized threshold to the signal fading level in order to reject decisions on unoccupied RBs in the case when these RBs experience deep fading. In this way, the potential PU transmission that could not be detected due to strong fading is automatically protected. Despite the diversity of the received PUs signals and their fading patterns, the improvement in the SS and SP performance has been achieved. This, in consequence, allows us to maintain or improve PU's transmission protection from the interference caused by SUs while increasing SUs transmission opportunities. The main advantage of the proposed solution is an upswing in spectrum occupancy prediction performance, which will facilitate SU preparation for prospective transmission and further improve the effectiveness of secondary spectrum reuse and spectral efficiency in a given area, time, and frequency band.

In conclusion, DL algorithms, and mostly CNNs, proved to be a valuable and versatile tool for estimating occupancy state of frequency resources.

Chapter 4

Federated Learning for Cooperative Spectrum Sensing

ML-based SS using autonomous sensors has limited reliability due to distortions of a wireless channel. However, if frequency-selective fading dependencies can be uncovered (within the channel coherence time), the probability of misdetection can be reduced. Alternatively, a centralized ML approach would require extensive training datasets with high-resolution localization data, which may be impractical to acquire.

The individual sensing performed by each SU and employing ML is computationally complex and may need to be more accurate. This is because ML algorithms require considerable training data to recognize the time, frequency, and location dependencies existing in the transmitted and received signal. The end-user terminal usually does not have enough computing and memory resources to store and process the volumes of training data required to train an ML algorithm. Another problem in individual sensing is in obtaining labeled data for supervised learning. In practice, the end user (individual SU) cannot produce their own labeled data, so a need to download them from an external server appears. This approach is also impractical, requiring communication resources and data download time. The delay in downloading the data may cause them to be useless when a mobile SU changes location, which would require retraining the ML model and a new training dataset.

Thus, the ML model creation stage should be delegated to more computationally capable (centralized) devices so that SU could benefit from intelligent SS methods without spending time, energy, and computational resources on the ML model training. The popular idea is to employ Cooperative Spectrum Sensing (CSS), where SUs exchange their sensing results or collected data to decide cooperatively on the current spectrum state. This approach solves the problem of generating an ML model, which the elected end-user device, fusion center (FC), or central server can create. However, it still needs to answer the problem of collecting labeled data by SUs.

A promising solution to the abovementioned problems is Federated Learning (FL), an iterative procedure in which edge devices (called FL nodes) create their ML models on their local data. The created models are then exchanged in a centralized or decentralized manner to create one common ML model that can be shared among the mentioned devices. FL nodes then adjust the standard model to their local data, and the process of local training, exchanging models, and creating the standard model is repeated. This method has many advantages:

1. It allows to supply the incoming SUs with an ML model suited for their current

wireless environment and locations without the SUs needing to collect data.

2. It is robust against the changing radio environment conditions.

3. It limits the information volume exchanged between SUs and the FL server and provides data *privacy by design* (only the models' parameters need to be exchanged instead of raw data).

Whenever SU changes its location or the radio channel quality changes, it receives a new ML model adapted to the channel state.

The FL algorithms have been applied in various contexts related to wireless networks. For example, in [187], the client selection and bandwidth allocation for wireless FL networks are discussed, and the authors concentrate on the long-term resource allocation perspective. An interesting analysis of non-independent and identically distributed (non-IID) data processed in dynamically changing wireless networks is presented in [204], where the averaging scheme is proposed to reduce the distribution divergence of such kinds of data. Next, in [193], energy efficiency is discussed in the context of federated learning over wireless networks. A good summary of the challenges and opportunities for wireless federated learning is provided in [23]. Referring to SS, in 2019, the authors of [168] proposed the application of FL to spectrum access system for the Citizens Broadband Radio Service band system. In particular, evaluating their non-coherent spectrum-sensing system called FaIR (Federated Incumbent Detection in CBRS) showed that FL-based solutions may obtain an improved detection model compared to a naive distributed sensing and centralized model framework. One of the recent papers [33] deals with introducing the FL framework for CSS and proposed FL-based SS. Only the two abovementioned articles relate to SS.

As mentioned above, the FL algorithm can be centralized or decentralized [112]. The centralized FL means that one central server orchestrates the process, i.e., manages FL nodes and creates a global ML model. In decentralized FL, there is no central server, and the participating nodes must coordinate among themselves to create a global model. In this chapter, the author of this thesis considers the centralized version of the FL algorithm. She presents her original contribution to the unexplored field of the application of FL to SS. In Section 4.1, the basics of the FL-based SS method are described. Next, in Section 4.2 details of the new FL algorithm proposed by the author for SS in the diverse wireless environment conditions is presented. This is followed by the description of the computer simulation experiments and their results in Section 4.3. Research results and key findings are summarized in Section 4.4.

4.1 The basics of FL-based SS

As already mentioned, the issue with SS by individual agents is limited reliability; the quality of decisions is affected by the agent's specific radio environment, limited computational and memory resources, as well as limited availability of labeled datasets (so there is a need to download them from an external database, which may introduce errors and delay). FL is a concept that can resolve the problem of handling distributed datasets. In this case, all the training data are collected (or measured) locally, train local models and only their weights (or parameters) are transferred to the central FL server. Such an approach reduces the required radio

resources (bandwidth, time, and energy) and the data processing latency by sending only the model parameters instead of the raw data stream [121].

FL is an iterative procedure employing edge devices (sensors onboard CR User Equipment (UE)s, called FL nodes or FL agents) that develop their ML models based on locally measured data. The local models are then exchanged in a centralized or decentralized way to create one aggregated model shared among the devices. In the case of the centralized approach, locally developed model parameters, either all of them or only the ones that define some parts of a model (e.g., distilled models' parameters [33]), are transmitted to an FL server, where the aggregated (standard) model is created. FL nodes then adjust the corporate model to their local data, and the process of local training, exchanging models, and aggregating them is repeated (see Fig. 4.1, where FL nodes are considered CRs). Apart from the model aggregation, the FL server may also be in charge of FL node clustering, reflecting the location-specific availability of spectrum resources.

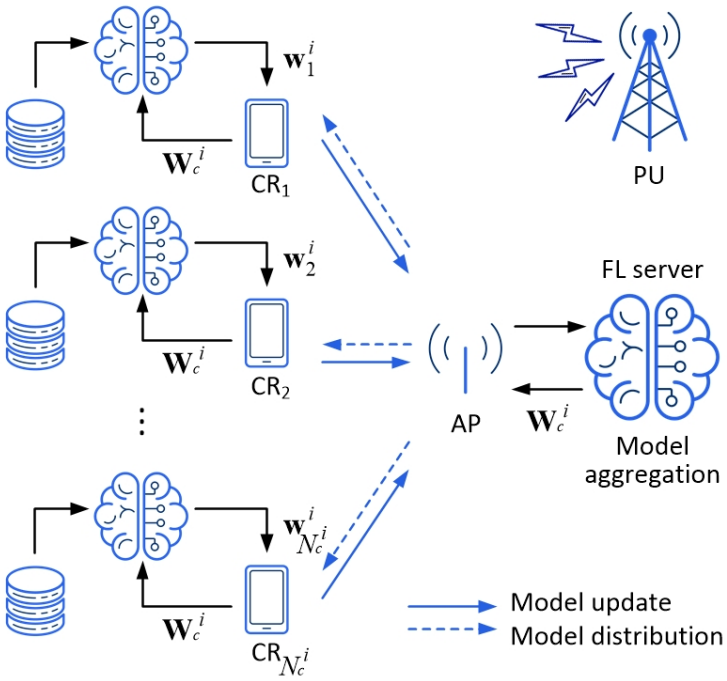


FIGURE 4.1: Federated learning for spectrum sensing.

Apart from the advantages mentioned above, FL-based SS provides a new incoming CR UE with a spectrum-occupancy-reflecting model suited for its current wireless environment and location without the need to collect data and train a model. It can also adapt to the changing radio environment. Whenever the radio channel quality changes for a CR UE, it receives a new FL model adapted to the channel state.

4.2 New algorithm for FL-based SS

4.2.1 Selecting the individual sensors' ML method for spectrum sensing and prediction

When analyzing the specific features of the transmitted downlink signal (in terms of the periodicity, which the ML tools can reveal), one can observe that some oc-

curring patterns characterize it. In the context of large time scales, the intensity of the traffic fluctuates periodically, following daily changes in resource requirements. However, on a short time scale, one can notice that adjacent RBs (both in time and frequency) are used for transmission, creating the RB group or their concatenation. It means that RBs typically occupy a solid frequency band.

The data used during the training and testing phase consist of three values: the energy measured per RB and the frequency and the time denotation per RB (time-frequency coordinates of RB). These three values can be presented as two-dimensional tables (pictures) and combined into three layers of one picture (or Red, Green, Blue (RGB) color-model components of an image). This representation makes it possible for Convolutional Neural Network (CNN) to process and find inner dependencies in the input data because the considered signal shows the correlation in both time and frequency. The additional two layers of frequency and time data emphasize those dependencies and make it easier for CNN to find and predict repeating patterns in time and frequencies that are potentially used more frequently.

It is assumed that the FL nodes can use the collected data to perform ML, namely CNN-based learning, which can extract information in an RB from the surrounding (in time and frequency) RBs that are not affected by the Rayleigh fading in a given moment and extrapolate decisions regarding occupancy of these RBs onto the currently faded RBs. This is why the preference of CNN over simpler algorithms such as kNN or DTs or the even more complex RNN is in place. For kNN and RF to achieve the same results, additional data would be needed for each RB. These data should consist of not only energy, time, and frequency characterizing a given RB but also information on the energy of adjacent RBs. The more information on the energy of closer RBs, the better. To take into account the history of the signal, the energy value of RBs appearing a few time steps in the past should also be considered. The complexity of input features needed per RB grows when a less complex algorithm is chosen. The DL RNN algorithm can also be a good choice, and the results are comparable. However, because CNN naturally works on two-dimensional data, the author decided to employ it for the proposed FL-based time and frequency sensing.

4.2.2 Creating the FL model

Below, the FL algorithm is proposed by the author of this thesis for the 5G spectrum sensing. The learning algorithm has four major steps repeated iteratively. In the first step of the ith iteration, clustering of FL nodes is performed, i.e., the grouping of sensors that experience similar channel conditions expressed in mean SNRs. Here, the author proposes the *k-means* clustering algorithm [100] based on the mean SNR values estimated by the FL nodes. It is assumed that these values are transmitted to, and are known by the FL server. As a result, the FL server is aware of the clusters.

In the second step of the ith iteration, the sensors in clusters receive the corporate models' weights $\mathbf{W}_c^{i-1}$ common for each sensor in cluster $c \in \mathcal{C}^i$, where $\mathcal{C}^i$ is a set of clusters in the ith iteration, and aggregated in the previous $(i-1)$-th iteration. In the case of the first iteration, i.e., before the aggregated (federated) models exist, the sensors have to create and train their individual models independently. It is assumed that the FL nodes use CNNs of the same structure.

After the second step, the corporate CNN model becomes a new standard for each cluster of FL nodes (except for the case of the first iteration discussed above). Then, in the third step, the local models are modified at the FL nodes by training based on new locally collected data. The training of the received CNN model is performed in each FL node by calculating new weights $\mathbf{w}_s^i$ for each sensor s, where at the second step the initial value of $\mathbf{w}_s^i$ equals $\mathbf{W}_c^{i-1}$, and $s \in \mathcal{S}_c^i$ is the index (or identifier) of a sensor in the cluster c in algorithm iteration i, and $\mathcal{S}_c^i$ is a set of these sensors. The cardinality of set $\mathcal{S}_c^i$ (the number of sensors in the cluster c) is denoted as N_c^i. Training modifies the weights of the federated model received $\mathbf{W}_c^i$ to get $\mathbf{w}_s^i$ that minimize the loss function $l_n^i(X_n^i, Y_n^i; \mathbf{w}_s^i)$, where X_n^i defines the collected training data, and Y_n^i is a classification target (set of labels of the training data). The loss function is selected according to the training data used and the problem for which the ML model is applied. After this training phase, all the weights of the modified CNN models are transmitted back to the server.

In the final (fourth) step, the FL server merges the individual CNN models. The resulting global model for a given cluster is created by applying *federated averaging*, which means that the weights of the corporate cluster-specific model are equal to the average of individual CNN model weights received from FL nodes being part of this cluster, i.e.,

$$\mathbf{W}_c^i = \frac{1}{N_c^i} \sum_{s \in \mathcal{S}_c^i} \mathbf{w}_s^i \,. \tag{4.1}$$

The whole process is repeated iteratively. The number of iterations is generally not defined. The iterations of the FL algorithm are performed with a predetermined frequency, or whenever some additional information (not included in the algorithm explanation) signals the need for updating of the local models. The FL iterations should be performed every time there are significant changes in the data collected by the sensors in order to adapt the clustered models' weights to the changing environment. The algorithm of FL in the form of a diagram is presented in Figure 4.2, and a pseudocode in Algorithm 1.

The corporate model generated in the FL server can be used by an outside user who wishes to perform the task for which the FL model is intended but is unable for some reason (for example, cannot generate or download labeled training data, or his computational resources are limited) to create its own local model. In the scenario considered in this chapter, the outside user wants to perform SS on the received signal but does not have access to the labeled data.

4.2.3 Sharing the FL model

In the considered scenario, SU may appear anytime and anywhere in the considered area. If SU wants to gain information on the spectrum occupancy, it should send its location information to the FL server. The FL server is assumed to be aware of the area's average SNR map. Knowing the current location of the SU, the FL server can pick the best CNN model for this SU. Every time SU changes its location, it may request a new model for this location to ensure the best sensing performance. Moreover, even if the location of SU does not change, but the environmental conditions do, assigning a new model for a given location might be beneficial. The algorithm of serving the SUs by the FL server is illustrated in Figure 4.3.

Algorithm 1 An algorithm of FL for SS.

$i \leftarrow 0$
$I \leftarrow$ Number of iterations
while $i < I$ **do**
  $\mathcal{C}^i \leftarrow$ set of clusters (clustering sensors by mean SNR values)
  for each $c \in \mathcal{C}^i$ **do**
    for each $s \in \mathcal{S}_c^i$ **do**
      if $i == 0$ **then**
        initialize $\mathbf{w}_s^i$
      else
        $\mathbf{w}_s^i \leftarrow \mathbf{W}_c^{(i-1)}$
      end if
      update $\mathbf{w}_s^i$ to minimize $l_s^i(\mathbf{w}_s^i)$
    end for
    $W_c^i \leftarrow \frac{1}{N_c^i} \sum_{s \in \mathcal{S}_c^i} \mathbf{w}_s^i$
  end for
  $i \leftarrow i + 1$
end while

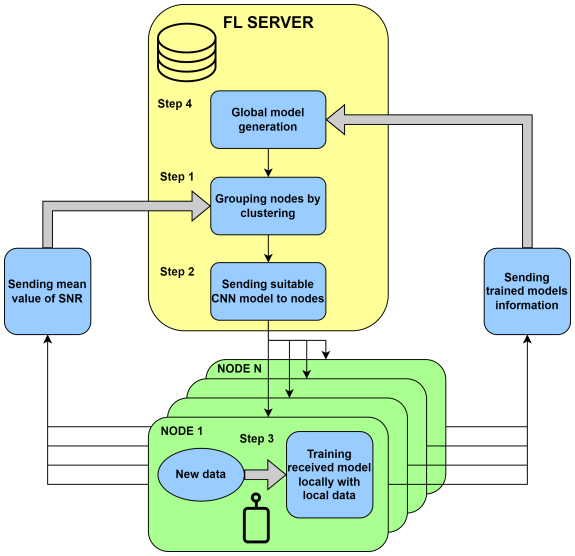


FIGURE 4.2: The algorithm of FL for SS.

After a new incoming SU receives the CNN model, it can feed it with its own collected data and perform sensing. The data collected by this SU is of the same type as the data collected by the FL nodes. The main difference (and advantage of the presented method) is that this new incoming SU does not need to acquire training data, as it already possesses a ready CNN model for SS. This means it can measure energy values per RB and use them as CNN input.

4.3 Simulation Experiment

Before proceeding with examining the FL-based SS in a particular clustered-sensors scenario, let's consider the benefit of using FL in a single cluster. The example results of FL-based SS performance are presented in Fig. 4.4. There, the

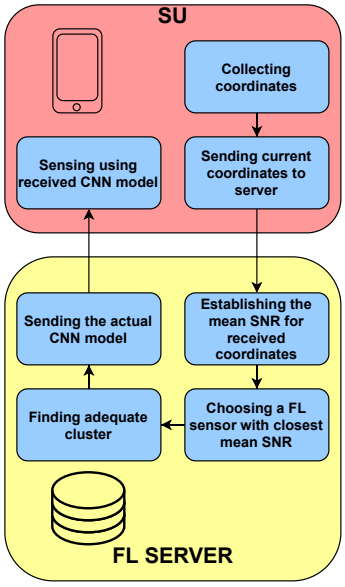


FIGURE 4.3: The algorithm of assigning the model to SU at a given location.

estimated probability of detection (true positive detection) P_{d} and the probability of false alarm P_{fa} (false positive detection) of the spectrum occupancy are presented as measures of the sensing quality. Five thousand patterns per each SNR value represented the PU transmission in the form of 50 x 100 resource elements in frequency and time (thus, 5000 RBs per one pattern) generated by the 5G downlink signal simulator, which mirrors the traffic-related time and frequency dependencies in the allocated RBs. The received SNR has been considered in the 0–20 dB range. The shared standard model has been built based on 8 CRs FL updates over 20 iterations by averaging the weights of their CNN models (with two hidden layers). CRs participating in FL have different channel models: Third Generation Partnership Project (3GPP) pedestrian EPA, Extended Vehicular A Model (EVA), and varying Doppler frequencies (in the range of 0.5 to 70 Hz). The corporate model has been shared, used and tested by three other (tester) UEs that have not taken part in the FL algorithm and have specific channel conditions and locally collected datasets.

Moreover, the tester UEs have also tested the locally learned models of individual CRs by importing them for sensing. The effects presented in Fig. 4.4 have been obtained by averaging the testers' results in these two sensing scenarios. The averaging has been done over varying Doppler frequencies of CRs (in the range of 2.5–55 Hz) and testers (in the range of 0.5–2.5Hz and 60 Hz–70 Hz) over the number of CRs, from which the models are randomly imported (only for basic sensing).

Figure 4.4 shows the FL-based SS performance against the so-called basic SS defined as SS performed by the CNN model trained only on local data, and not created or updated in the FL process. It can be observed that FL-based SS performs better than basic sensing in terms of both higher P_{d} and lower P_{fa}. This is because it builds a universal, standard model for data collected with different channel conditions than models built using data specifically for one channel type. The discrepancy in the channel conditions between CRs and testers explains this effect.

Following the above observations, the author proceeds to a simulation experiment of the application of the centralized FL algorithm for SS and prediction, which has been proposed in Section 4.2. The author assumes the correct operation of sensors and focuses on solutions for detecting and predicting the Resource Blocks (RBs)

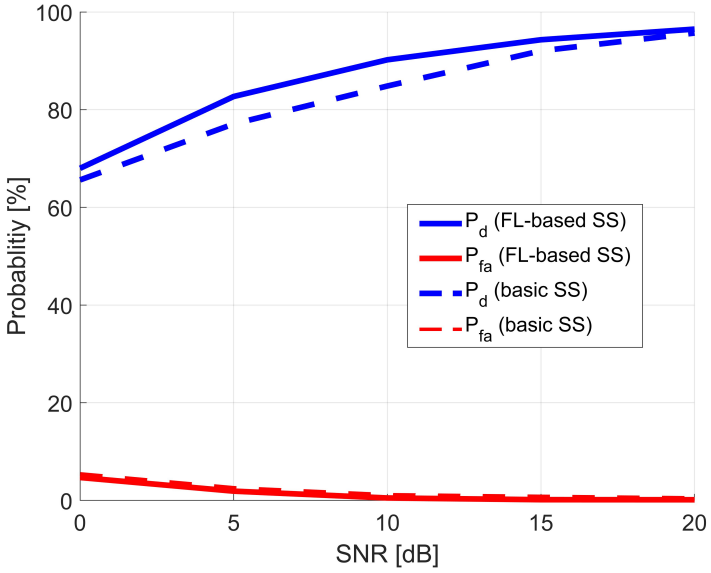


FIGURE 4.4: FL-based SS and basic SS performance.

occupancy in a 5G system. However, it should be emphasized that in the case of FL implementation, authentication of the participating FL nodes and information security must be supported. Examples of works on this subject can be found in [173, 130].

As for the primary system, the focus is detecting specific time-and-frequency patterns present in RBs, which are defined in time (as a set of time slots) and frequency (as a set of adjacent OFDM subcarriers) as physical resources. The Frequency Division Duplex (FDD) scheme is considered, and specifically, the downlink PU transmission. The SS system is aimed at detecting occupied RBs to enable the transmission of SUs in a way that does not interfere with the PU's signal.

Another type of information that the learning mechanism can process is related to the wireless channel. Apart from the path loss (function of the distance between an SU's or FL node's receiver and a PU's transmitter), the electromagnetic shadowing effect occurs and results in a specific relationship between the mean SNR value and the location of an SU's sensor. Finally, short-term frequency-selective Rayleigh fading (affecting a whole RB) and the AWGN are also present in the received signal.

In the considered system model, there is one central FL server and a number N of sensors ($\forall i\ N = \sum_{c \in \mathcal{C}^i} N_c^i$) used as FL nodes located in a given area, as shown in Figure 4.5. The FL nodes are assumed to be able to perform ideal SS. This ability makes the collection of labeled data possible. The FL nodes are distributed in the considered area, so the data they collect are diverse and represent wireless channels with different propagation conditions. In Figure 4.5, one can observe an example SNR heat map that represents the distribution of the actual SNR value of the signal originating from the PU. The variety of the SNR values, which change from -30 to 20 dB, is due to the terrain changes.

The size of the area is intentionally expressed in normalized units (i.e., distance units) to make the analysis more generic. The appropriate granularity of the SNR map will need to be adjusted depending on the transmit power and true changes of the SNR value. Thanks to such an approach, the experiment analysis can easily be adjusted to various 5G scenarios, mainly for classic cellular mass traffic and IoT

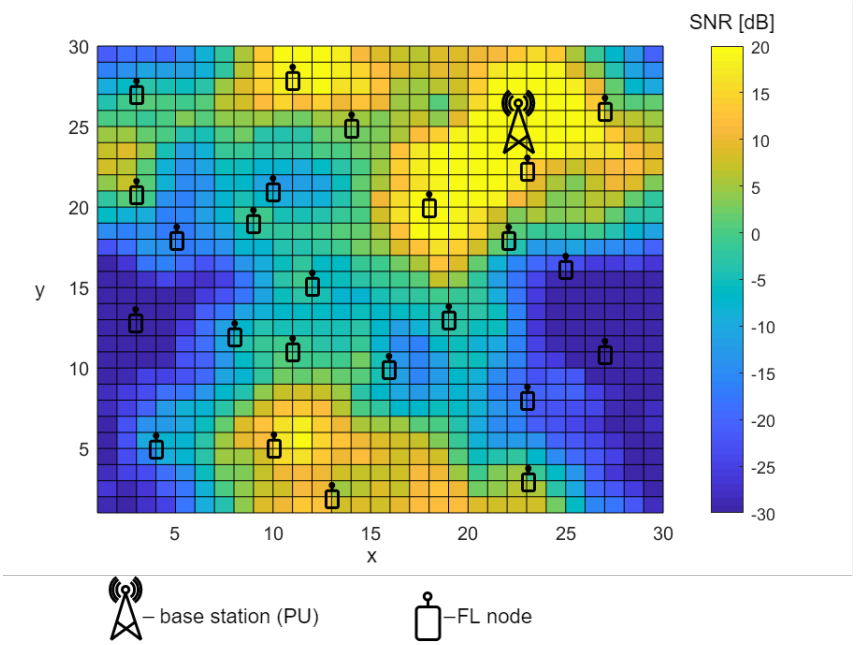


FIGURE 4.5: A map of mean SNR values in space, including FL nodes and a signal source-a PU.

applications.

When an FL node creates its own CNN model, it is well-fitted to the local data collected by this node. The author postulates using the FL method to take advantage of multiple differently trained models and create new ML models that will be more general in their application but still specialize in SS for some specific channel conditions. It is assumed that the SUs do not participate directly in the FL process but rather take advantage of the ready CNN models generated as a result of the FL algorithm. It seems a good solution for SUs with limited computational resources, as SUs do not have to perform any ML training.

4.3.1 Simulation Setup

The experiment focused on sensing the 5G downlink signal. The tested signals have a 10 MHz bandwidth containing 50 RBs. Each of the RBs comprises 12 OFDM subcarriers. Each of the RBs lasts 0.5 ms and consists of 7 OFDM symbols. Full synchronization is assumed. The energy, frequency, and time information collection occur during the first OFDM symbol only, which leaves time for a decision on the occupancy of considered RB and the SU opportunistic use of the resources, i.e., transmission of signals, preferably orthogonal to the PU's signal. A periodicity is present in the PU signal due to daily intensity oscillation. The oscillation occurs every 80 slots in time to simplify the simulations.

A four-layer CNN model has been deployed. The input images consist of 50 pixels vertically and 100 pixels horizontally and have three layers. The activation function in all of the layers is the rectified linear activation function except for the last layer, where the softmax function is used as an activator. As an optimizer, the Adam optimizer is applied with a learning rate of 0.0001. The loss value is calculated using Sparse Categorical Crossentropy. As a result, the CNN returns two vectors of length 50. The vectors represent the occupancy probability of all RBs in a frequency band for a given moment. Each FL sensor creates a CNN model

like this and trains it on its collected data. CNN models of the same structure are created as a product of the FL procedure. The SU receives a ready CNN model that is adequate for its location. The data collected by the SU is of the same type as training data. Figure 4.6 shows the proposed CNN algorithm's model details.

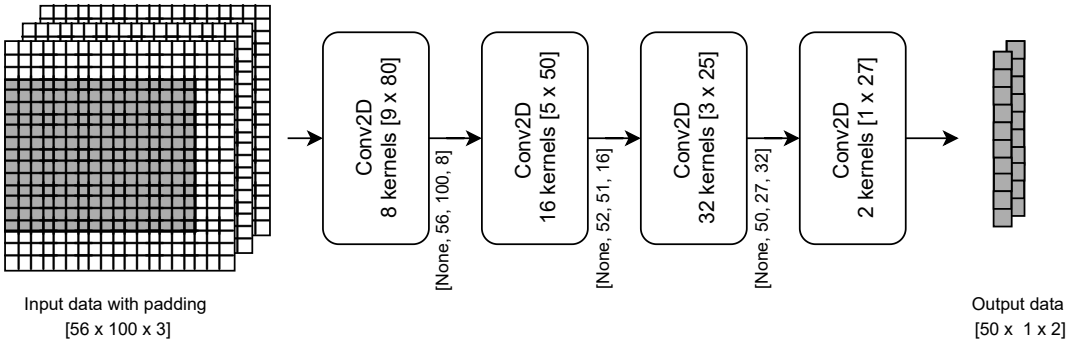


FIGURE 4.6: CNN algorithm model.

The FL sensors collect data of length 800 slots in one iteration step. The fading effect is still in time as the FL sensors are immobile. The environment is assumed to be quite dynamic, and the fading is different in each FL algorithm iteration. This assumption equals a situation where, in each iteration, different FL sensors are picked with a different fading channel but with the same mean SNR as sensors from the previous iteration. It is assumed that the mean SNR values oscillate around values constant in time and characteristic of a given location. In the simulation, the author chose 24 sensor locations randomly. The mean SNR values for those sensors are presented in Table 4.1.

TABLE 4.1: Mean SNR [dB] of each FL sensor.

sensor 1	**sensor 2**	**sensor 3**	**sensor 4**	**sensor 5**	**sensor 6**
−30	−30	−22.9	−20.6	−17.04	−13
sensor 7	**sensor 8**	**sensor 9**	**sensor 10**	**sensor 11**	**sensor 12**
−11.8	−7.6	−7.3	−5.8	−3.4	−2.9
sensor 13	**sensor 14**	**sensor 15**	**sensor 16**	**sensor 17**	**sensor 18**
−1.8	−0.9	0.8	2.6	4.8	4.8
sensor 19	**sensor 20**	**sensor 21**	**sensor 22**	**sensor 23**	**sensor 24**
5.9	7	9.7	17.6	19.8	20

In the simulation, k-means clustering categorizes FL sensors into groups of similar SNR values. Eight clusters are used in the experiment. Table 4.2 presents an example of how the sensors were clustered in one of the FL algorithm iterations.

Table 4.2: Mean SNR (dB) of FL sensors in each cluster.

cluster 1	−30	−30	−22.9		
cluster 2	−20.6	−17.04			
cluster 3	−13	−11.8			
cluster 4	−7.6	−7.3	−5.8		
cluster 5	−3.4	−2.9	−1.8	−0.9	
cluster 6	0.8	2.6			
cluster 7	4.8	4.8	5.9		
cluster 8	7	9.7	17.6	19.8	20

In the testing phase, signals were generated with different fading channels and combinations of SNR values. The SNR values are within the range $[-24, -23, -22, \ldots, 16]$ dB . The tests are performed for all iterations from the range $[1, 2, 3, \ldots, 30]$. The FL server chooses the best CNN model created by clustered FL sensors for each iteration. The SUs choose the best model' and clusters' mean SNR comparison.

Matlab software was used to simulate signals and the channel. The data for training and testing were also generated using Matlab. The TensorFlow Python library was applied to create and use CNN models in the FL algorithm.

4.3.2 Simulation Results

Two probability measures were derived to evaluate the results, namely the probability of detection P_{d} and the probability of false alarm P_{fa}. As the first set of results, an example of how P_{d} and P_{fa} of SU detection change with each iteration for each cluster is presented. The results are included in Figure 4.7. Each plot was obtained by SU for all of the integer SNR values from a corresponding range included in the plot title.

Each plot is titled with the cluster index and SU SNR range associated with this cluster. It can be observed that for clusters with relatively low SNR values, the change in results can be pretty significant between each of the iterations. For example, P_{d} results for cluster 1 tend to switch between values around 50% and around 15%, which correspond with higher and lower P_{fa} values. The high diversity of results for low SNR values can be explained by the high sensitivity of the CNN models for changes when the signal is so weak compared to noise. Clusters 7 and 8, which refer to high SNR values, show the most minor variety of results. These clusters were created using CNN models with similar and relatively high detection quality, so changes in the fading channel between iterations do not significantly impact the results. Another interesting observation is that for the first two clusters, all P_{d} curves are very similar to each other. The same applies to P_{fa} plots. Starting with the third cluster, up to the fifth, it is visible that those plots began to differ significantly, though the downward and upward trends of the curves remain the same. The results are similar again for clusters 6, 7, and 8.

Figure 4.7 presents one exemplary FL algorithm run. Figure 4.8 shows how the algorithm behaves on average. It includes mean results of P_{d} and P_{fa} results for each cluster. There, P_{d} was calculated as an average of estimated P_{d} results of several

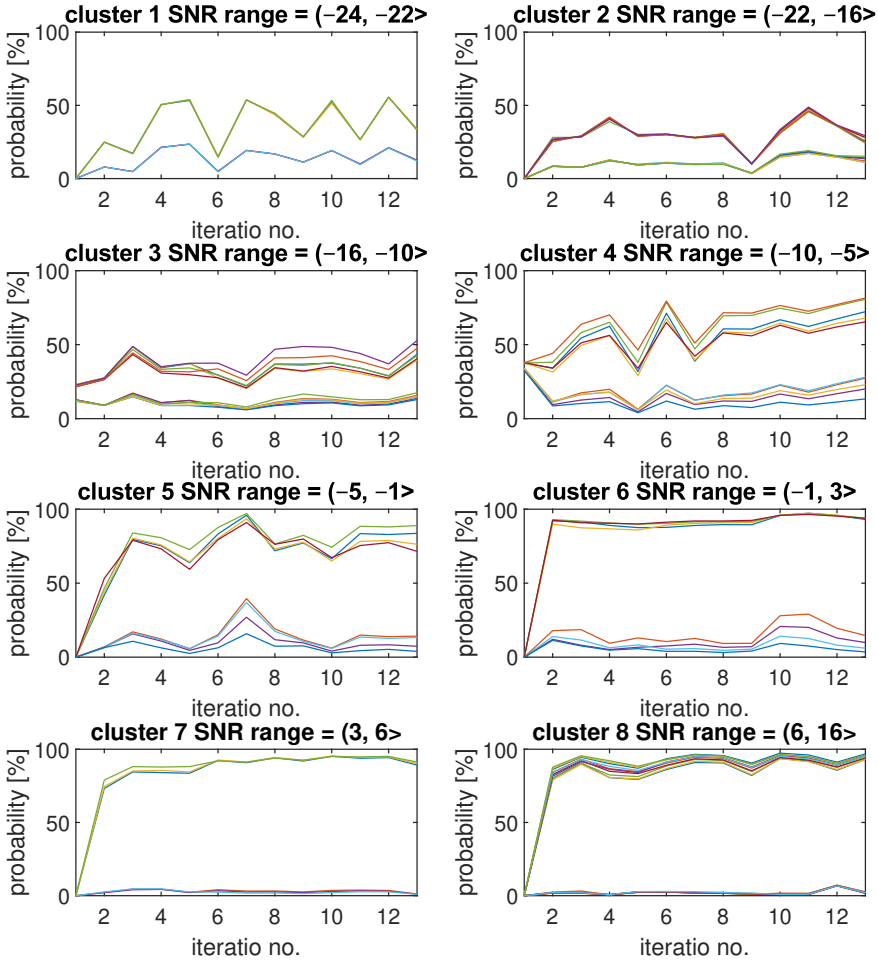


FIGURE 4.7: Exemplary changes of P_{d} (upper set of curves) and P_{fa} (lower set of curves) for each FL algorithm iteration and for 8 different clusters.

simulations and all integer SNR values (in dB). The same method was applied for P_{fa} results. It can be observed that, on average, the results are quite steady after the second iteration. It is also visible that P_{d} values do not necessarily grow with growing cluster numbers. For example, P_{d} is around 40% for the first cluster, and it reaches lower values (around 30%) for the second cluster.

Finally, to evaluate the FL-based SS results against the individual CNN-based SS, P_{d} and P_{fa} were calculated for the same SU data, but employing separate CNNs that specialize in every SNR value from a considered range. Figure 4.9 presents P_{d} and P_{fa} results for the final iteration of FL algorithm and P_{d} and P_{fa} for individual CNN-based sensing. To improve the analysis of the final results, the plot was divided into separate sections marked by different colors. Each section represents a range of SNR values categorized into a different cluster.

The main conspicuous difference is that the CNN sensing probability plots are smooth and consistently change values with growing SNR without excessive fluctuations. On the other hand, the FL sensing results are diverse and depend a lot on a cluster and the range of SNR values of that cluster. For example, results for the first two clusters are quite stable. For clusters 3, 4, 5, and 6, results are quite various,

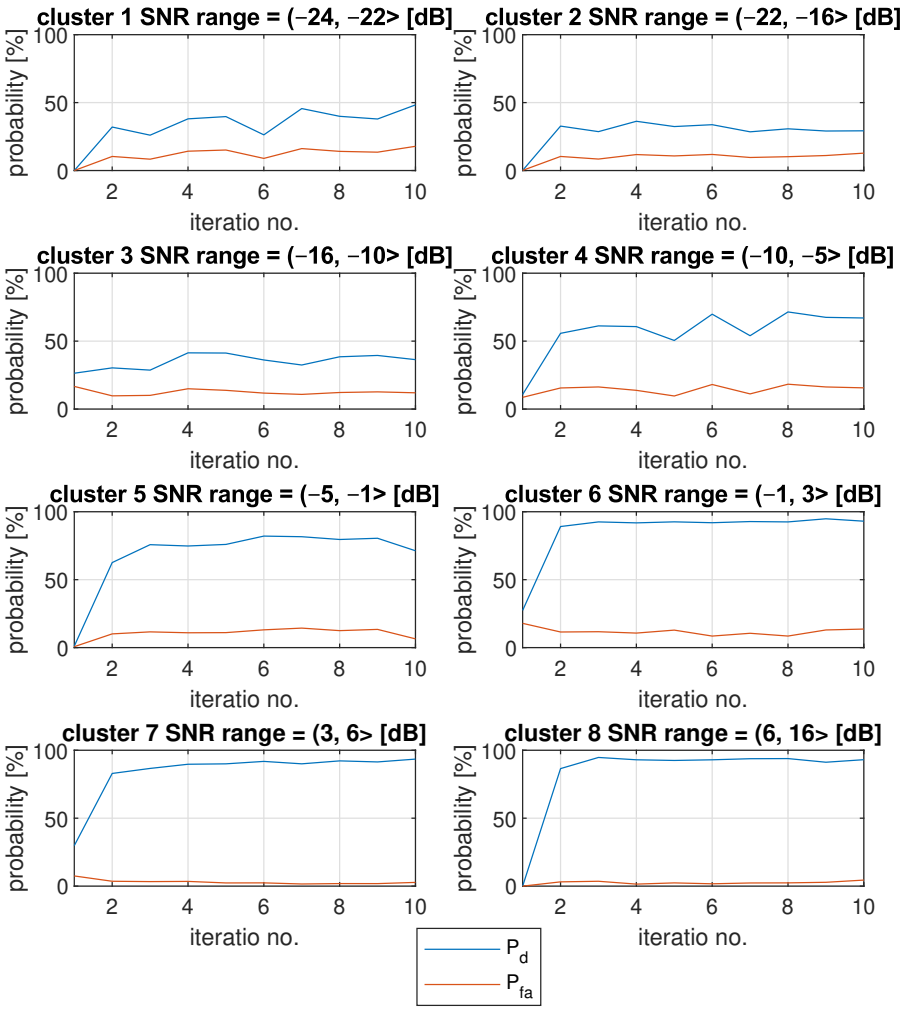


FIGURE 4.8: Mean changes of P_{d} and P_{fa} for each FL algorithm iteration and for 8 different clusters.

which could already be seen by analyzing Figure 4.7. These clusters represent SNR ranges, for which P_{d} results grow fast, and P_{fa} results begin to decline. It causes more imbalance in the results, as there is a need for more specialized CNN models for each SNR value. For the last 3 clusters, the results are again more stable due to less variability in the expected results. The FL results show clearly that the clustering of the CNN models has an impact that is not negligible. The border SNR values between clusters are quite visible, as they often correlate with a sudden change in P_{d} and P_{fa} results. Despite the variability in the results, the overall trend, similar to CNN sensing results, is maintained. FL results generally improve for the same SNR values as CNN results. The one quite noticeable drawback of FL sensing is P_{fa} results for the middle clusters. Along with the dynamically changing P_{d} results come higher values of P_{fa}.

The same type of results were generated additionally for 12 clusters. Figure 4.10 shows these results. Now, fewer FL sensors contribute to model generation for each cluster. The consequences are visible, especially for P_{fa} results for the middle SNR range. The fluctuations in the results are more dynamic than in the results for eight clusters, but in general, the SS results are similar in both cases.

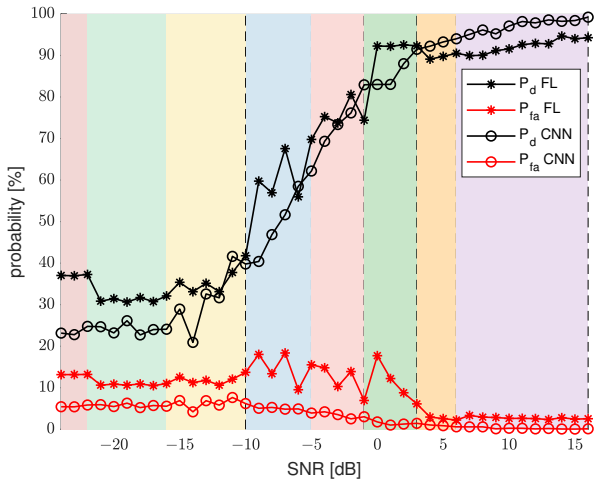


FIGURE 4.9: FL performance for different SNR values, for 8 clusters, compared with CNN results.

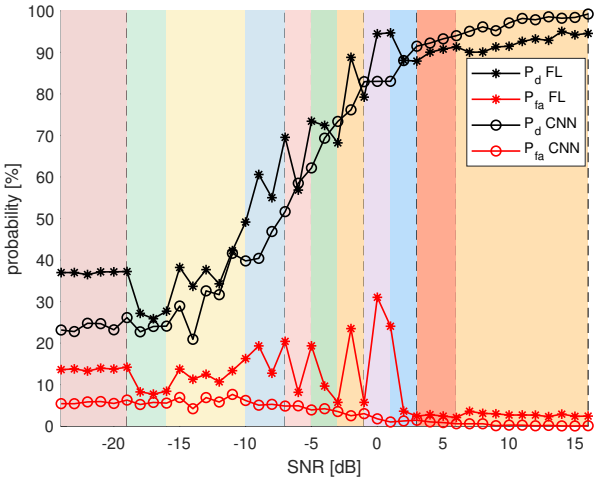


FIGURE 4.10: FL performance for different SNR values for 12 clusters compared with the CNN results.

By analyzing the above figures, one can conclude that, in general, the FL results, albeit a little variable, present a solid alternative to the more channel-fitted SS ML models. The learning process does not need many iterations on different channel conditions and can generate well-working models using data collected by only a few sensors.

4.4 Chapter summary

In this chapter, the author of this thesis discussed the application of FL for SS. The FL method has proven its adaptability, enabling SUs to perform intelligent sensing without the need for extensive data collection and CNN model training. Instead, new incoming SUs can take advantage of earlier created FL models. Despite FL sensors being limited to one mean SNR value, the global models they generate are well-prepared for SNR values from a given range. The FL algorithm proposed by the author requires only a few iterations to achieve satisfactory results in terms

of high probability of detection and low probability of false alarm.

The proposed FL-based SS algorithm employs FL sensors to create not one but multiple ML (CNN) models that enable SS for diverse sensing conditions. (Each model specializes in a separate SNR range.) Its performance is comparable with the performance of the SS performed by CNN models specialized in a given SNR value. It means that data collected from a few FL sensors are enough to create averaged SS models that can perform SS from the whole continuous range of SNR. Thus, the proposed algorithm also allows for using these FL corporate averaged models by SUs not participating in their generation and offloading these SUs from ML training responsibility.

To summarize, the advantages of the author's proposed FL-based SS over the alternative SS schemes are the following:

1. It results in a higher quality of decisions (expressed in higher P_{d} and similar P_{fa}) than autonomous (by individual sensors) sensing, including classical SS by sensors in adverse locations and ML-base.

2. It allows for spectrum prediction, unlike schemes not incorporating ML, such as classical SS and CSS.

3. It ensures data privacy unlike centralized ML since training data transmission is not required.

4. It allows for building a universal model for all SUs, ready to be used by new incoming users.

FL-based SS limitations hail from communication bottlenecks. The main challenge is decreasing the total number of communication rounds and transmitting minor model updates. Moreover, the UEs participating in the training process may differ regarding storage, computational ability, power supply, and network connectivity capabilities. Therefore, the FL-based SS approach must be resilient to UE failures and work with low participation. Finally, the local dataset cannot be cleaned for missing values and irrelevant aspects.

Chapter 5

Secure Federated Learning for Spectrum Sensing

As discussed in the previous chapter, FL is a suitable approach to SS ([176], [33]), which enables a group of sensors to perform a common learning task by exchanging their local model parameters (or a distilled representative part of a model) instead of raw data to achieve aggregate analysis. Hence, FL is considered a privacy-by-design technique while achieving high learning accuracy [30]. However, FL applied to wireless systems, including SS, does not guarantee the levels of security required by modern communication systems. The FL systems are vulnerable to attacks that target each stage of training and decision-making. Attackers can exploit flaws in FL systems in various ways, such as by corrupting training data or local model updates at CR UE or intercepting the model updates exchanged with the central server [18]. Thus, although FL improves privacy, privacy is not guaranteed without additional protection [106].

In this chapter, the author focuses on data poisoning attacks launched against the UEs' local training data, specifically, on *label-flipping*, when an adversary changes the training data labels, causing the model to misclassify them during training [183]. She presents her original contribution to the design of detection and mitigation of such poisoning attacks in FL-based SS. In Section 5.1, the author reviews and analyses the security aspects of the FL-based SS. This section is an original analysis of possible attacks and countermeasures. Next, in Section 5.2 the author presents related works and discusses how her contribution differs from solutions presented in the existing literature. In Section 5.3, poisoning attacks targeted to FL-based SS are considered. In Section 5.4, details of the new algorithm proposed by the author for attack detection and mitigation are presented. This is followed by the description of the computer simulation experiments and their results in Section 5.5. Research results and key findings are summarized in Section 5.6.

5.1 FL-based Spectrum Sensing Security

Recent reviews on FL for CR ([105], [176], [33], [30], [121]), mainly focus on general-purpose FL algorithm security ([18], [106], [133], [19], [53], [73], [111]) and ML-based resource sharing and wireless security ([174], [27]). Some research papers focus on specific threats and countermeasures (e.g., [142] on FL SS robustness to poisoning attacks). Unlike the aforementioned papers, below, in this section, the author presents her view on reliable and secure FL-based SS. She discusses the

application of FL for SS in radio access networks, reviews possible security and privacy attacks on these algorithms, and suggests suitable countermeasures to such attacks.

FL-based SS involves sensing by individual CRs and ML on these devices and creating a corporate model. Therefore, the attack surface for FL-based SS includes typical attacks on sensing, attacks on individual ML algorithms, and attacks on FL.

5.1.1 Attacks on SS in CR

Two classical SS attacks in the context of CR are Primary User Emulation (PUE) attacks and Spectrum Sensing Data Falsification (SSDF) attacks. Both types aim to disturb the spectrum observation and users' access to the system [174]. When an attacker sends PU-like signals during the sensing time, it is referred to as a PUE attack and can prevent authorized CR access to the channels. A system's regular operation could be disrupted by malicious attackers or selfish ones wanting to use the spectrum exclusively. PUE attacks can result in bandwidth wastage, Denial of Service (DoS) connection instability, and service degradation. Identifying malicious users is crucial for protection against PUE attacks.

Sending incorrect local SS reports to others, which causes wrong SS decisions, is how an SSDF attack (referred to as a Byzantine attack) is launched in cooperative (also FL-based) SS. Attacks using SSDF are intended to reduce the probability of detection and disrupt the primary system's operations. They may also aim to increase the probability of false alarms to prevent access to spectral opportunities. Three categories of SSDF attackers can be identified:

1. A selfish SSDF (which aims to secure exclusive access to the target spectrum by deceptively reporting high PU activity)
2. An interference SSDF (deceptively reporting low PU activity to cause a CR to interfere with the PU and other CR secondary users)
3. A confusing SSDF (randomly reporting true or false results on PU activity to prevent CRs from reaching a consensus on the spectrum. occupation.

A Generative Adversarial Network (GAN) is an approach to generative modeling using DL methods that can create fake examples statistically representative of training data without having access to the client's confidential data (FL-nodes). Their operation is based on two sub-models: the generator model, which is trained to create new examples (which could potentially be considered as belonging to the original dataset), and the discriminator model, which tries to categorize these examples as either real (from the domain) or fake (generated). The two models are trained together in a zero-sum game until the discriminator model is tricked about half the time, meaning the generator model generates plausible samples [18], [133]. Thus, GANs can be considered an intelligent method of PUE or SSDF.

Most contemporary defense strategies against attacks on SS make direct judgments based on the most recent data on SS and the reputation of the sensors [174].

5.1.2 Attacks on ML

On the one hand, ML can help manage the CR network operation (e.g., by streamlining SS in the considered area); on the other, it opens the access network to a

new kind of attack. The possible risks brought on by the use of ML in communication networks may be roughly categorized into two groups: ML, which is used to develop sophisticated assaults, and ML, a target for attacks aimed at lowering the security and efficiency of ML algorithms used in operating networks. The latter type of attack, which is the main focus of this chapter, is designed to cause ML systems to learn wrong models, make erroneous decisions, make false predictions, or reveal confidential information. Attacks of this kind can be exploratory if they target the inference phase and are causal if they target the learning phase (training, model development). Depending on whether the attacker has complete, partial, or no knowledge of the training data, the training method, and its parameters, these attacks can be run in a white-box, gray-box, or black-box setting. The main types of attacks on ML algorithms are:

- Poisoning attacks
- Evasion attacks
- Inference attacks [133]

A study of techniques used to deceive or mislead an ML model is called Adversarial Machine Learning (AML). AML can be used to attack or crash an ML system. It can also defend against sophisticated adversaries that utilize AI/ML algorithms to damage a system [142].

Poisoning Attacks

Poisoning attacks aim to influence the learning outcomes by manipulating the data or the learning algorithm in the model development phase (i.e., the training phase). The need for new model learning based on new data makes this attack attractive to attackers because it offers them a chance to influence the trained model through data injection, data manipulation, and logic corruption (a corruption of an algorithm or its learning logic).

Evasion Attacks

An evasion attack is aimed at the inference stage based on the previously learned model. The attacker tries to bypass the model in the test phase by introducing small perturbations in the input values. An example of an evasion attack is generating a signal that mimics the transmission of an authorized user at the authentication stage.

Inference Attacks

As a service in modern networks, ML algorithms are susceptible to new attacks via Application Programming Interfaces (APIs). These kinds of attacks are called inference attacks (also called reverse engineering) and include:

- Model inversion attacks
- Model extraction attacks
- Membership inference attacks

An inversion attack aims to recover training data or their labels using the ML algorithm results. A model extraction (stealing) attack focuses on constructing a stolen (or surrogate) model replicating the functionality or performance of the victim model. The stolen model may have a different architecture than the victim model. It is based on observing the prediction results or the time of its implementation. A membership inference attack determines whether a sample was used to train a target model by observing the model's results.

5.1.3 Attacks on FL

Apart from the typical threats on ML, specific attacks can be observed in FL due to communication and collective operations to create an aggregated model. Moreover, attackers can take advantage of FL 's design benefit: local privacy that prevents the FL server from seeing the agents' local data or training procedures. On the other hand, the collective operation of the legitimate CRs participating in FL may dominate the FL model creation process and prevent malicious nodes from negatively impacting the model. Below is an overview of the attacks specific to FL-based sensing.

Data Poisoning Attacks

Poisoning attacks targeting a subset of FL nodes can be launched as local models and re-trained with freshly collected data. An adversary may covertly affect the local training datasets to control the corporate model's outcome by embedding a well-crafted sample to data-pollute the FL process. A particular case of this attack in FL-based sensing is an SSDF attack. In this case, the training data is falsified to reflect high, low, or random PU activity and impact the spectrum occupancy model. This kind of poisoning attack is called Data Poisoning. Data poisoning attacks can be divided into clean- and dirty-label attacks [183]. The clean-label attacks modify the training samples while the data labels remain unchanged. Conversely, dirty-label attacks modify the labels of the training data set. A typical example of a dirty-label attack is *label-flipping*. In a label-flipping attack, an adversary changes the training data labels, causing the model to misclassify them during training.

Model Poisoning Attacks

Model poisoning is related to Byzantine attacks where hostile agents can send arbitrary gradient updates to the FL server. In these situations, the adversarial objective is to induce a distributed implementation of the stochastic gradient descent algorithm to converge to completely ineffective or suboptimal models. The vulnerability of FL to adversaries that exploit the privacy these models are supposed to provide is investigated in [19].

Inference Attacks

By examining locally computed updates, inference attacks can extract meaningful information about the training dataset or the model itself. The types of inference attacks to which FL is vulnerable include the ones described above for ML [18], [53]. However, in the case of FL, they may be launched against FL nodes (Cognitive Radios - CRs) or the FL server.

Communication Attacks

Usually, in FL, many communication messages must be sent back and forth between the FL server and each FL node over the iterative learning process to reach convergence. Therefore, a non-secure communication route (usually wireless) is vulnerable to communication attacks. For example, a Man-in-the-Middle (MITM) attack can alter the exchanged messages. A DoS or signaling storm attack would aim to occupy radio resources in the control channels by massive system access requests. A PUE attack can also be considered a communication attack since it is based on transmitting a fake PU signal. Moreover, inference attacks are partially based on eavesdropping on the globally shared model parameters, thus requiring the decoding of encrypted messages. Communication constraints (e.g., limited bandwidth or radio resources) can also undermine the FL system.

Freeriding/Spoofing Attacks

Another group of attacks involves creating fictitious local updates (e.g., using GANs) to obtain the shared global model without participating in the FL process. The primary reasons for submitting false updates in free-riding (spoofing) attacks are to conserve local computing resources, compensate for the lack of necessary data, or avoid violating data privacy laws so that local data are unavailable for model training.

An illustration of the above-described attacks on FL-based SS is presented in Figure 5.1

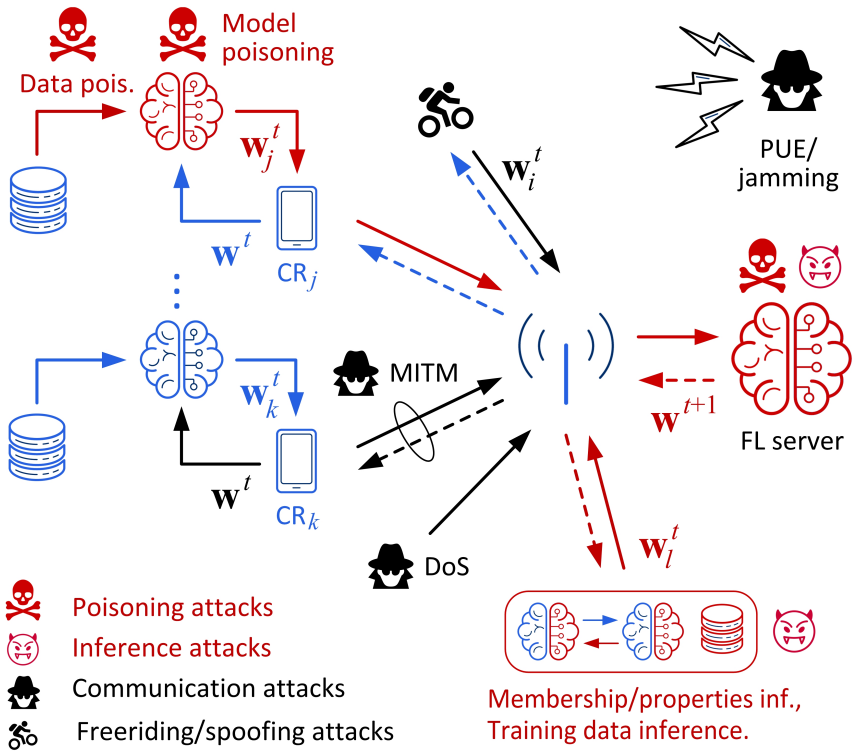


FIGURE 5.1: FL-based SS security attack (marked in red) and countermeasure (marked in blue) classification.

5.1.4 FL-based SS Security Measures

To improve the resistance of FL techniques to adversarial attacks, an assessment of their vulnerability is needed first, followed by applying the appropriate defense measures. Several techniques can prevent these attacks, which are briefly discussed below. Moreover, several methods have been developed to counteract communication and spoofing attacks in radio access networks. It should be emphasized, however, that existing defense mechanisms that are resilient to attacks reviewed in the previous section are still imperfect; they cannot fully protect FL-based SS methods for CR. The taxonomy of these attacks and countermeasures is provided in Fig. 5.2.

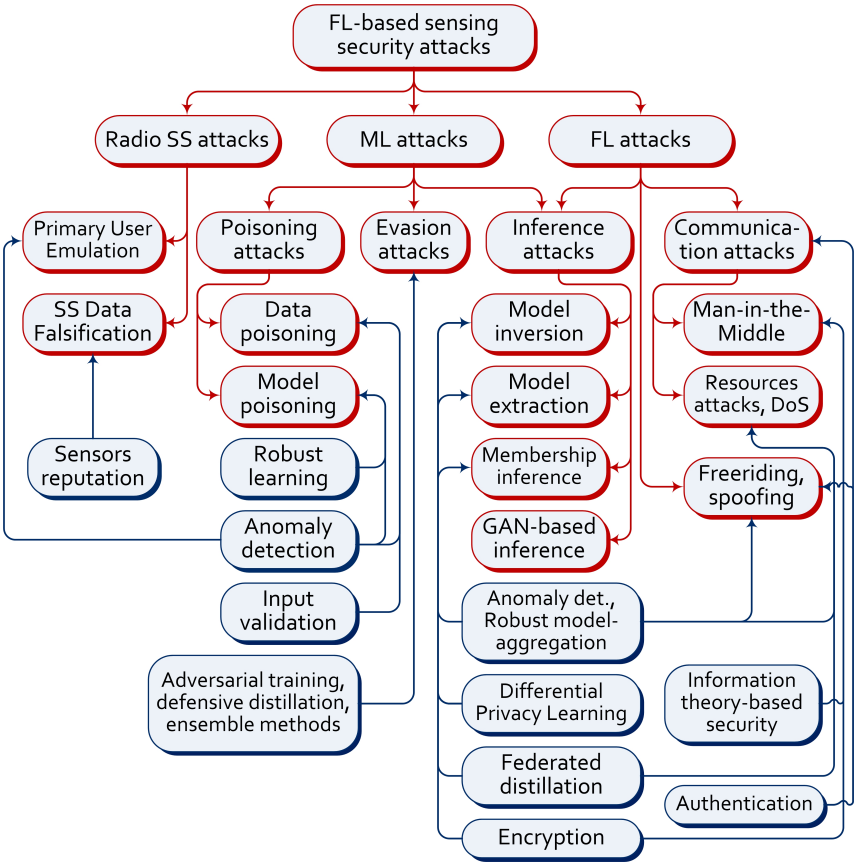


FIGURE 5.2: Illustration of FL attacks in SS.

Defense against poisoning attacks

Defenses against poisoning attacks (including SSDF) have been put forth and surveyed in many research papers (e.g., [133]). They can be roughly divided into input validation and robust learning. Before feeding the data into the ML model, input validation aims to clean the training (and retraining) data from malicious and anomalous samples. For instance, the reject on negative impact technique cleans data by eliminating examples that negatively affect learning outcomes [73]. The relevant methods first create several micro-models trained on a disjoint fraction of input samples to accomplish data cleaning. The anomalous training data subsets are then omitted by combining the micro-models in a majority voting process. In contrast to input validation, robust learning uses robust statistics to create learning algorithms resistant to contaminated training data [111].

Defense against evasion attacks

Defense against evasion attacks includes adversarial training (training the model on a dataset augmented with adversarial examples), defensive distillation (training using the knowledge inferred from an ML model to strengthen its robustness), ensemble methods (combining multiple models to build a robust model), defensive GANs, and techniques to counteract the detuning of the model. By projecting input samples onto the range of the GAN's generator before feeding them into the ML model, defense GANs seek to clean them from adversarial perturbations. In other words, they seek to identify the sample the GAN's generator can produce most similar to the adversarial example and send it as input to the ML model.

Defenses against inference attacks

Defenses against inference attacks related to the theft of the ML model and APIs of ML algorithms include the following methods:

- Learning with Differential Privacy (DP) to prevent disclosure of training data by making the model prediction independent of a single input
- Homomorphic encryption, which enables the model to be trained on encrypted data, thus ensuring data privacy
- Limitation of sensitive information available through the API of the ML algorithm

DP in FL-based SS aims to ensure that no sensing record in a given FL-node dataset can be meaningfully distinguished from the other records in a highly likely scenario. The primary method of this technique would be to add noise to the sensed PU's time-frequency RB occupation or the detected energy before exchanging individual updates with the FL server. The statistical data quality loss caused by the noise introduced by each FL node should be negligible in comparison to the strengthened data privacy protection. Given the required quality of SS for the efficient operation of CRs, the DP may not be practical for FL-based SS.

The model compression technique called federated distillation is a method whereby a globally shared model that has received the necessary training gradually imparts the essential knowledge to a local model. The concept of disseminating information alone rather than model parameters may be used to increase security while lowering communication costs and computation overhead.

Defenses against communication attacks

Defenses against communication attacks, particularly in radio access networks, have been a research focus for many years. This is because of the open nature of the radio communication medium. A recent survey of relevant attacks and defenses in radio access networks is found in [27]. The critical defense strategies against jamming, spoofing, eavesdropping, altering communication messages, and DoS attacks are the following: authorization and authentication procedures, data encryption, information-theory-based security algorithms, and anomaly detection mechanisms.

Anomaly detection methods

Anomaly detection methods can detect events that deviate from a typical pattern or activity by fundamental analysis and statistical analysis. Anomaly detection algorithms can spot problematic clients in FL environments to detect poisoning attacks, free riders, PUE, jamming, DoS-type attacks, or incorrect model updates. If these anomalies can be identified, they can also be eliminated in some cases.

In this chapter of the thesis, the author presents her new method for detecting anomalous local CNN models for SS that are used in FL subject to poisoning attacks. Anomaly detection for attacked FL-based SS has a great potential what is illustrated in Figure 5.3. This figure presents P_{d} and P_{fa} vs. the iteration number of an FL-based sensing algorithm. In this example simulation scenario, the PU signal is a 5G downlink transmission consisting of 5000 patterns in the form of 50 x 100 RBs in frequency and time (5000 RBs per pattern). The FL model is built based on three FL nodes: two eligible ones and an attacker poisoning the training data by incorrect labeling (50 % of RBs are labeled as occupied). Moreover, one additional SU node, the tester node, tests the corporate model, although it does not participate in its creation. All nodes are characterized by randomly generated EVA channels, random Doppler frequency in the 30–55 Hz range, and a mean SNR of 10 dB. The security algorithm implemented in the FL server detects the model update anomalies using its energy-sensing dataset to test the received models. Suppose the decisions on the spectrum occupation using a particular model do not exceed a set percentage threshold of accordance with the decisions using other models. In this case, the model is rejected; that is, it does not participate in creating a corporate model. In Figure 5.3, the solid lines represent the fully protected FL-based SS resulting from adopting a relatively high decisions accordance threshold of 65 %. The results represented by the dashed lines have been obtained for a less protective algorithm with a threshold of 55 %. Note that the less protective algorithm accepted the attacker's model in iterations 12, 13, 16, and 17, which increased P_{fa}. Thus, the spectrum opportunities would be lost if the secure algorithm did not ban the attacker's model. These considerations motivated the author to design a more sophisticated but practical algorithm for anomaly detection in the set of SUs models participating in FL-based SS.

5.2 Related works

In Chapter 4, the author discussed the state of the art in FL-based SS. Despite the FL approach avoids transferring large training datasets with high-resolution localization data and provides data privacy measures, it is also vulnerable to attacks threatening both local and global training and inference (such as data poisoning or model poisoning, evasion or inference attacks) or communication between UEs and an FL server (e.g., Man-in-the-Middle, Denial of Service, or jamming). An overview of attacks on FL-based SS has been presented in the previous section. Here, the author focuses on data poisoning attacks launched against the UEs' local training data, specifically, on *label-flipping*, when an adversary changes the training data labels, causing the model to misclassify them during training [183].

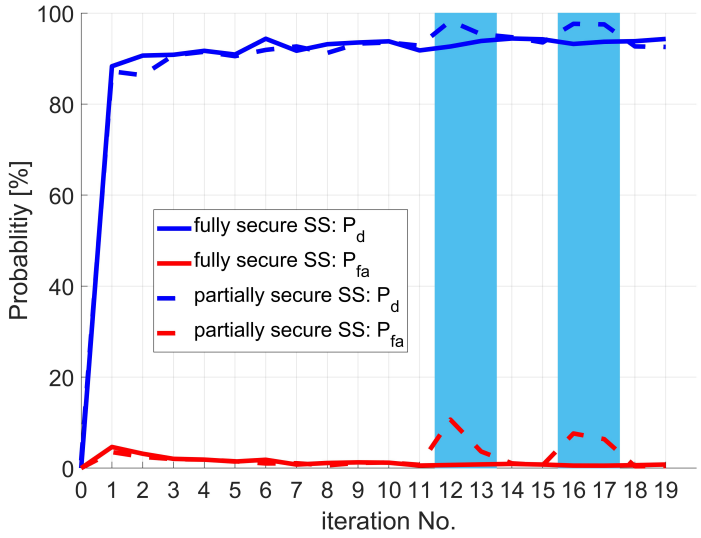


FIGURE 5.3: Secure FL-based SS performance. Attacks marked in light blue in iterations: 12, 13, 16 and 17.

5.2.1 FL under label-flipping attack

The label-flipping attacks have been broadly analyzed in the literature; however, most works abstract from a particular application scenario. Moreover, most papers focus on the impact of label-flipping on ML model performance in general, not necessarily in the FL scenario. The impact of label-flipping attacks on FL performance has been analyzed in [170]. There, the authors examine the global model accuracy in the presence and absence of an attack in the case of 20% of clients contributing to FL being malicious. The global FL model has been created for the classification of the MNIST dataset (popular for testing ML algorithms for text or image recognition) [87]. The results show that the performance of the created models decreases with the amount of poisoned data.

Another work [11] focuses on evaluating the robustness of ML-based malware detectors against different volumes of poisoned data. Similarly, as in [170], the ML models perform worse for higher volumes of poisoned data, which is quite an obvious conclusion. Moreover, the impact of the percentage of poisoned data on the quality of the global model has been examined. The authors of [159] point out that label-flipping attacks can be class-sensitive in the case of multi-class classification models. The attack accuracy varies with the different target classes that are applied. The considered malicious algorithm can find the best target class of the label-flipping attack and, therefore, increase attack effectiveness.

Another interesting analysis of label-flipping attacks on an FL algorithm has been presented in [205] where a label smoothing method gradually changes a global model prediction distribution, further increasing the poisoning attack's negative impact on FL performance. Note that [170], [11], [159], and [205] abstract from or have a different application scenario than the scenario considered in this chapter. Moreover, they do not consider methods to counteract label-flipping attacks.

5.2.2 Label-flipping attacks countermeasures

Several papers explore methods of counteracting the label-flipping attacks. Different anomaly-detection metrics are examined in [135], and a new aggregation

method in the FL server is considered for attack detection. According to the findings of that paper, the proposed *minimum aggregation* method is more effective for label-flipping detection than *FedAvg aggregation* (which is a typical FL model-aggregation method based on calculating an average value of the received local models' parameters [112]), and *geometric median aggregation* (based on averaging geometric median values of models' parameters [99]).

In [167], the authors show that label-flipping attacks can significantly impact the classification of ML models shared by the clients (users). This classification aims at detecting malicious clients. The authors propose a method to detect malicious users by employing a dimensionality reduction method (that extracts the most relevant parameters from the delivered models while still capturing the original models' meaningful properties).

Paper [90] extends the label-flipping detection and mitigation method proposed in [167] and improves it by employing a *kernel principal component analysis* [50] and K-means clustering. Note that classifying local models as the ones delivered by genuine and malicious users allows to detect the attackers, identify them, and disregard them in the global model creation. How the global model is created also impacts the FL's robustness against poisoning attacks. These are key conclusions from [135, 167, 90]. However, these papers do not consider the SS application and wireless communication scenario, where the consequences of detecting or not detecting an attack are multifold. Neither do they consider nor exploit the intention of an attacker that translates to his specific actions, which is the focus of attack methods proposed in this chapter.

Very few papers study protection methods against FL attacks in specific scenarios with a particular motivation behind the choice of attackers' actions. The examples of such works are [175], [139], and [108], but these papers do not consider wireless communication scenarios. To the best of the author's knowledge, only a few articles address how attacks affect the effectiveness of FL when applied to SS and how to mitigate these attacks. The authors of [124] and [143] do focus on FL for SS, but the considered attack scenario concerns *SS data falsification* attacks and not label-flipping. (In the SS data falsification attack, the attacker mimics the sensed signal and reports misleading information to the FL server.)

In summary, no prior work tackles the problem of methods to detect and eliminate poisoning attacks based on label-flipping in an FL-based SS application scenario, which is the author's focus in this chapter. Moreover, the author considers the particular intention of attackers that translates to *attack design* to increase interference to operating communication systems or to decrease the efficiency of spectrum utilization randomly or in a coordinated manner. Such a specific attack design is ignored or neglected in the literature.

5.3 Attack design in spectrum sensing

The author considers specifically targeted label-flipping attacks on FL-based SS in this section. The attacks aim to fake RBs occupancy. If the attack aims at a false increase of the spectrum occupancy, SUs refrain from transmitting and thus allow attackers free access to radio resources. Another attack under consideration aims to display increased spectrum opportunities, encouraging SUs to utilize them and create interference in RBs. Such attacks are considered to be either random

(randomly selecting RBs under attack) or coordinated (collectively selecting RBs adjacent to the legitimately occupied resources). The above attackers' motivation and actions translate to a specific *attack design* that is considered.

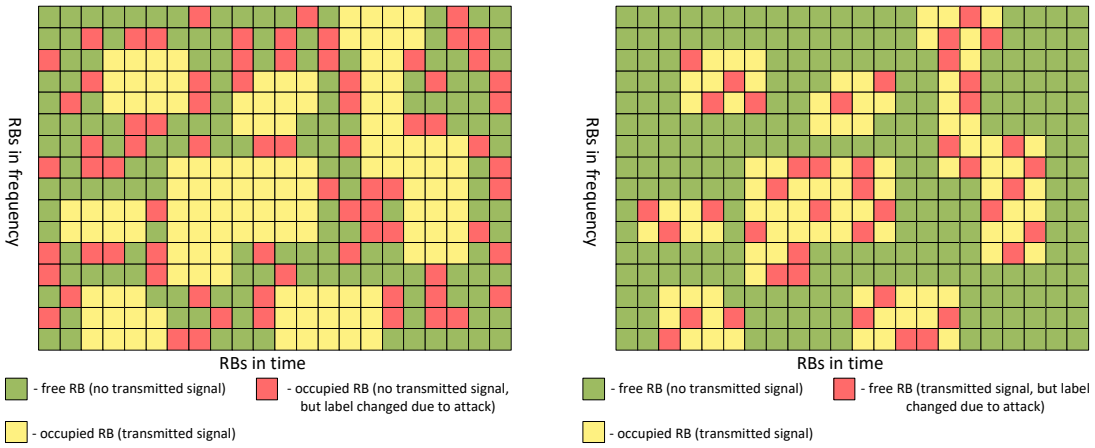
It is assumed that the attackers have passed the authentication process and have access to the local training data labels, like regular UEs. (Unauthenticated UEs would not be allowed to participate in FL.) A correctly labeled training dataset is required in supervised learning and, at the same time, sufficient to launch a label-flipping attack.

Two primary goals of the attack on SS models can be formulated as part of the label-flipping attacks. The first is the apparent increase in the occupancy of the radio spectrum, which is associated with SUs' resignation from using it. This results in the availability of transmission bands for attacking users. The second goal is the apparent decrease in spectrum occupancy. This causes SUs to use the radio resources that appear to them as free even though they are occupied. As a consequence, this will increase the level of interference with PU transmission. Moreover, the author considers these two types of attacks to be either *random* or *coordinated*. The formulation of these four possible attacks is given below.

5.3.1 Random label-flipping attacks

As part of random attacks, the attacking UEs generate training data with labels changed from "resource-free" labels to "resource-occupied" labels for randomly selected resources so that changes applied in the training data of attacking users are not correlated. However, all attackers change the labels in such a way as to achieve a new degree (agreed in advance by them) of spectrum occupancy according to the new labels.

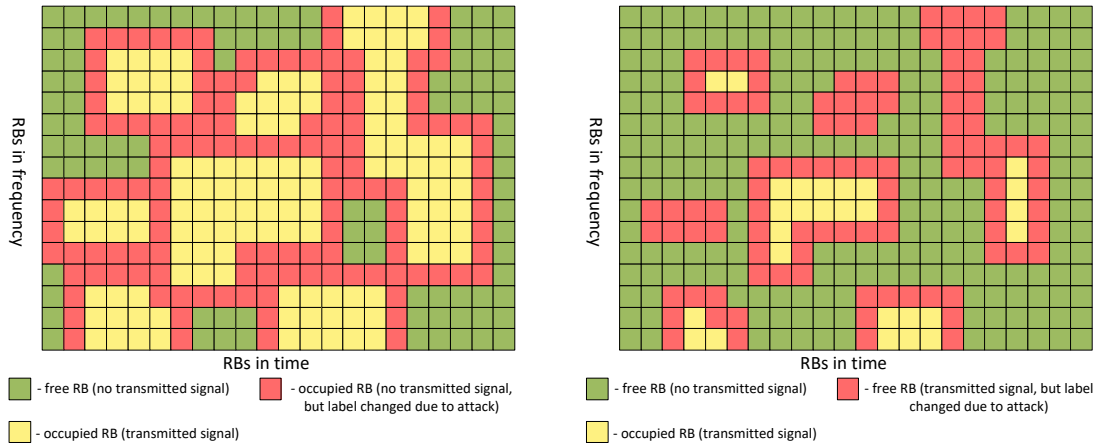
Figure 5.4 shows an exemplary set of labels assigned to RBs in the time and frequency domains, as well as the impact of the random attack method on the labels used by attackers when generating their local models. There, the baseline RBs matrix showing the occupied and free RBs in the absence of a random attack are the yellow RBs in Fig. 5.4.a and green RBs in Fig. 5.4.b respectively.



(A) Random attack aimed at the false increase in RBs occupancy

(B) Random attack aimed at the false decrease in RBs occupancy.

FIGURE 5.4: Labels of an exemplary dataset after random label-flipping attack with different targets.



(A) Coordinated attack (encapsulation) aimed at the false increase in RBs occupancy

(B) Coordinated attack (encapsulation) aimed at the false decrease in RBs occupancy

FIGURE 5.5: Labels of an exemplary dataset after coordinated label-flipping attacks with different targets.

5.3.2 Coordinated label-flipping attacks

In coordinated attacks, all attacking UEs change labels similarly and for the same RBs in the training data. Since the signal to be detected follows a particular standard (5G), an attacker uses this knowledge to modify the training data labels thoughtfully and thus conduct a more effective attack than changing the training labels randomly. Here, an attack algorithm is considered called *encapsulation*, consisting firstly in finding RBs groupings correctly marked as occupied and secondly in changing the labels adjacent to such groupings to increase the area of RBs considered occupied during network training.

This type of attack will be further referred to as *encapsulation* (t_1, t_2, f_1, f_2), where t_1, t_2, f_1, and f_2 refer to the number of RBs in time and in frequency which label has been altered. For example, encapsulation (1, 1, 1, 1) would mean that around a group of RBs that have the actual label "occupied", there is a layer of RBs falsely (and additionally) labeled as "occupied," i.e., for RBs before ($t_1 = 1$), and after ($t_2 = 1$) that group, and for lower ($f_1 = 1$), and higher frequencies ($f_2 = 1$) than those in the truly occupied RBs group. Encapsulation (–1, –1, –1, –1) would mean that inside a group of RBs with the actual label "occupied", a layer of RBs is falsely labeled as "unoccupied."

The attackers agree upon the attack's type and degree of aggressiveness in advance. All users (correctly behaving and the attackers) are assumed to have access to the same training data labels. Thus, attackers have sufficient information about which labels are agreed to be flipped. Figure 5.5 shows an exemplary set of RB labels and the impact of the coordinated attacks on the labels used by attackers when generating their local models. Analogously to Fig. 5.4, the baseline occupied and free RBs matrix in the absence of a coordinated attack are the yellow RBs in Fig. 5.5.a and green RBs in Fig. 5.5.b respectively.

5.4 New protection method against label-flipping attacks on FL-based SS

In this section, the author proposes a new *semi-blind* attack detection and mitigation algorithm in response to the targeted label-flipping attacks presented in the previous section. The *blindness* of an algorithm here means that there is no initial and correct operation phase of the system to detect the genuine UEs before an attack is launched. The proposed *zero-trust* method that continuously monitors, detects, and eliminates an attacker is based on the calculation of carefully selected statistical tests to compare UEs' models (their weights) in pairs and cluster them by k-means algorithm in two groups for "being alike" or not. Then, the decision is made on which group is suspected to contain models of the attackers. These models are rejected and do not participate in the FL global model creation.

Here, the author proposes the following *semi-blind* algorithm to detect the attackers' poisoned models and mitigate their impact on the global model creation in the FL-based SS. It is illustrated in Figure 5.6. The *blindness* of the algorithm means that there is no initial knowledge of which UEs are correctly behaving (genuine). After collecting the CNN models' weights (not to be confused with averaging weights γ_n applied for FL global model creation) from UEs participating in the FL, the first step is to detect *abnormal* models suspected of being attacked.

The author proposes statistical comparison methods on the vectors of models' kernels' weights to detect attacked models and remove their harmful influence on the global model. The idea is to identify "alike" models and those that deviate from them. Comparing the models' weights (e.g., calculating the standard error between relative weights) would not work since the models are trained on different data. Therefore, several statistics have been examined for comparison.

5.4.1 Statistical tests

It has been concluded that the estimated mean, variance, and cumulative distribution function are worth further investigation. Therefore, the proposed algorithm employs three statistical tests: Fisher's [152], Tukey's [169], and Kolmogorov-Smirnov test [127], which reflect the similarity of pairs of models based on their weights' variances, means, and distribution functions, respectively.

Fisher test

Fisher's test for equality of variances tests two populations for the null hypothesis that they have the same variances. This test assumes that the tested populations have a normal distribution. The formula for the test statistic is:

$$F_{n,m} = \frac{\sigma_n^2}{\sigma_m'^2}, \tag{5.1}$$

where σ_n^2 and $\sigma_m'^2$ are the variances of the compared sets of n and m elements respectively (in our case, $n = m$) [152]. To decide on the alternative to the null hypothesis and evaluate its quality the author selects $\sigma_n^2 \geq \sigma_m^2$ in this test, which results in $F_{n,m} \geq 1$.

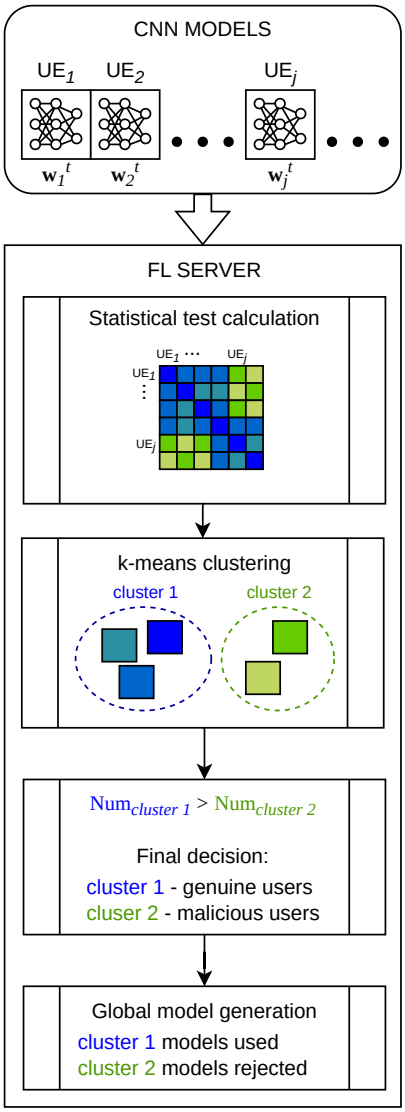


FIGURE 5.6: Label-flipping attackers detection and mitigation algorithm.

Tukey test

Tukey's test compares two sample vectors by comparing their mean values by the following operation [169]:

$$Q_{n,m} = \frac{|\mu_n - \mu'_m|}{\text{SE}}, \tag{5.2}$$

where μ_n and μ'_m are the mean values of the two sets of elements with n and m cardinality, respectively, and SE is a standard error (estimated standard deviation) of the sum of the means. Tukey's test assumption is that the samples are normally distributed, which is not entirely true in our experiments, but this approach still proves useful.

Kolmogorov-Smirnov test

The two-sample Kolmogorov-Smirnov (Kol-Smir) test involves their empirical (cumulative) distribution functions and is given by [127]:

$$D_{n,m} = \sup_x |f_n(x) - f'_m(x)|, \tag{5.3}$$

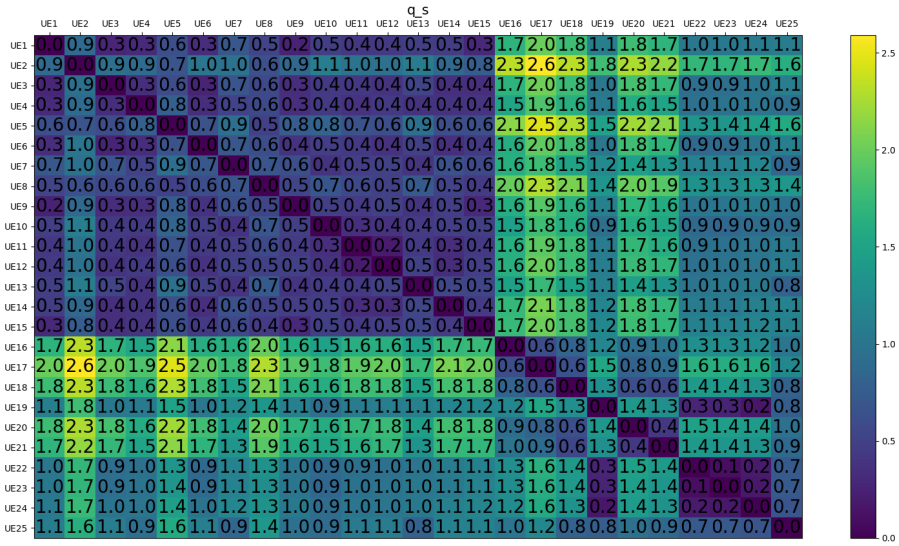


FIGURE 5.7: Example of test values comparing pairs of UE. Test used: Kolmogorov-Smirnov.

where $f_n(x)$ and $f'_m(x)$ are the empirical distribution functions of the compared n-element and m-element set of collected values of random variable x respectively, and *sup* is the supremum.

The example values of the Kolmogorov-Smirnov test for 25 UEs participating in FL-based SS is presented in Figure 5.7. The first 15 UEs sense and report rightfully, and the next 10 UEs attack by coordinated label-flipping aiming at the false increase of the RBs' occupancy. Note that the first 15 UEs' models are "alike" (the Kolmogorov-Smirnov test results have low values, which is exposed in the shades of blue in Fig. 5.7), although some attackers' models (numbered 22 to 24) also display similarities.

5.4.2 Model clustering and decision making

After the chosen tests for all pairs of models are calculated, they are grouped into two clusters using the k-means algorithm. K-means clustering is a popular unsupervised learning algorithm used for data clustering. It groups unlabeled data points (in our case, results of the users' models' similarity test) into clusters (in our case, representing the *alike* models and the outlying ones) to minimize within-cluster variances (squared Euclidean distances). The optimization problem can be approached by iterative *Lloyd's algorithm* [100]. The k-means method was used instead of making hard decisions on the null hypothesis based on comparing test results with thresholds set for a selected significance level, which proved ineffective.

In the proposed application, there are just two clusters. One cluster groups models that are "alike" (represented by the shades of blue in the "k-means clustering" step of the algorithm presented in Fig. 5.6) and the other – the outlying models (represented by the shades of yellow in Fig. 5.6).

The next step is to decide which cluster is larger. It has been assumed that the number of attackers is smaller than that of correctly behaving (genuine) UEs. Suppose this assumption does not hold (e.g., an aggressive malware compromises most UEs); a defense against such a massive attack is possible only if some other assumptions are made, e.g., the genuine UEs are detected before the attack is launched.

Under such an assumption, detecting attackers would be much easier and narrow down to calculating statistical tests against genuine UEs only. However, as mentioned, the author's algorithm is *semi-blind* and does not make this assumption. However, another one was made that the number of attackers is smaller than the genuine UEs.

The last step of the presented label-flipping attack protection algorithm is to reject models suspected of being generated by attackers and omit them when creating a global model. The identified attackers' UEs could also be punished by denying them access to the network.

5.5 Simulation Experiment

As mentioned before, FL for SS is a promising variant of cooperative SS because of its adaptability to changing propagation conditions over time and because it delegates the task of generating ML models to multiple users. It also provides data privacy protection measures. In the scenario illustrated in Figure 5.1, the signal to be detected is a 5G-like signal, primarily characterized by the selection of RBs occurring in the time and frequency domains. The 5G system's RB allocation prevents resource fragmentation, adheres to time-frequency traffic patterns, and is based on channel attributes. Hence, RB allocation and time and frequency fading patterns reveal dependencies that ML can identify. Analogously to image processing, various ML techniques (or structures) can be employed, including CNNs. Predictions of future spectrum occupation can also be made.

The CNN model architecture applied in UEs taking part in the FL algorithm is the same as in the FL server for generating a global model [179] presented already in Chapters 3 and 4. The proposed CNN architecture has been designed to suit the input data best. The subsequent CNN layers' kernels are of a size that can capture the underlying features of the input data. The input histograms have been generated with occupied RBs clustered together. Within a given group of clustered, occupied RBs, the average number of these RBs, both in frequency and time, is equal to 4, while the variance equals 10. The layers of CNNs consist of kernels whose size is enough to capture dependencies between adjacent RBs in time and frequency, whether occupied or free. Table 5.1 presents CNN's structure and input and output data size. The input data is in the form of a spectrogram.

The output consists of two vectors that contain probabilities of the RBs occupation and availability, respectively. The activation function in all layers is the rectified linear activation function except for the last layer, where the softmax function is used as an activator. As an optimizer, the Adam optimizer is applied with a learning rate of 0.0001. The loss value is calculated using *Sparse Categorical Crossentropy* [68].

As shown in Figure 5.1, the model aggregation is based on weighted averaging of the UEs' model kernel weights. Here, equal weights γ_n were assumed, i.e., that no UE is assumed to have a better (more accurate) model than the others. This is because the FL server cannot anticipate which UE experiences better or worse channel conditions and the quality of data that trains the local model. Thus, no UE is prioritized in federated model creation. Apart from the model aggregation, the FL server is also responsible for UEs clustering for integrity. This operation will be discussed in the following sections.

Table 5.1: CNN structure for SS

Input size	56 x 100
Input Layer Con2D	8 kernels [9 x 80]
Hidden Layer 1 Con2D	16 kernels [5 x 50]
Hidden Layer 2 Con2D	32 kernels [3 x 25]
Output Layer Con2D	2 kernels [1 x 27]
Output size	50 x 2

5.5.1 Simulation Setup

This section presents the testing scenario setup and parameters applied to verify the proposed blind protection algorithm against the label-flipping attack in FL-based SS. A system is considered with 25 sensors onboard UEs and one FL server. The sensors detect the signal energy in PU's RBs, where PU is a 5G gNodeB. The PU 5G signal in the 10 MHz band has been simulated. As mentioned previously, the PU RBs occupation is generated with occupied RBs clustered randomly according to the two-dimensional Gaussian distribution. Within a group of occupied RBs, the average number of RBs in frequency and time equals 4, while the variance equals 10. The summary of a distribution of generated clustered RBs is included in Table 5.2.

For simplicity, and without the loss of generality, it is assumed that all UEs have the same Doppler frequency equal to 2.5 Hz. Each of them experiences the influence of a different wireless channel between PU and themselves, and therefore, different frequency-selective fading patterns in the received signal. The same average SNR averaged over 50 RBs in frequency and 800 RBs in time is assumed for all UEs. This results from assumed low-noise power amplification at the receivers' front ends. The 3GPP *Extended Pedestrian A model* [153] has been assumed for simulations.

Label-flipping poisoning attacks and their goals as described in Section 5.3. Table 5.2 presents a summary of the experimental parameters. They have been selected to test various attack scenarios comprehensively. The research was carried out during the operation of the FL algorithm, which was run for 15 iterations indexed from 0 to 14 for various configurations of input parameters. One is the ratio of the number of attacking sensors (K) to the total number of sensors (N). The impact of the moment of launching an attack (in terms of FL iteration number) was also tested. The original spectrum occupancy of the training and test data is 30%. Random attacks aiming at increasing the spectrum occupancy percentage to 55% and 75% and decreasing this percentage to 22% and 16% were investigated. The considered coordinated attacks encapsulating truly occupied RB groups result in the same respective increase or decrease in the apparent spectrum occupancy.

TABLE 5.2: Simulation parameters

System parameters	*Values*
The average number of RBs in time-frequency plane	4×4
Variance of number of RBs in time-frequency matrix	10×10
PU's RBs occupancy	30 %
SNR [dB]	$[0, ..., 20]$
Doppler frequency	2.5 Hz
Number of FL iterations	15
Attacks - parameters	
Random	
- K/N	10/25
- RBs occupancy	16%, 22%, 55%, 75%
- attack-launched iteration no.	5, 10
Coordinated	
- K/N	10/25
- encapsulation [RBs]	(-1, -1, 0, 0), (-1, 0, 0, 0), (1, 1, 1, 1) (2, 2, 2, 2)
- attack-launched iteration no.	5, 10

5.5.2 Simulation Results

This section presents two sets of results to evaluate the proposed algorithm employing different statistical tests. First, the impact of the attacks on the FL-based SS performance is evaluated. The second set of results includes the performance of our proposed attack detection and mitigation algorithm.

Let us stress that the notation in this section and in the remainder of this chapter is changed. This is because now, we deal with two kinds of probabilities of both detection and false alarm. The values used to evaluate the negative effect of an attack are two (estimated) probabilities that entirely describe the FL-based SS algorithm's quality. These are the RBs occupation true positive detection probability denoted as $P_{\mathrm{d}}^{\mathrm{SS}}$ (until now denoted as P_{d}) and the false alarm (false positive) probability denoted as $P_{\mathrm{fa}}^{\mathrm{SS}}$ (previously denoted as P_{d}). Precisely, $P_{\mathrm{d}}^{\mathrm{SS}}$ is the probability of correct identification of occupied RBs as occupied ones. In contrast, $P_{\mathrm{fa}}^{\mathrm{SS}}$ is the probability of incorrect identification of unoccupied RBs as occupied in the FL-based SS algorithm.

Moreover, in this section, performance of the proposed algorithm for detecting the attackers' models is evaluated based on the estimated probabilities of *detection of an attacker* (not the detection of occupied RBs) and related false alarm. Hence, in the remainder of this chapter, $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ will denote the probability of true positive detection of an attacker and the probability of false alarm (false positive) detection of an attacker, respectively.

A. Label-flipping attacks impact on FL-based SS performance

In the experiment, $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ are estimated based on the number of true positive and false positive decisions, respectively, over the total number of occupied RBs. A well-performing SS algorithm maximizes $P_{\mathrm{d}}^{\mathrm{SS}}$ while minimizing $P_{\mathrm{fa}}^{\mathrm{SS}}$. High $P_{\mathrm{d}}^{\mathrm{SS}}$ increases PU protection from interference generated by SUs. Low $P_{\mathrm{fa}}^{\mathrm{SS}}$ increases the spectrum opportunities for SUs.

Note that the label-flipping attacks discussed in Section 5.3 aiming at the false increase in RBs' occupancy should increase $P_{\mathrm{fa}}^{\mathrm{SS}}$ of the global model. (This is examined in subsection 5.5.2 in paragraph "Attacks aimed at the false increase in RBs occupancy".) The attacks aiming at the false decrease in RBs' occupancy (also discussed in Section 5.3) should result in a reduction of $P_{\mathrm{d}}^{\mathrm{SS}}$ of the global model. (This is examined in subsection 5.5.2 in paragraph "Attacks aimed at the false decrease in RBs occupancy".) Note that in all considered cases of the binary classification (discussed in Sections 5.5.2 and 5.5.2) is that increase (or decrease) of false positive decision rate ($P_{\mathrm{fa}}^{\mathrm{SS}}$) always has a side-effect on the increase (or decrease respectively) of true positive decision rate ($P_{\mathrm{d}}^{\mathrm{SS}}$) and vice versa.

The simulation results to evaluate label-flipping attacks' impact on FL-based SS are described below, assuming the parameters from Table 5.2. Attack mitigation techniques are not yet considered in this subsection. The results were obtained by generating global models for all 15 iterations and testing them on the same number of test sets of input data. (This set of test data was sufficient for all considered scenarios.) The presented results are averaged over these tests' results.

Attacks aimed at the false increase in RBs occupancy. The impact of the random and coordinated attacks that aim at the false increase in RBs occupancy on $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ was tested and the results for SNR = 20 dB are presented in Fig. 5.8. The attacks have been simulated in the 5th and 10th iterations of FL to show their impact on the global model created after initial training. The selection of the iteration number of the attack launch is dictated by the need to increase the readability of the resulting plots. Note that after the 4th iteration, the FL algorithm is already achieving a well-adjusted global model resulting in stable low $P_{\mathrm{fa}}^{\mathrm{SS}}$ and high $P_{\mathrm{d}}^{\mathrm{SS}}$. Estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ for the FL system in which attacks do not occur or are eliminated (also called *fully secure* system) are marked in red. The first experiment has been performed for random attacks with spectrum occupancy 55% (green curves in Fig. 5.8). The attack starts and continues from the 10th iteration. An increase by a few percent in $P_{\mathrm{fa}}^{\mathrm{SS}}$ can be observed since that iteration (what is the aim of this kind of attack). Moreover, a slight increase in $P_{\mathrm{d}}^{\mathrm{SS}}$ is observed as an effect of the increase of $P_{\mathrm{fa}}^{\mathrm{SS}}$. For this relatively mild attack, the impact on false alarm growth is small and does not exceed the natural increase in detection level.

Another tested attack that starts at the 10th FL iteration is an encapsulation (1,1,1,1) attack that surrounds the true occupied RBs with one "layer" of RBs that have falsely switched labels to occupied. The encapsulation (1,1,1,1) increases the percentage of labels denoting occupied resources from the original 30% to 55%, corresponding with the previously described random attack, increasing this percentage to the same value. It can be observed that although the overall number of changed labels in this attack is the same for the random attack, the impact on $P_{\mathrm{fa}}^{\mathrm{SS}}$ is different (see the blue lines in Fig. 5.8). An increase of $P_{\mathrm{fa}}^{\mathrm{SS}}$ up to 10% is observed, which indicates that this form of attack is more successful than the $P_{\mathrm{fa}}^{\mathrm{SS}}$ achieved

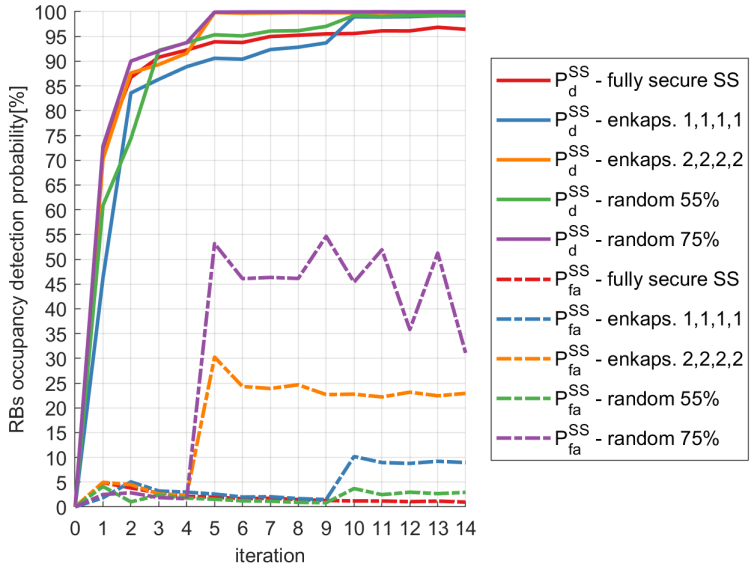


FIGURE 5.8: Estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ for FL-based SS under attacks for SNR = 20 dB vs. the iteration number; Attacks aimed at the false increase in RBs occupancy.

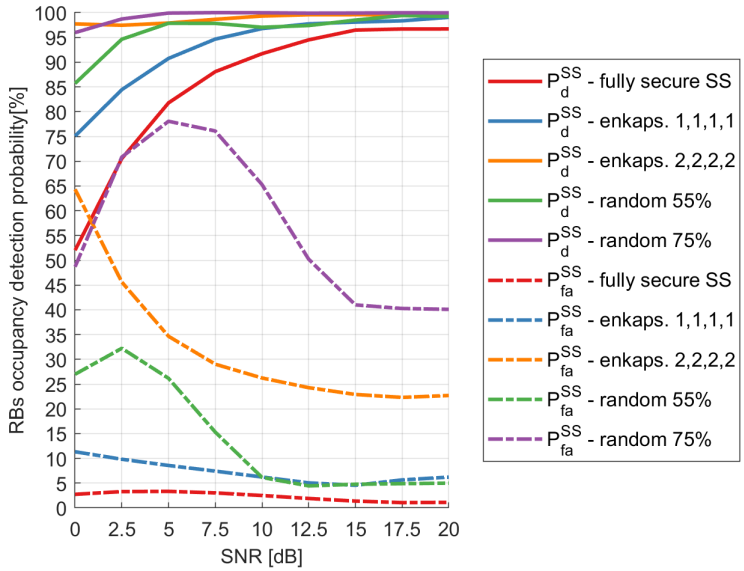


FIGURE 5.9: Estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ for FL-based SS under attacks after the last FL iteration vs. SNR; Attacks aimed at the false increase in RBs occupancy.

by random 55%-occupation attack. This is because an encapsulation attack mimics the true data labels better than a random attack. Labels introduced by a random attack can be easily filtered out by the CNN model used for SS.

A different observation can be made for another set of attacks: the encapsulation (2, 2, 2, 2) and random 75%-occupation attack (yellow and purple curves in Fig. 5.8). Both attacks falsely introduce “occupied" labels that increase the perception of the spectrum occupancy up to around 75%. Also, both of these attacks start in the 5th FL iteration. The encapsulation (2, 2, 2, 2) increases the $P_{\mathrm{fa}}^{\mathrm{SS}}$ to around 25%, i.e., it is more damaging than encapsulation (1, 1, 1, 1). However, the random 75% attack increases $P_{\mathrm{fa}}^{\mathrm{SS}}$ even more, up to 50%. The high spectrum occupancy percentage (of 75%) in this scenario can explain the higher negative impact of the

random attack than the encapsulation attack. The spectrum occupancy percentage reflected in the models reported to the FL server by the attackers is large enough to mimic the true RB occupation even better in the case of a random attack. This causes CNNs to make more incorrect SS decisions.

Figure 5.9 shows estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ versus SNRs for the last (15th) FL iteration. The general observation is that $P_{\mathrm{d}}^{\mathrm{SS}}$ of the attacked systems are even higher than that of the fully secure SS. This is because of the increase of $P_{\mathrm{fa}}^{\mathrm{SS}}$ (as expected, because the considered attack aimed at false increase of the RBs occupation increases $P_{\mathrm{fa}}^{\mathrm{SS}}$). In particular, higher $P_{\mathrm{fa}}^{\mathrm{SS}}$ can be observed for the more aggressive attacks such as encapsulation (2, 2, 2, 2) and random 75%-occupation attack.

Moreover, $P_{\mathrm{fa}}^{\mathrm{SS}}$ in case of a random attack reflecting 75% of RBs occupation is not monotonic vs. SNRs. It is around 50% for SNR = 0 dB because, in this SNR region, the noise effect dominates the attack effect, and deciding on RBs occupancy is random. In the moderate SNR region between 2.5 and 10 dB, $P_{\mathrm{fa}}^{\mathrm{SS}}$ is increased to around 70% because, in this SNR region, random noise has the same effect as the random attack, increasing its negative impact on the false alarm probability. For SNR $\geq$ 15dB, $P_{\mathrm{fa}}^{\mathrm{SS}}$ drops to around 30% because the noise becomes insignificant and does not impact the attack negative performance. Analogous effect on $P_{\mathrm{fa}}^{\mathrm{SS}}$ can be observed for the performance of the other random attack falsely increasing RBs occupation to 55%. This is because the noise in the lower and moderate SNR regions contributes highly to the attack efficacy. It is harder for the CNN model to work correctly in lower SNR conditions. Still, the attacks add another level of difficulty, especially the random attacks that mimic how the channel affects the training data.

Attacks aimed at the false decrease in RBs occupancy. The author has performed similar experiments for the second attack scenario, where the attacking UEs aim to decrease apparent RB occupancy levels to persuade other users to use these resources and increase interference to PUs. As mentioned, this attack should decrease $P_{\mathrm{d}}^{\mathrm{SS}}$. Figure 5.10 shows the results obtained in each FL iteration for SNR = 20 dB. Here, the following attacks have been tested: encapsulation (-1, 0, 0, 0) and random 22%-occupation attack, which both decrease the occupancy level (reflected in the attackers' reported models) to 22%, as well as encapsulation (-1, -1, 0, 0) and random 16%-occupation attack, which both decrease the occupancy level reflected in the attackers' models to 16%. It can be noted that this RBs occupancy percentage change is not as significant as in the previously considered (in subsection 5.5.2) attacking scenario, where the occupancy level has been changed from 30% to 55% or 75% (by 25% and 45% respectively). In the scenario considered in this subsection, the occupancy decrease equals 8% and 14%. This is because the starting occupancy level equal to 30% does not leave much room for spectrum occupancy to decrease. Still, the author examines more scenarios in the research by not choosing the same decrease levels as the increase levels.

Now, the random 16%-occupancy attacks and the encapsulation (-1, -1, 0, 0) attack are considered starting in the 5th FL iteration. The encapsulation (-1, 0, 0, 0) attack and random 22%-occupation attack are launched in the 10th iteration. Note that the encapsulation (-1, -1, 0, 0) attack has a higher impact on decreasing $P_{\mathrm{d}}^{\mathrm{SS}}$ than the random 16%-occupation attack. It can also be observed that the encapsulation (-1, 0, 0, 0) attack and random 22%-occupation attack, as expected, have a lower impact on decreasing $P_{\mathrm{d}}^{\mathrm{SS}}$. The random 22%-occupation attack, which

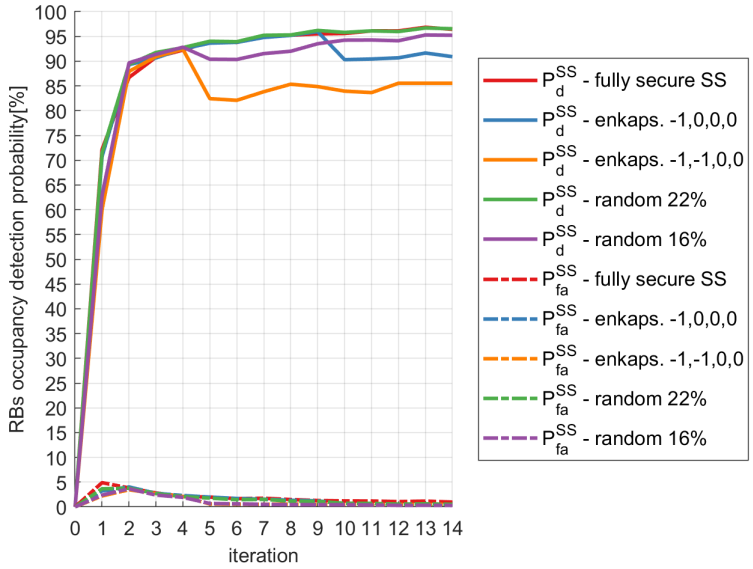


FIGURE 5.10: Estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ for FL-based SS under attacks for SNR = 20 dB vs. the iteration number; Attacks aimed at the false decrease in RBs occupancy.

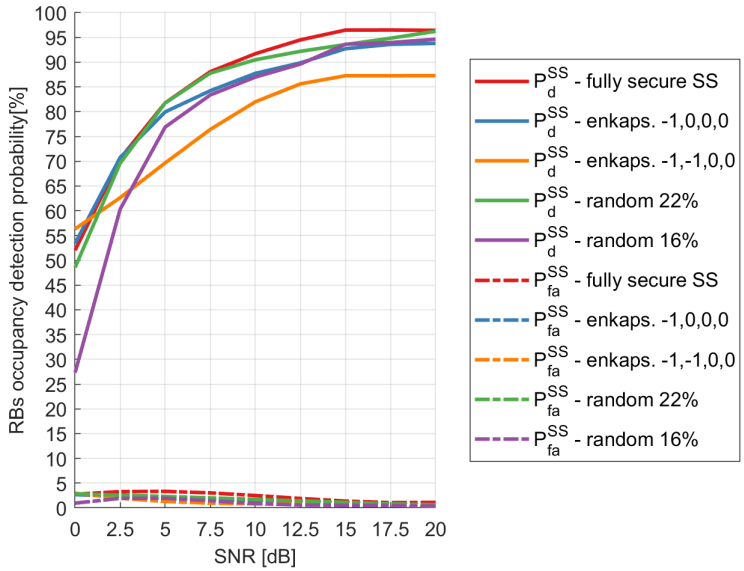


FIGURE 5.11: Estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ for FL-based SS under attacks after the last FL iteration vs. SNR; Attacks aimed at the false decrease in RBs occupancy.

introduces only an 8% change in occupancy compared to the original (not attacked) labels, does not introduce any noticeable changes to $P_{\mathrm{d}}^{\mathrm{SS}}$ nor the $P_{\mathrm{fa}}^{\mathrm{SS}}$. Therefore, the conclusion can be drawn that under random attacks aimed at the false decrease in RBs occupancy, the change in the occupancy levels resulting from label-flipping has to be larger than by 8%. Moreover, the change in this level by 14% is enough to introduce a noticeable decrease in SS performance. For all considered attacks, $P_{\mathrm{fa}}^{\mathrm{SS}}$ varies insignificantly because it is already close to 0%.

Analogically to Figure 5.9, Figure 5.11 presents $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ for the last FL iteration and different SNR values. Here, the results are as anticipated. This is because the examined attacks aiming at the false decrease of RBs occupancy are less aggressive than those performed for the false increase of the apparent RB occupancy

percentage. $P_{\text{fa}}^{\text{SS}}$ is close to 0% for all SNR values and is never larger than 4%. As expected, P_{d}^{SS} values are all lower than those of a fully secured system for all SNRs.

B. Label-flipping-attacks detection method performance

In this subsection, the author presents the performance of her proposed algorithm for detecting the attackers' models. The obtained results are estimated probabilities of detection and false alarm; however, in this case, these probabilities relate to the *detection of an attacker*, not the detection of occupied RBs. Hence, P_{d}^{AD} and $P_{\text{fa}}^{\text{AD}}$ now denote the probability of true positive detection of an attacker and the probability of false alarm (false positive) detection of an attacker, respectively. For every considered attack type (described in Section 5.3), and for every statistical test that the author has employed in her proposed detection algorithm (described in Section 5.4), the values of P_{d}^{AD} and $P_{\text{fa}}^{\text{AD}}$ have been estimated based on the number of true positive decisions and false positive decisions, respectively, over the total number of truly launched attacks.

Here, it is assumed that every time an attack is launched, the attackers modify an entirely correct global model, or in other words, they can receive a correct global model created by FL (based on genuine users' models). This is to eliminate the impact of an already corrupted model on the examined system performance, i.e., it is tested how easy or difficult it can be to detect attackers' models, assuming that the other local models have been generated without any history of previous attacks. As previously mentioned, the attacks are assumed to be launched after the first few FL iterations, as seen in Figure 5.8 and 5.10. After these first few iterations, the FL global model reaches a stable state with target high P_{d}^{SS} and low P_{fa}^{SS} values.

Attacks aimed at the false increase in RB occupancy. Similarly, as in section 5.5.2, the author will first focus on occupancy-increasing attacks. As shown in Figures 5.8 and 5.9, the random 75%-occupancy attack has the most significant negative impact on the SS performance in terms of increasing $P_{\text{fa}}^{\text{SS}}$. Results of the proposed algorithm of detection of attackers launching this type of attack are presented in Figure 5.12. The Kolmogorov-Smirnov test algorithm has obtained the best detection results. The probability of true-positive detection of malicious users P_{d}^{AD} is relatively high, and $P_{\text{fa}}^{\text{AD}}$ equals 0% for all SNR values. The algorithm that uses Fisher's test achieves the worst detection performance (P_{d}^{AD} is lower than 60%, and $P_{\text{fa}}^{\text{AD}}$ is between 10% and 20% for every SNR).

The P_{d}^{AD} and $P_{\text{fa}}^{\text{AD}}$ non-monotonicity for all tests in case of this aggressive random attack (aiming at 75% of false increase of RB occupancy) result from the following phenomenon. In low-SNR regions, high-power noise may contribute to the increase of the attackers' detection probability and the increase in the rate of false alarms. In medium-SNR regions, the effect is less visible; the noise can compensate for the random attack, making it hard to distinguish the attack and noise effect and detect the attacks. In high-SNR regions, the noise does not play much of a role, and mainly, the effect of the aggressiveness of an attack can be observed.

The attack that is the same aggressive and has the second most negative impact on the FL-based SS performance is the encapsulation (2, 2, 2, 2) attack. Estimated P_{d}^{AD} and $P_{\text{fa}}^{\text{AD}}$ of this type of attack are presented in Figure 5.13. Here, the Kolmogorov-Smirnov-test-based detection method also has the best performance. Application of the Tukey test results in slightly lower P_{d}^{AD} and almost the same

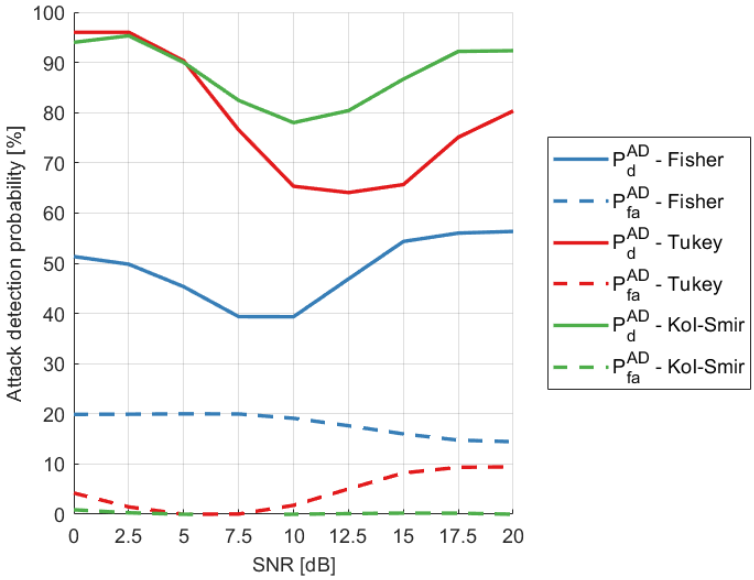


FIGURE 5.12: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Random attacks aimed at the false increase in RB occupancy to 75%.

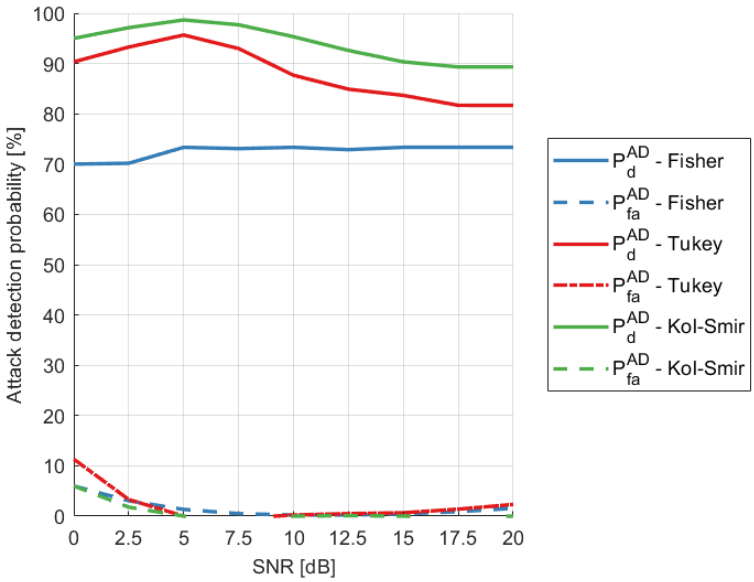


FIGURE 5.13: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Encapsulation (2,2,2,2) attacks aimed at the false increase in RB occupancy.

$P_{\mathrm{fa}}^{\mathrm{AD}}$. Similarly to the case of a random 70%-occupation attack, the Fisher test performs worse than the other two. The following conclusions can be drawn by comparing the results of the encapsulation (2, 2, 2 2) attack and random 70% attack. The random attack aiming at a very high false increase of the RB occupancy (up to 75%) decreases the FL-based SS performance and makes it hard to detect the attackers. On the other hand, the encapsulation (2, 2, 2, 2) attack, which also falsely increases the RB occupancy to 75%, is not as destructive for SS and is also easier to detect. The Kolmogorov-Smirnov-test-based algorithm detected attacking UEs with over 90% accuracy while maintaining $P_{\mathrm{fa}} \approx 0\%$ for almost all SNRs. Again, slightly better performance in terms of $P_{\mathrm{d}}^{\mathrm{AD}}$ and worse performance in terms of $P_{\mathrm{fa}}^{\mathrm{AD}}$ is visible in the low-SNR region.

The proposed algorithm is now examined in the presence of attacks that intro-

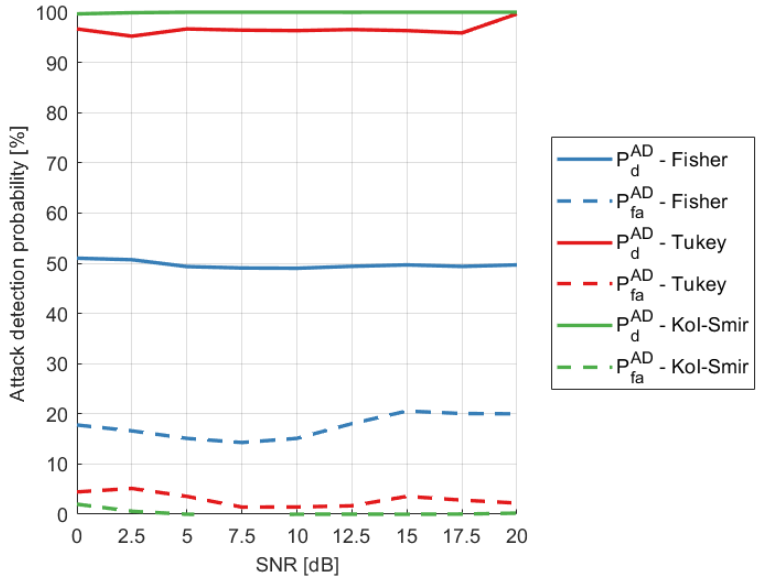


FIGURE 5.14: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Random attacks aimed at the false increase in RB occupancy to 55%.

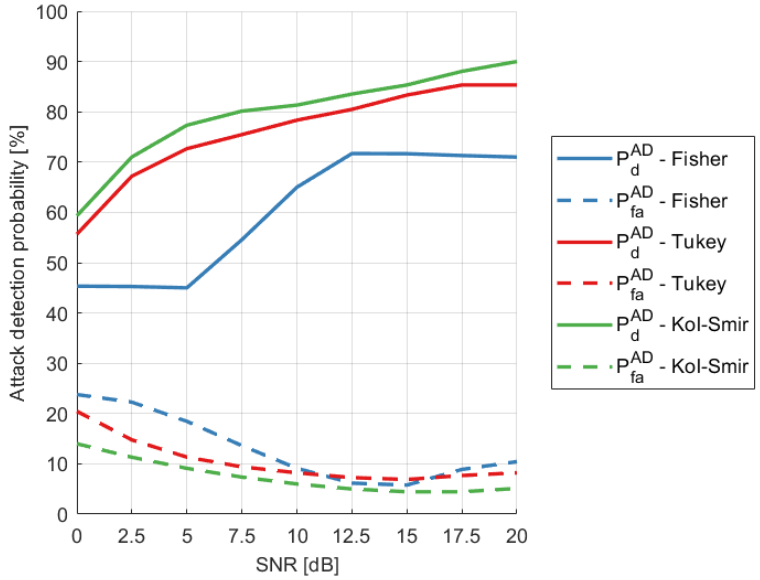


FIGURE 5.15: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Encapsulation (1,1,1,1) attacks aimed at the false increase in RB occupancy.

duce fewer fake “occupied" labels to the training data. As presented in Figure 5.14, the random 55%-occupation attack is detected with $P_{\mathrm{d}} = 100\%$ for all SNR values by Kolmogorov-Smirnov-based algorithm, while $P_{\mathrm{fa}} = 0\%$ for this applied test. The encapsulation (1,1,1,1) attack, which is more destructive for the FL-based SS performance, is less likely to be detected, as shown in Figure 5.15. There, Kolmogorov-Smirnov-based algorithm achieves $P_{\mathrm{d}} = 90\%$ and $P_{\mathrm{fa}} = 5\%$ for SNR $= 20$ dB.

The conclusions of the tests of the proposed attackers-detection method performance in the presence of false RBs occupancy-increasing attacks are as follows. The attacks that, for a given number of introduced fake labels, cause more damage to the FL-based SS algorithm, are more challenging to detect. Random attacks are generally easier to detect for a low number of introduced fake labels because the random changes in labels are harder to map in CNN weights. The encapsulation

attacks with the same false RBs occupancy percentage increase are better camouflaged in the data labels. On the other hand, for a high percentage of false increase in RBs occupancy, the random attack has a more damaging effect on FL-based SS and is harder to detect than the coordinated attack of the same occupancy level.

The Kolmogorov-Smirnov test applied to the attackers' detection algorithm performs the best out of the three examined statistical tests, although, for a high percentage of falsely increased RBs occupancy, $P_{\mathrm{d}}^{\mathrm{AD}}$ is not high for the examined SNRs. This is because the negative effect made on the models outweighs the impact of a channel, even for high SNRs. (Note that 10 out of 25 UEs are malicious, and the attacks launched are highly damaging, i.e., they can reverse the decision on the genuine UEs versus the attackers' clusters.) This problem does not occur in the case of less aggressive attacks.

Attacks aimed at the false decrease in RBs occupancy. In this section, the occupancy-decreasing attacks are examined. The results obtained for the random 16%-occupation and the encapsulation (-1, -1, 0, 0) attack (resulting in the same decrease of RB occupancy percentage) are presented in Figure 5.16 and Figure 5.17 respectively. The encapsulation (-1, -1, 0, 0) attack was the most successful one (had the most negative impact on FL-based SS performance) out of the considered decreasing-occupancy attacks. Similarly, as in the case of the increasing-occupancy attacks, the Kolmogorov-Smirnov-based attack detection method has the best performance in terms of high $P_{\mathrm{d}}^{\mathrm{AD}}$ and low $P_{\mathrm{fa}}^{\mathrm{AD}}$, while the Fisher-based algorithm has the worst. By comparing this set of results, it can be observed that a random 16% attack is easier to detect. Analogous results can be observed in Figures 5.18 and 5.19 where the attackers' detection results are presented for the random 22%-occupation and encapsulation (-1, 0, 0, 0) attack, respectively. These kinds of attacks are harder to detect than the ones resulting in 16% occupation of RBs. The encapsulation (-1, 0, 0, 0) attack is slightly more easily detectable than the random 22%-occupancy attack, which acts like random noise in the truly occupied RBs.

Note that the attacks that aim at decreasing the RBs' occupancy to 16% have similar effects on the performance of our attackers' detection method, as the occupancy-increasing attacks that aim at increasing the RBs occupancy level to 55%. In both cases, encapsulation attacks are harder to detect than random attacks aiming at the exact relative RBs' occupancy change. The decreasing-occupancy attacks that reduce RBs' occupancy to 22% are even harder to detect. However, they are not so harmful to the FL-based SS performance (in terms of $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$).

C. Defence mechanism impact on the SS performance.

In this subsection, the author verifies the impact of the proposed label-flipping attack protection method on FL-based SS performance. The experiments were performed for all considered SNR values for the last iteration of FL (as in the experiments presented in the former subsections). The estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ for all attack types and variants of the attackers detection method are presented below.

Protection against attacks aimed at the false increase in RBs occupancy. Figs. 5.20a, 5.20b, and 5.20c present SS results after the proposed protection method against model poisoning has been applied, employing the Kolmogorov-Smirnov,

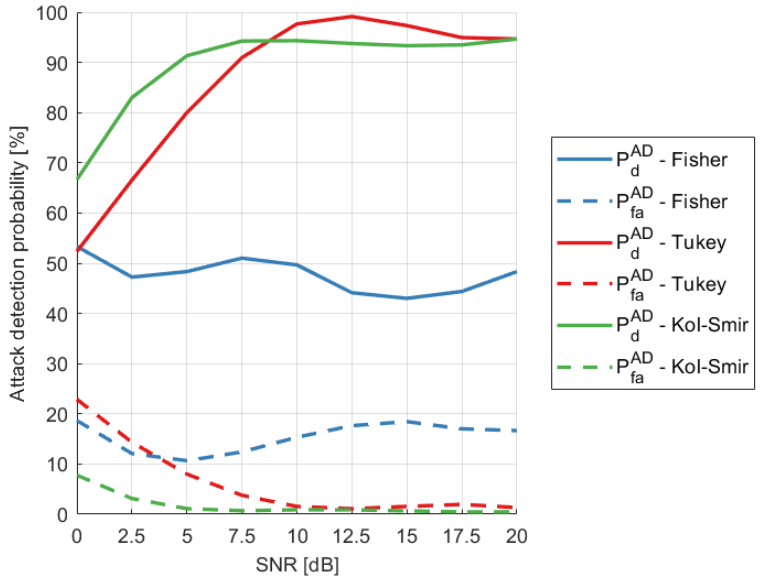


FIGURE 5.16: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Random attacks aimed at the false decrease in RBs occupancy to 16%.

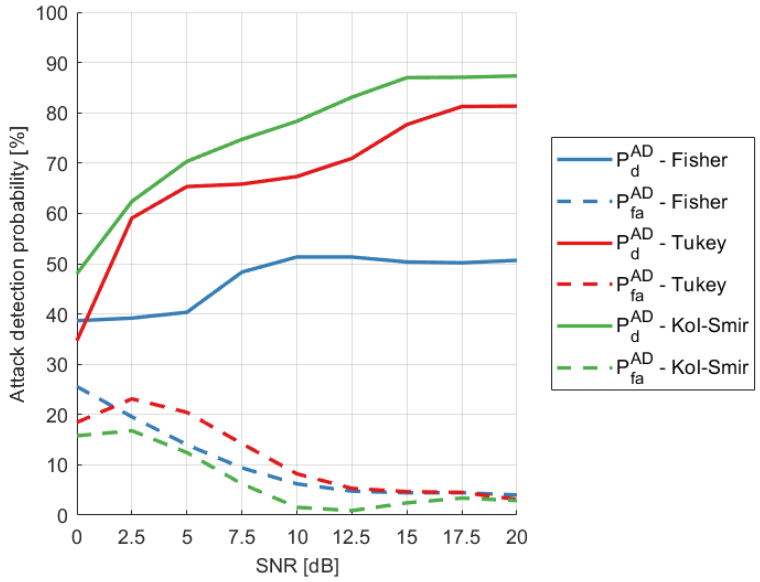


FIGURE 5.17: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Encapsulation (-1,-1,0,0) attacks aimed at the false decrease in RBs' occupancy.

Tukey, and Fisher test respectively. There, attacks aimed at a false increase in RBs occupancy (increase of $P_{\mathrm{fa}}^{\mathrm{SS}}$) have been simulated.

By comparing Fig. 5.20a and Fig. 5.9, one can observe that the author's method using the Kolmogorov-Smirnov test results in the decrease of $P_{\mathrm{fa}}^{\mathrm{SS}}$ (and as a side-effect also in a slight reduction of $P_{\mathrm{d}}^{\mathrm{SS}}$). Moreover, the detection of the most aggressive attacks results in the highest decrease of $P_{\mathrm{fa}}^{\mathrm{SS}}$ relative to the original (not protected) system under attack.

For example, $P_{\mathrm{fa}}^{\mathrm{SS}}$ in case of random 75% -occupation attack and SNR=0 dB, 5 dB, 10 dB, and 20 dB decreases from 48.70%, 78.06%, 65.30%, and 40.09% to 8.75%, 14,31%, 22.36% and 10,80% respectively (which constitutes the relative reduction by 82.03%, 81.67%, 65.76%, and 73.06% respectively) when the author's defense method is applied. The relative reduction of $P_{\mathrm{fa}}^{\mathrm{SS}}$ in case of the encapsu-

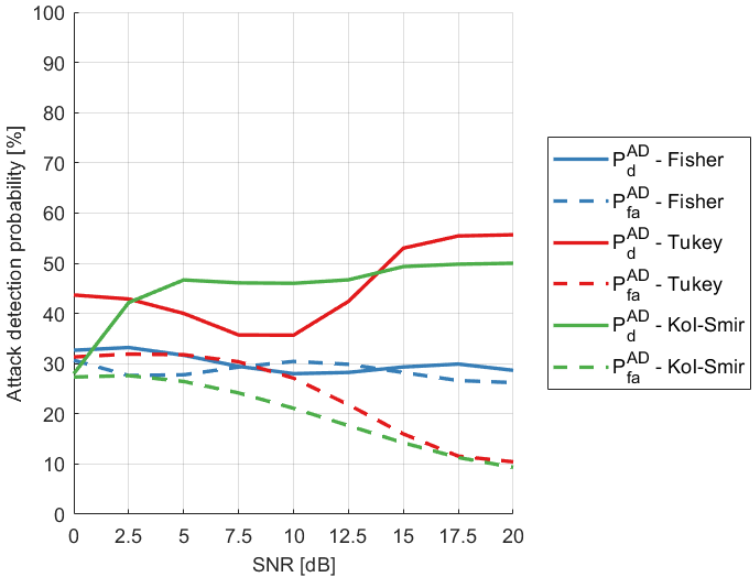


FIGURE 5.18: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Random attacks aimed at the false decrease in RBs occupancy to 22%.

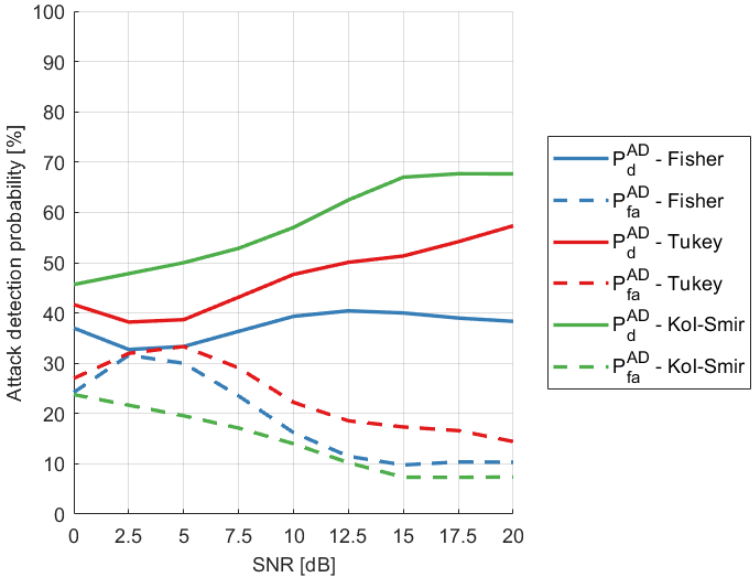
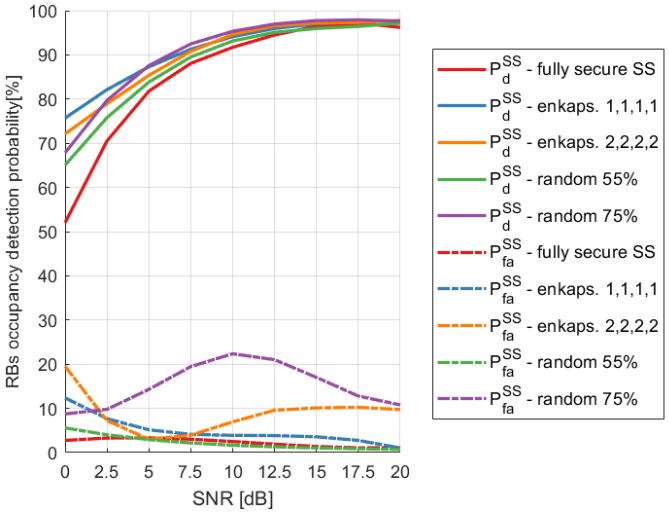


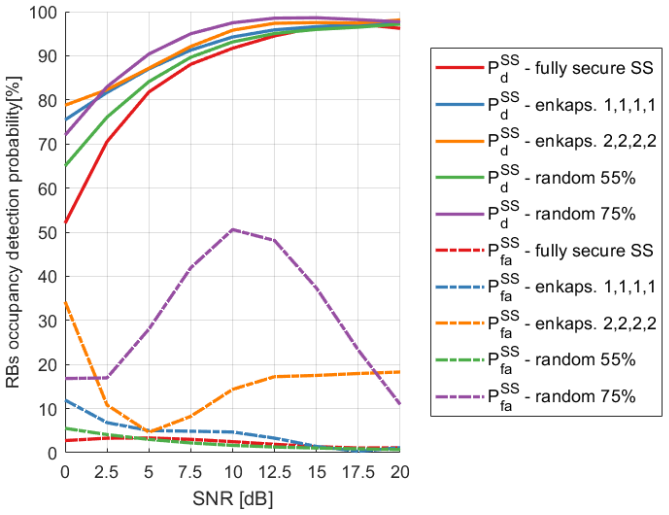
FIGURE 5.19: Estimated $P_{\mathrm{d}}^{\mathrm{AD}}$ and $P_{\mathrm{fa}}^{\mathrm{AD}}$ vs. SNR; Encapsulation (-1,0,0,0) attacks aimed at the false decrease in RBs occupancy.

lation (2, 2, 2, 2) attack is achieved up to 69.66%, 88.70%, 73.54%, and 57.12% for the same respective SNRs. In case of the other considered less aggressive attacks aiming at the false increase of the RBs occupancy, the author's method using the Kolmogorov-Smirnov test reduces $P_{\mathrm{fa}}^{\mathrm{SS}}$ to values close to that of the full-secure system (particularly for SNR$\geq$ 5 dB).

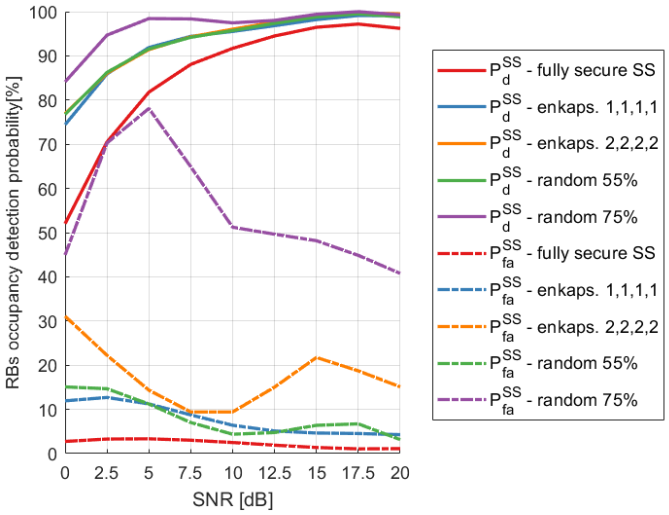
Figs. 5.20b and 5.20c also show a decrease of $P_{\mathrm{fa}}^{\mathrm{SS}}$ when our protection method is applied using the Tukey and Fisher test, respectively, but this decrease relative to an unprotected system is not as significant as for the Kolmogorov-Smirnov test (in Fig. 5.20a). The reason is the fact that the Kolmogorov-Smirnov test yields the best results in attacked models detection (as shown in Figs. 5.12–5.15 by green solid line). Comparing Figs. 5.20b and 5.20c, it is also visible that the application of the Tukey test results in a higher decrease of $P_{\mathrm{fa}}^{\mathrm{SS}}$ than in the case of employing



(A) Kolmogorov-Smirnov test applied



(B) Tukey test applied



(C) Fisher test applied

FIGURE 5.20: Estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ vs. SNR for FL-based SS (after the last FL iteration) when applying the attack protection method; Attacks aimed at the false increase in RBs occupancy.

the Fisher test. Again, this effect is a projection of the higher accuracy in attacker detection (compare red and blue solid lines in Figs. 5.12–5.15).

Note that none of the similarity tests applied in the author's attack protection method results in $P_{\mathrm{fa}}^{\mathrm{SS}}$ reaching the exact values of a fully secure FL-based SS. This is because none of these methods reaches 100% accuracy in attacker detection. The results closest to fully secure FL-based SS are obtained by employing the Kolmogorov-Smirnov test in case of the random 55%-occupancy attack (green dashed line in Fig. 5.20a), which is due to the near-perfect attack detection performance (green solid line in Fig. 5.14).

Protection against attacks aimed at the false decrease in RBs occupancy. As mentioned, attacks aiming at a false decrease in RBs occupancy cause a reduction of measured $P_{\mathrm{d}}^{\mathrm{SS}}$ (and in $P_{\mathrm{fa}}^{\mathrm{SS}}$, as a side-effect). Therefore, if successful, the author's method should increase $P_{\mathrm{d}}^{\mathrm{SS}}$. Such an effect can be observed when comparing Figs. 5.21a, 5.21b, and 5.21c (showing the results of our method employing the Kolmogorov-Smirnov, Tukey and Fisher test respectively) with Fig. 5.11 (presenting unprotected FL-based SS results). The highest increase of $P_{\mathrm{d}}^{\mathrm{SS}}$ was obtained for the most damaging attacks aiming at a false decrease in RBs' occupancy. For example, in the case of the encapsulation (-1, -1, 0, 0) attack, an increase of $P_{\mathrm{d}}^{\mathrm{SS}}$ is observed for SNRs equal to 5 dB, 10 dB, and 20 dB from 69.65%, 81.99%, and 87.26% to 80.73%, 91.10%, and 96.11% respectively (which constitutes an increase by 15.91%, 11.11%, and 10.18% respectively, relative to the original values for the unprotected system) when the Kolmogorov-Smirnov test was applied in our attack detection algorithm (compare yellow solid lines in Figs. 5.11 and 5.21a). As shown in Fig. 5.17, the application of this test results in the best attacker detection performance.

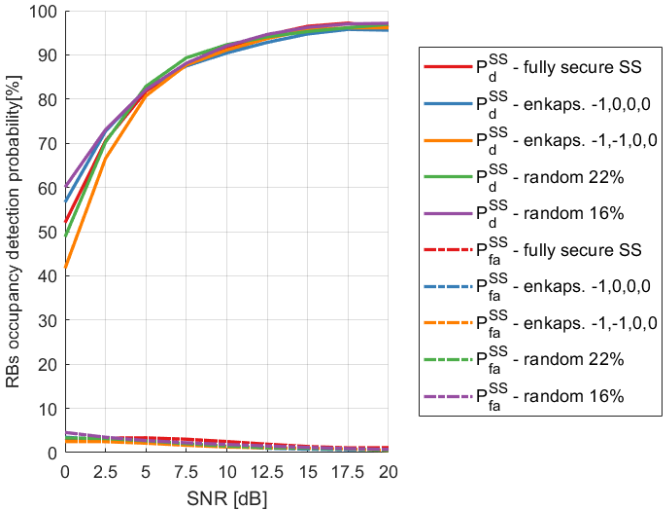
Finally, note that all statistical tests employed for attackers' detection and elimination in FL-based SS yield similar results in terms of SS performance for SNR$\geq$10 dB.

5.6 Chapter summary

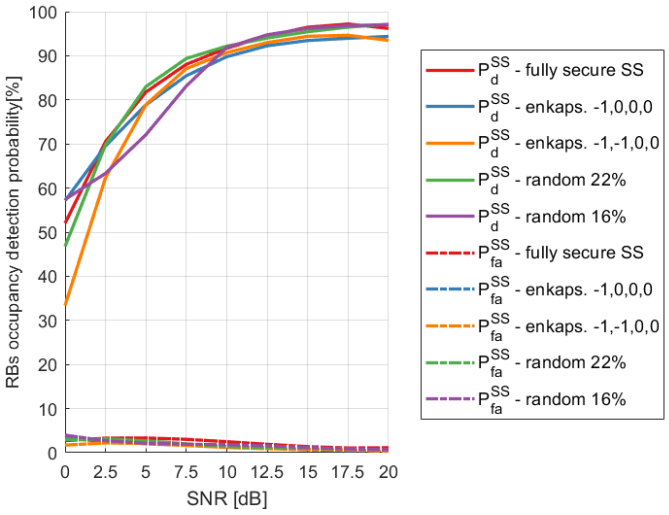
FL-based SS is characterized by higher reliability than autonomous sensing. It allows for spectrum prediction as opposed to schemes not incorporating ML techniques. It also ensures the privacy of local data since only the transmission of local model parameters is required. Finally, it allows for building a corporate spectrum occupation model ready to be used by new incoming users. However, FL-based SS can be a target of cyberattacks. The security threats originate from vulnerabilities of the applied ML and FL algorithms and the ubiquitous nature of the radio communication medium. This chapter summarized potential attacks on FL-based SS and indicated methods to detect, analyze, and defend against them. A taxonomy of attacks and defense methods has also been provided.

Despite the capabilities of the defense methods against the attacks on FL-based SS discussed in this chapter, each has its limitations, and none can be a one-stop-shopping solution to combat all threats. Thus, given FL's potential for SS in cognitive radio, robust security mechanisms are of considerable interest for future CR systems.

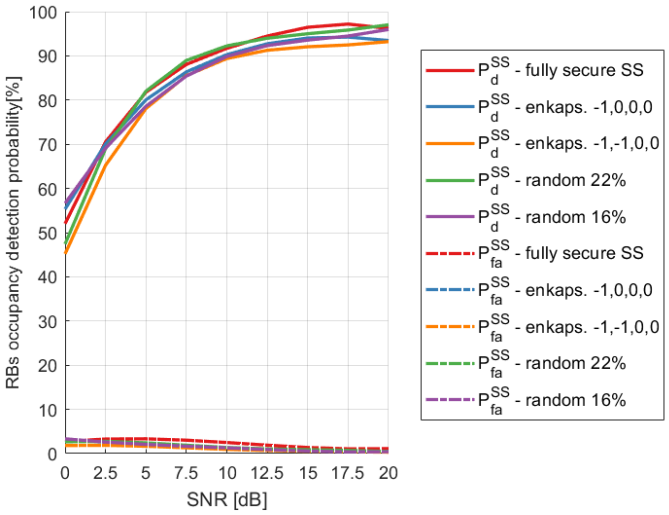
This chapter has also examined the impact of the label-flipping random and coordinated attacks on FL-based SS in the FL system. It has been concluded that



(A) Kolmogorov-Smirnov test applied



(B) Tukey test applied



(C) Fisher test applied

FIGURE 5.21: Estimated $P_{\mathrm{d}}^{\mathrm{SS}}$ and $P_{\mathrm{fa}}^{\mathrm{SS}}$ vs. SNR for FL-based SS (after the last FL iteration) when applying the attack protection method; Attacks aimed at the false decrease in RBs occupancy.

highly aggressive random attacks (with a high percentage of changed labels) aimed at increasing the false alarm (false positive decisions) rate impact the FL-based SS performance more negatively than similarly aggressive coordinated attacks. Conversely, coordinated attacks with the same aim introduce more damage to the FL-based SS performance than random attacks if they are less aggressive. Moreover, coordinated attacks aiming at decreasing the true-positive decision rate are more (negatively) effective than random ones.

The author has also proposed a new method for continuous (*zero-trust*) monitoring, detection of the attacked models, and their elimination in the FL-based SS algorithm. The proposed semi-blind method does not require an apriori knowledge of who are the genuine actors participating in FL. The proposed algorithm applies statistical tests and clustering to determine the genuine and attacked models. The author tested the method in three different variants, each using a different statistical test: Fisher, Tukey, and Kolmogorov-Smirnov.

It has been concluded that, in principle, highly aggressive attacks that aim to increase the FL-based SS false alarm rate are hard to detect. Interestingly, the same issue is observed for very mild (extremely non-aggressive) attacks that aim to decrease the (true-positive) detection rate. However, in this case, the attackers' misdetection is not so harmful to the FL-based SS performance.

Except for these kinds of attacks, the general observation is that coordinated attacks are harder to detect than random ones. This is because a coordinated attack mimics the true data labels better than a random attack and can better fool the FL-based SS algorithm. The Smirnov-Kolmogorov test performs best out of all considered similarity tests for model clustering, resulting in the attackers' model classification (and attack detection). In case of moderately aggressive attacks, the resulting probability of attackers' detection $P_{\mathrm{d}}^{\mathrm{AD}}$ is usually between 90% and 100% for all considered SNR values, while $P_{\mathrm{fa}}^{\mathrm{AD}}$ is close to 0%. Thus, the proposed method proved efficient in these considered attack scenarios. Moreover, it can decrease the SS probability of false alarms $P_{\mathrm{fa}}^{\mathrm{SS}}$ by as much as 89 % and increase the SS probability of detection $P_{\mathrm{d}}^{\mathrm{SS}}$ by 16 %in case of the most severe targeted attacks in the most critical SNR ranges.

The complexity of the author's proposed attack detection and mitigation method is not particularly high (when compared to other ML-based anomaly detection methods), as it involves relatively low-complex statistical test calculation and k-means classification (using Lloyd's algorithm), both of which have $\mathcal{O}(\mathcal{C}_2^N)$ complexity, where $\mathcal{C}_2^N$ is the N-choose-2 combination and N is the number of sensors. (Lloyd's algorithm complexity is also a linear function of the number of iterations). Regarding the implementation challenges, the scalability of the proposed method in the case of diverse and dynamically changing numbers of users and attackers should be further examined. The impact of continuous attacks, lasting for many iterations, genuine users' and attackers' heterogeneity (e.g., regarding their computational capabilities) on the method efficacy should also be considered for future research.

Chapter 6

Conclusions

In this dissertation, the problem of the application of Machine Learning (ML) methods to Spectrum Sensing (SS) and Spectrum Prediction (SP) in cognitive radio networks has been studied. First, in Chapter 1, the author of this thesis reviewed existing AI/ML techniques for enhancing context awareness in radio communication networks. She analyzed ML algorithms and frameworks for autonomous and cooperative SS, considering pattern recognition techniques in time, frequency, and spatial dimensions. Particular consideration has been given to the design recommendations for intelligent context-aware radio communication.

Next, in Chapter 2, the author proposed new ML algorithms to improve energy-measurement-based SS by learning signal features and traffic intensity correlation in time and frequency. The first stage of the proposed algorithms uses Energy Value (EV) and Energy Detection (ED) hard decisions, while the second phase uses classifiers like Gaussian Naive Bayes (NB), k-Nearest Neighbors (kNN), Random Forest (RF), and Support Vector Machine (SVM). The author demonstrated that kNN and RF, based on EVs input dataset, outperform ED hard decisions and increase the probability of spectrum detection (Resource Blocks (RBs) occupancy in 5G networks). Moreover, the proposed ML algorithms were implemented in various locations to reduce the need for multiple SNR estimations and improve performance by increasing detection probability and eliminating continued noise power estimation.

Then, in Chapter 3, the author explored the use of Deep Learning (DL) algorithms to improve autonomous SS performance, predict future spectrum occupancy, and estimate fading levels. Three DL algorithms, namely Neural Network (NN), Recurrent Neural Network (RNN), and Convolutional Neural Network (CNN), were designed and tested on two data sets representing different communication systems. The evaluation results show that CNN-based SS and SP is the best fitting method, resulting in the highest detection probability and the lowest false alarm probability in the considered 5G transmission scenarios. The author also presented a new CNN-based algorithm for improving spectrum sensing and spectrum occupation prediction. The algorithm combines CNN-based SS and SP with CNN-based fading level estimation, and optimization of the related decision threshold in order to reject the decisions on unoccupied RBs in the case when these RBs experience deep fading. This is to better protect the PU transmissions. The proposed algorithm also improves the spectrum occupancy prediction performance, facilitating SU preparation and enhancing secondary spectrum reuse and spectral efficiency.

The next contribution of the author of this thesis, described in Chapter 4, is

the design and application of Federated Learning (FL) to SS. The FL algorithm is based on locally trained CNN models, and requires few iterations to achieve very good performance in terms of high probability of RBs occupancy detection and low false alarm probability. Moreover, the proposed FL-based SS algorithm uses FL sensors to create multiple ML (CNN) models for diverse sensing conditions, enabling SS from the whole continuous range of SNR. This allows for offloading SUs from ML training responsibility. The author's proposed FL-based SS offers advantages over alternative schemes, including higher decision quality, better spectrum prediction, data privacy, and the ability to build a universal model for all SUs, making it suitable for new users.

The next major contribution by the author of this dissertation was discussed in Chapter 5. There, she explored the effects of label-flipping random and coordinated attacks on FL-based SS. She found that aggressive random attacks negatively impacted the SS performance, while less aggressive coordinated attacks, on the contrary, caused more damage. Moreover, coordinated attacks aiming at a decrease in the true positive decision rate were more effective (negatively) than random ones. A new method for continuous monitoring, detection, and elimination of attacked models in the FL-based SS algorithm was also proposed, using statistical tests, namely Fisher, Tukey, and Kolmogorov-Smirnov tests, for the SUs' models similarity, as well as clustering of these tests results. The author concluded that extremely aggressive attacks increasing the FL-based SS false alarm rate and very mild attacks decreasing the (true-positive) detection rate were hard to detect; however, the latter ones were not so harmful to the FL-based SS performance. Except for these kinds of attack, generally, coordinated attacks were harder to detect than random ones because they better mimic the true data labels. The Smirnov-Kolmogorov test performed best for genuine and attacked model clustering. The attack detection and mitigation method suggested by the author resulted in a nearly 100% detection probability for moderately aggressive attacks, proving the proposed method to be efficient in the scenarios considered.

Based on the original research conducted by the author of this thesis, as well as reviewing other published works in the area of ML methods applied to SS, the following conclusions can be drawn.

ML can improve the performance of spectrum detection implemented in autonomous sensors. This is especially true for DL methods designed by the author for 5G communication systems based on CNNs. These new methods are also capable of RBs occupancy prediction, which allows to better protect the PU transmission or better utilize the resources by SUs. The new cooperative SS methods based on CNNs in each sensor and FL algorithm proposed by the author can perform even better than the autonomous ones. Furthermore, in such a scenario, new incoming SUs can take advantage of the global FL model without the need for extensive data collection and training of the local model. Although FL assures data privacy by design, it is still prone to poisoning attacks. The new proposed anomaly detection method based on the application of statistical tests for model similarity and clustering for genuine and attacked models is capable of detecting attackers and compensating for their harmful effects on FL-based SS.

All the solutions proposed by the author of this thesis have been verified in computer simulation experiments. The reliability and performance of the SS methods was evaluated in terms of the probability of spectrum detection and the probability of false alarm. Additionally, the probability of detection of attackers and false

alarm have been estimated in the cyberattack scenario. It should be stressed that the results achieved depend on the scenarios considered, which in most cases reflect the 5G radio communication system.

In conclusion, the author believes that the thesis of the dissertation has been proved, i.e., There exist new methods for spectrum sensing in wireless communication systems that are based on machine learning and that are more reliable than the existing ones. The author has proposed these methods and evaluated them extensively in this dissertation.

Appendix A

TABLE A.1: ML for single node SS improvement (selected papers)

Paper	Number of PUs and SUs	Signal type	Channel type	Feature set	Blind sensing	ML method	ML used for:	Gained context information
[165]	1 PU, 1 SU	OFDM (DVB-T2)	6-tap typical urban (TU6)	SNR samples	Yes	NB with class reduction	SNR are first mapped to classes, NB with class reduction is used for spectrum sensing	Knowledge of PU presence, SNR classes
[74]	1 PU, 1 SU	-	AWGN	Three statistical features based on signal samples	-	SVM	SS; when spectrum is classified as free; the received signal is classified into several subclasses	Knowledge of PU presence, signal classes
[162]	1 PU, 1 SU	AM	AWGN	Four features are detected: Energy, three features for cyclostationarity	Partially (ANN is adjusted to AM cyclistationarity features)	BPNN	SS	Knowledge of PU presence; knowledge on AM signal presence
[195]	Not applicable (measurements)	GSM1800 downlink, satellite signals (measured)	Real measurements	Signal's power spectral density samples	Yes	LSTM network	Spectrum state prediction, estimation channel quality	Channel occupancy prediction with dedicated metrics (root mean square and classification accuracy)
[123]	1 PU, 1 SU	OFDM (4×2.5 MHz channels	AWGN (not well specified)	IQ samples, spectrograms	Yes	Deep CNN	Detection of channel utilization pattern	Presence of PU, channel utilization pattern
[199]	1 PU, 1 SU	Gaussian	AWGN	Energy feature vectors	Partially	K-means; SVM	K-means used for discovery transmission patterns and PU statistics, SVM for PU status based on energy feature set	PU presence, PU statistics, PU transmission patterns

Continued on next page

Table A.1 – continued from previous page

Paper	Number of PUs and SUs	Signal type	Channel type	Feature set	Blind sensing	ML method	ML used for:	Gained context information
[21]	Multiple SUs, multiple PU channels sensed	not specified	Binary Symmetric Channel	-	-	DEC-POMDP	signal presence sensing, improvement of medium access scheme	PU's behaviour learnt in distributed way by each SU
[89]	Not applicable (measurements)	Radar signal - SPN-43 (measured)	Real measurements	spectrograms data	Yes	SVM, kNN, GMM, 7 different CNNs, LSTM RNN	SS	Knowledge of PU presence; knowledge of spectrum occupancy statistics and power distributions
[166]	N PUs, 1 SU	Not specified	Rayleigh fading channel, AWGN	Spectral values	Yes	TxMiner, Log-Rayleigh Mixture Model	multiple PUs detection	Knowledge of PU presence; modulation classification
[194]	1 PU, 1 SU	Frequency hopping	Real measurements	Windowed historical SS results	Yes	LSTM, back propagation network	Predicting feature spectrum state	Knowledge of PU presence
[4]	1 PU, 1 SU	-	-	Traffic patterns	Partially	Multilayer perceptron, RNN, LSVM, GSVM	Predicting feature spectrum state	Prediction of PU absence periods
[200]	1 PU, 1 SU	OFDM	AWGN	Features derived from cyclostationarity	Yes	Softmax regression	Improving cyclostationarity SS	Knowledge of PU presence
[42]	1 PU, 1 SU	BPSK	AWGN	Signal samples	Partially	SVM	Detection of PU presence	Knowledge of PU presence

Continued on next page

Table A.1 – continued from previous page

Paper	Number of PUs and SUs	Signal type	Channel type	Feature set	Blind sensing	ML method	ML used for:	Gained context information
[144]	1 PU, 1 SU	-	AWGN	differential entropy vector	Yes	SVM, kNN, Random Forest, logistic regression	Detection of PU presence	Knowledge of PU presence
[190]	1 PU, 1 SU	BPSK	AWGN	Signal samples	partially	SOM, genetic algorithms (GA), SVM	Detection of PU presence	Knowledge of PU presence
[16]	1 PU, 1/many SUs	BPSK	AWGN	Signal samples	Partially	SVM	Detection of PU presence	Knowledge of PU presence
[13]	Many PUs, many SUs	BPSK	AWGN	Signal samples	partially	SVM, multi-class SVM, beamforming-aided SVM	Presence of PU signal; prediction of PU status	Knowledge of the presence of multiple PUs; Prediction of PU behaviour
[189]	1 PU, 1 SU	Gaussian	AWGN and some fading	Signal samples	Yes	K-means, SVM	Detection of PU presence	Knowledge of PU presence; knowledge of PU traffic patterns
[154]	Not applicable (measurements)	GSM 850 band measurements	Real measurements	Spectral data	Yes	kNN, SVM, DT, LR	Detection of PU presence	Knowledge of PU presence
[14]	Many PUs, many SUs	BPSK	AWGN and Rayleigh fading	Signal samples	Yes	K-means supported by Kalman filter tracking	Detection of PU presence, channel estimation	Knowledge of PU presence; enhanced channel estimation

Continued on next page

Table A.1 – continued from previous page

Paper	Number of PUs and SUs	Signal type	Channel type	Feature set	Blind sensing	ML method	ML used for:	Gained context information
[172]	Not applicable (measurements)	FM band, EGSM band, DCS band, UHF TV band	Real measurements	Signal samples	Yes	BPNN	Detection of spectrum occupancy	Knowledge of PU presence
[60]	1 PU, many SUs	BPSK	AWGN	Signal samples; cyclostationarity features	Yes	CNN	Detection of PU presence	Knowledge of PU presence
[191]	1 PU, 1 SU	Artificially generated burstly transmission	Measured data	Signal samples	Yes	CNN, LSTM, fully connected NN (FCNN)	Detection of PU presence	Detailed knowledge of PU presence

TABLE A.2: ML for decision making in Fusion Centers in Cooperative Spectrum Sensing

Paper	**Number of PUs and SUs**	**Sensed signal type**	**Channel type**	**Blind sensing**	**ML input data**	**ML method**	**ML used for:**	**Gained context information**
[115]	1 PU, N SUs	PU's signal modeled as Gaussian process	AWGN	No	Energy values used (input features), energy detection decisions (labels during training)	kNN, SVM, NB, DT	Decision making in FC - PU presence detection	spectrum state
[161]	M PUs, N SUs	TV broadcasting	AWGN	Yes	Collected signal power values in different locations	kNN	Completing the missing spectrum points	Spatial power map, spectrum white space database
[147]	1 PU, N SUs	-	AWGN	Yes	Energy detection decisions	kNN	Local classification of sensing data into PU absent or PU present classes	Spectrum state
[103]	1 PU, N SUs	-	AWGN	No	Probability vector (probabilities of signal presence/ absence)	K-means, SVM	Decision making on spectrum occupancy in FC	Spectrum state
[71]	M PUs, N SUs	-	Path loss, AWGN	Yes	Energy detection test statistics	NP-FSVM (fuzzy SVM with non-parallel hyperplane), for comparison: SVM and K-means clustering	Alleviating noise uncertainty effect, decision making on spectrum occupancy in FC	Spectrum state
[29]	1 PU, N SUs	TV	AWGN, fading	No	Energy detection decisions collected from SUs	SVM (RBF-SVM, polynomial-SVM, linear-SVM)	Decision making in FC - PU presence detection	Spectrum state

Continued on next page

Table A.2 – continued from previous page

Paper	Number of PUs and SUs	Sensed signal type	Channel type	Blind sensing	ML input data	ML method	ML used for:	Gained context information
[88]	1 PU, N SUs	-	AWGN, path loss, fading, shadowing	Yes	Energy detection decisions collected from SUs	CNN	Decision making on spectrum occupancy in FC	Spectrum state
[101]	1 PU, N SUs	-	AWGN, fading, shadowing	Yes	Local sensing decisions collected from SUs	Proposed reinforcement learning algorithm (Q-learning, Sarsa, Action-Critic-based)	Decision making on spectrum occupancy in FC	Spectrum state
[188]	M PUs, N SUs	-	AWGN	Yes	Energy values vectors collected from SUs (energy values in different time moments)	Proposed beta-process sticky hidden Markov model (BP-SHMM)	Defining number and type of different spectrum states	Spectrum state and information on PUs' locations
[31]	1 PU, N SUs	-	Path loss, AWGN	Yes	Energy values measured by SUs	Hybrid SVM based AdaBoost and decision stumps based AdaBoost	Decision making on spectrum occupancy in FC	Spectrum state
[118]	1 PU, N SUs	PSK	AWGN	No	Energy detection decisions collected from SUs	ANN	Decision making on spectrum occupancy in FC and in cluster head	Spectrum state
[107]	M PUs, N SUs	-	AWGN, fading	Yes	Energy values collected from SUs	ELM	Decision making on spectrum occupancy in FC	Spectrum state

Continued on next page

Table A.2 – continued from previous page

Paper	Number of PUs and SUs	Sensed signal type	Channel type	Blind sensing	ML input data	ML method	ML used for:	Gained context information
[91]	1 PUs, N SUs	-	AWGN	Yes	Energy values collected from SUs	SVM	Grouping SUs in order to make cooperation more efficient, decision making on spectrum occupancy	Spectrum state, the quality of a given SU's data
[163]	1 PU, N SUs	-	AWGN, path loss	No	Normalized energy values collected from SUs	NB	Estimating channel occupancy probability	Spectrum state
[52]	M PUs, N SUs	-	AWGN, path loss	No	Normalized energy values collected from SUs	SVM, kNN, NB	Decision making on spectrum occupancy in FC	Spectrum state
[32]	1 PU, N SUs	-	AWGN	Yes	Data extracted by applying PCA	K-means	Decision making on spectrum occupancy in FC	Spectrum state
[164]	M PUs, N SUs	-	AWGN, fading, shadowing	No	Normalized energy values collected from SUs	kNN, SVM, K-means, GMM	Decision making on spectrum occupancy in FC	Spectrum state
[113]	1 PU, N SUs	-	AWGN	No	Energy detection decisions collected from SUs	kNN, SVM, NB, DT	Decision making on spectrum occupancy in FC	Spectrum state
[56]	1 PU, N SUs	-	AWGN, fading	Yes	Power Spectral Density	two SVM-based algorithms	Decision making on spectrum occupancy in FC	Spectrum state
[208]	M PUs, N SUs	-	-	-	Sensing decisions collected form SUs	No-regret learning	detecting malicious SUs	Knowledge of the presence and number of malicious users

Continued on next page

Table A.2 – continued from previous page

Paper	**Number of PUs and SUs**	**Sensed signal type**	**Channel type**	**Blind sensing**	**ML input data**	**ML method**	**ML used for:**	**Gained context information**
[38]	M PUs, N SUs	-	AWGN, fading, shadowing	No	Energy detection decisions collected from SUs	Fisher linear discriminant analysis	Determining linear coefficients for all SUs sensing decisions in cooperative sensing	Spectrum state and PUs' locations
[104]	M PUs, N SUs	-	AWGN	No	Energy detection with soft combining decisions collected from SUs	Proposed multi-agent reinforcement learning	Determining the free frequency bands	Spectrum state

TABLE A.3: ML for traffic pattern recognition

paper	pattern type [T - time, F - frequency, S - spatial]	Sensed signal/ system type	Data measured/simulated	ML algorithm	ML used for	ML input data
[177]	T + F (separately for different locations in space)	LTE	simulated	kNN, Random Forest	spectrum sensing	Energy values/energy detection decision per a given resource block, information on energies of adjacent resource blocks, time and frequency information
[26]	T + F	IEEE 802.11	simulated	CNN	Transport protocol detection, traffic pattern, transmission rate	2D images - IQ samples in time, Short-Time Fourier Transformation of IQ samples in frequency
[195]	T (for different frequencies)	GSM1800 downlink, satellite signal	Measured	RNN	Prediction of PU's next spectrum state	Power spectral density values
[197]	T + F	3 MHz - 5.4 MHz band	measured	LSTM-based architecture	Prediction of PU's next spectrum state	Availability status of current and previous spectrum states
[138]	T	450 MHz-800 MHz band of land mobile radio	Measured	RNN, proposed ConvLSTM algorithm	Prediction of PU's next spectrum state	-
[137]	T	Generated random QPSK signal	Measured	RNN	Prediction of PU's next spectrum state	IQ samples
[203]	T	Signal generated according to Poisson distribution	Simulated	SVR-based online learning	Prediction of PU's next spectrum state	Received signal power values
[98]	T + F + S	big data in 5G	- (theoretical paper)	K-means clustering for feature extraction, other ML algorithms for spectrum prediction	Spectrum data feature extraction, spectrum prediction	(not specified)

Continued on next page

Table A.3 – continued from previous page

paper	pattern type [T - time, F - frequency, S - spatial]	Sensed signal/ system type	Data measured/simulated	ML algorithm	ML used for	ML input data
[178]	T + F + S	LTE	Simulated	kNN, Random Forest	Spectrum sensing	Energy values/energy detection decision per a given resource blocks, information on energies of adjacent resource blocks, time and frequency information
[179]	T + F	LTE/5G, sensor network signals	Simulated	NN, RNN, CNN	Next time slot occupancy prediction	For CNN: 2D images - energy values per resource block, time and frequency information
[67]	T + F	IoT signals	Measured	DBSCAN	Spectrum sensing	Energy detection decisions
[59]	T + S	MIMO-OFDM signals	Simulated	Delayed feedback reservoir (RNN)	Spectrum sensing	Signal samples

References

[1] The ericsson mobility report, q2 update 2024. Available online: `https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/june-2024`. (accessed on September-2024).

[2] Mw online dictionary. `https://www.merriam-webster.com/dictionary/context`. Accessed: 2020-07-29.

[3] F. Adly, S. Muhaidat, and P. D. Yoo. Cumulant-based automatic modulation classification over frequency-selective channels. In 2018 IEEE International Conference on Cyber, Physical and Social Computing (CPSCom 2018), Halifax, Canada, 7 2018.

[4] A. Agarwal, S. Dubey, M. A. Khan, R. Gangopadhyay, and S. Debnath. Learning based primary user activity prediction in cognitive radio networks for efficient dynamic spectrum access. In *2016 International Conference on Signal Processing and Communications (SPCOM)*, pages 1–5, June 2016.

[5] Anirudh Agarwal, Aditya S Sengar, and Ranjan Gangopadhyay. Spectrum occupancy prediction for realistic traffic scenarios: Time series versus learning-based models. *Journal of Communications and Information Networks*, 3(2):35–42, 2018.

[6] S. Ahn and D. Kim. Proactive context-aware sensor networks. In K. Römer, H. Karl, and F. Mattern, editors, *Wireless Sensor Networks*, pages 38–53, Berlin, Heidelberg, 2006. Springer.

[7] M. I. AlHajri, N. T. Ali, and R. M. Shubair. Indoor localization for IoT using adaptive feature selection: A cascaded machine learning approach. *IEEE Antennas and Wireless Propagation Letters*, 18(11):2306–2310, 2019.

[8] A. Annamalai, O. Olabiyi, S. Alam, O. Odejide, and D. Vaman. Unified analysis of energy detection of unknown signals over generalized fading channels. In *In Proceedings of the 2011 7th International Wireless Communications and Mobile Computing Conference, Istanbul, Turkey*, pages 636–641, 4-8 July 2011.

[9] Y. Aoki, T. Fujii, and T. Ide. Time series analysis of multiple primary user environment using HMM-based spectrum sensing. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–5, Aug 2018.

[10] T. Y. Arif, R. Munadi, and Fardian. Energy efficiency opportunity at same data rate and different MCS in IEEE 802.11n. In *2015 9th Asia Modelling Symposium (AMS)*, pages 142–147, 2015.

[11] Kshitiz Aryal, Maanak Gupta, and Mahmoud Abdelsalam. Analysis of label-flip poisoning attack on machine learning based malware detector. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 4236–4245. IEEE, 2022.

[12] A. Assra, J. Yang, and B. Champagne. An EM approach for cooperative spectrum sensing in multiantenna CR networks. *IEEE Transactions on Vehicular Technology*, 65(3):1229–1243, March 2016.

[13] O. P. Awe, A. Deligiannis, and S. Lambotharan. Spatio-temporal spectrum sensing in cognitive radio networks using beamformer-aided SVM algorithms. *IEEE Access*, 6:25377–25388, 2018.

[14] O. P. Awe, S. M. Naqvi, and S. Lambotharan. Kalman filter enhanced parametric classifiers for spectrum sensing under flat fading channels. In *Cognitive Radio Oriented Wireless Networks*, pages 235–247, Cham, 2015. Springer International Publishing.

[15] O. P. Awe, S. M. Naqvi, and S. Lambotharan. Variational bayesian learning technique for spectrum sensing in cognitive radio networks. In *In Proceedings of the 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Atlanta, GA, USA*, pages 1185–1189, 3-5 December 2014.

[16] O. P. Awe, Z. Zhu, and S. Lambotharan. Eigenvalue and support vector machine techniques for spectrum sensing in cognitive radio networks. In *Technologies and Applications of Artificial Intelligence (TAAI), 2013 Conference on*, pages 223–227, 2013.

[17] E. Bedeer, O. A. Dobre, M. H. Ahmed, and K. E. Baddour. Joint optimization of bit and power loading for multicarrier systems. *IEEE Wireless Communications Letters*, 2(4):447–450, 2013.

[18] Mourad Benmalek, Mohamed Ali Benrekia, and Yacine Challal. Security of federated learning: Attacks, defensive mechanisms, and challenges. *Revue des Sciences et Technologies de l'Information-Série RIA: Revue d'Intelligence Artificielle*, 36(1):49–59, 2022.

[19] Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pages 634–643. PMLR, 2019.

[20] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[21] M. Bkassiny, S. K. Jayaweera, and K. A. Avery. Distributed reinforcement learning based MAC protocols for autonomous cognitive secondary users. In *Proceedings of the 20th Annual Wireless and Optical Communications Conference, (WOCC '11)*, Taiwan, 2011.

[22] S. Boubiche, D. E. Boubiche, A. Bilami, and H. Toral-Cruz. Big data challenges and data aggregation strategies in wireless sensor networks. *IEEE Access*, 6:20558–20571, 2018.

[23] Pavlos S. Bouzinis, Panagiotis D. Diamantoulakis, and George K. Karagiannidis. Wireless federated learning (wfl) for 6g networks - part i: Research challenges and future trends. *IEEE Communications Letters*, pages 1–1, 2021.

[24] P. J. Brown, J. D. Bovey, and X. Chen. Context-aware applications: from the laboratory to the marketplace. *IEEE Personal Communications*, 4(5):58–64, 1997.

[25] Y. Cai, Z. Qin, F. Cui, G. Y. Li, and J. A. McCann. Modulation and multiple access for 5G networks. *IEEE Communications Surveys Tutorials*, 20(1):629–646, 2018.

[26] M. Camelo, T. D. Schepper, P. Soto, J. Marquez-Barja, J. Famaey, and S. Latré. Detection of traffic patterns in the radio spectrum for cognitive wireless network management. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020.

[27] Jin Cao, Maode Ma, Hui Li, Ruhui Ma, Yunqing Sun, Pu Yu, and Lihui Xiong. A survey on security aspects for 3gpp 5g networks. *IEEE communications surveys & tutorials*, 22(1):170–195, 2019.

[28] X. Cao, L. Liu, Y. Cheng, and X. Shen. Towards energy-efficient wireless networking in the big data era: A survey. *IEEE Communications Surveys Tutorials*, 20(1):303–332, 2018.

[29] C. Chembe, I. Ahmedy, R. M. Noor, D. Kunda, M. Oche, and A. B. Tambawal. Cooperative spectrum decision in cognitive vehicular network based on support vector machine. *Malaysian Journal of Computer Science*, 32(2):83–96, 2019.

[30] Mingzhe Chen, Deniz Gündüz, Kaibin Huang, Walid Saad, Mehdi Bennis, Aneta Vulgarakis Feljan, and H Vincent Poor. Distributed learning in wireless networks: Recent progress and future challenges. *IEEE Journal on Selected Areas in Communications*, 39(12):3579–3605, 2021.

[31] S. Chen, B. Shen, X. Wang, and H. Wu. SVM and decision stumps based hybrid AdaBoost classification algorithm for cognitive radios. In *2019 21st International Conference on Advanced Communication Technology (ICACT)*, pages 492–497, Feb 2019.

[32] X. Chen, F. Hou, H. Huang, and X. Jing. Principle component analysis based cooperative spectrum sensing in cognitive radio. In *2016 16th International Symposium on Communications and Information Technologies (ISCIT)*, pages 602–605, Sep. 2016.

[33] Zhibo Chen, Yi-Qun Xu, Hongbin Wang, and Daoxing Guo. Federated learning-based cooperative spectrum sensing in cognitive radio. *IEEE Communications Letters*, pages 1–1, 2021.

[34] P. Cheng, Y. Li, Z. Chen, and B. Vucetic. High-resolution wideband spectrum sensing based on sparse Bayesian learning. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–5, Oct 2017.

[35] X. Cheng, L. Fang, L. Yang, and S. Cui. Mobile big data: The fuel for data-driven wireless. *IEEE Internet of Things Journal*, 4(5):1489–1516, 2017.

[36] P. S. Cheong, M. Camelo, and S. Latré. Evaluating deep neural networks to classify modulated and coded radio signals. In I. Moerman, J. Marquez-Barja, A. Shahid, W. Liu, S. Giannoulis, and X. Jiao, editors, *Cognitive Radio Oriented Wireless Networks*, pages 177–188, Cham, 2019. Springer International Publishing.

[37] K. W. Choi and E. Hossain. Estimation of primary user parameters in cognitive radio systems via hidden Markov model. *IEEE Trans. Signal Process.*, 61:782–795, 2013.

[38] K. W. Choi, E. Hossain, and D. I. Kim. Cooperative spectrum sensing under a random geometric primary user network model. *IEEE Transactions on Wireless Communications*, 10(6):1932–1944, 2011.

[39] K. Cichoń, A. Kliks, and H. Bogucka. Energy-efficient cooperative spectrum sensing: A survey. *IEEE Communications Surveys Tutorials*, 18(3):1861–1886, thirdquarter 2016.

[40] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, January 1967.

[41] George Cybenko, Dianne P O'Leary, and Jorma Rissanen. *The mathematics of information coding, extraction and distribution*, volume 107. Springer Science & Business Media, 1998.

[42] Z. Dandan and Z. Xuping. SVM-based spectrum sensing in cognitive radio. In *Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on*, pages 1–4, 2011.

[43] Wim De Mulder, Steven Bethard, and Marie-Francine Moens. A survey on the application of recurrent neural networks to statistical language modeling. *Computer Speech & Language*, 30(1):61–98, 2015.

[44] A. K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Hum.-Comput. Interact.*, 16(2):97–166, December 2001.

[45] F. F. Digham, M. Alouini, and M. K. Simon. On the energy detection of unknown signals over fading channels. *IEEE Transactions on Communications*, 55(1):21–24, Jan 2007.

[46] S. Duan, K. Chen, X. Yu, and M. Qian. Automatic multicarrier waveform classification via PCA and convolutional neural networks. *IEEE Access*, 6:51365–51373, 2018.

[47] S. Enserink and D. Cochran. A cyclostationary feature detector. In *In Proceedings of the 1994 28th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA*, volume 2, pages 806–810, 31 October-2 November 1994.

[48] H. M. Farag and E. M. Mohamed. Improved cognitive radio energy detection algorithm based upon noise uncertainty estimation. In *In Proceedings of the 2014 31st National Radio Science Conference (NRSC), Cairo, Egypt*, pages 107–115, 28-30 April 2014.

[49] A. Galindo-Serrano and L. Giupponi. Distributed Q-learning for aggregated interference control in cognitive radio networks. *IEEE Transactions on Vehicular Technology*, 59(4):1823–1834, 2010.

[50] Alberto García-González, Antonio Huerta, Sergio Zlotnik, and Pedro Díez. A kernel principal component analysis (kpca) digest with a new backward mapping (pre-image reconstruction) strategy. *arXiv preprint arXiv:2001.01958*, 2020.

[51] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: continual prediction with LSTM. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, volume 2, pages 850–855, 1999.

[52] E. Ghazizadeh, B. Nikpour, D. A. Moghadam, and H. Nezamabadi-pour. A PSO-based weighting method to enhance machine learning techniques for cooperative spectrum sensing in CR networks. In *2016 1st Conference on Swarm Intelligence and Evolutionary Computation (CSIEC)*, pages 113–118, March 2016.

[53] Bimal Ghimire and Danda B Rawat. Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things. *IEEE Internet of Things Journal*, 9(11):8229–8249, 2022.

[54] Aritra Ghosh, Naresh Manwani, and P. S. Sastry. On the robustness of decision tree learning under label noise. In Jinho Kim, Kyuseok Shim, Longbing Cao, Jae-Gil Lee, Xuemin Lin, and Yang-Sae Moon, editors, *Advances in Knowledge Discovery and Data Mining*, pages 685–697, Cham, 2017. Springer International Publishing.

[55] A. J. Goldsmith and S.-G. Chua. Variable-rate variable-power MQAM for fading channels. *IEEE Transactions on Communications*, 45(10):1218–1230, 1997.

[56] Mehran Golvaei and Mohammad Fakharzadeh. A fast soft decision algorithm for cooperative spectrum sensing. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 68(1):241–245, 2021.

[57] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT Press, 2016.

[58] A. Habbal, S. I. Goudar, and S. Hassan. A context-aware radio access technology selection mechanism in 5G mobile network for smart city applications. *Journal of Network and Computer Applications*, 135:97–107, 2019.

[59] Kian Hamedani, Lingjia Liu, and Yang Yi. Energy efficient mimo-ofdm spectrum sensing using deep stacked spiking delayed feedback reservoir computing. *IEEE Transactions on Green Communications and Networking*, 5(1):484–496, 2021.

[60] D. Han, G. C. Sobabe, C. Zhang, X. Bai, Z. Wang, S. Liu, and B. Guo. Spectrum sensing for cognitive radio based on convolution neural network. In *Proc. of CISP-BMEI*, Shanghai, China, October 2017.

[61] E. Hayden. Data lifecycle management model show risks and integrated data flows. *Information Security Magazine*, 2008.

[62] S. Haykin, D. J. Thomson, and J. H. Reed. Spectrum sensing for cognitive radio. *Proceedings of the IEEE*, 97(5):849–877, 2009.

[63] A. He, K. K. Bae, and T. Newman. A survey of artificial intelligence for cognitive radios. *IEEE Trans. Veh. Technol.*, 59(4):1578–1592, May 2010.

[64] Poul E. Heegaard. Evolution of traffic patterns in telecommunication systems. In *2007 Second International Conference on Communications and Networking in China*, pages 28–32, 2007.

[65] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, Aug 1998.

[66] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[67] Bassel Al Homssi, Akram Al-Hourani, Zarko Krusevac, and Wayne S. T. Rowe. Machine learning framework for sensing and modeling interference in iot frequency bands. *IEEE Internet of Things Journal*, 8(6):4461–4471, 2021.

[68] Choong Seon Hong, Latif U. Khan, Mingzhe Chen, Dawei Chen, Walid Saad, and Zhu Han. *Federated Learning for Wireless Networks*. Springer, 2022.

[69] X. Hu, X. Z. Xie, T. Song, and W. Lei. An algorithm for energy detection based on noise variance estimation under noise uncertainty. In *In Proceedings of the 2012 IEEE 14th International Conference on Communication Technology, Chengdu, China*, pages 1345–1349, 9-11 November 2012.

[70] D.-T. Huang, S. Wu, and P. Wang. Cooperative spectrum sensing and locationing: A sparse Bayesian learning approach. In *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pages 1–5, 2010.

[71] Y. Huang, Y. Liang, and G. Yang. A fuzzy support vector machine algorithm for cooperative spectrum sensing with noise uncertainty. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec 2016.

[72] C.-L. Hwang and K. Yoon. *Multiple Attribute Decision Making, Methods and Applications A State-of-the-Art Survey.* Springer-Verlag Berlin Heidelberg, 1981.

[73] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE symposium on security and privacy (SP)*, pages 19–35. IEEE, 2018.

[74] S. U. Jan, V. H. Vu, and I. S. Koo. Performance analysis of support vector machine-based classifier for spectrum sensing in cognitive radio networks. In *In Proceedings of the 2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Zhengzhou, China*, pages 385–3854, 18-20 October 2018.

[75] H. Jo, C. Mun, J. Moon, and J. Yook. Interference mitigation using uplink power control for two-tier femtocell networks. *IEEE Transactions on Wireless Communications*, 8(10):4906–4910, 2009.

[76] A. K V, G. Naga Rama Mangaiah Naidu, K. K, K. D, and K. S. Spectrum sensing using sparse Bayesian learning. In *2019 International Conference on Communication and Signal Processing (ICCSP)*, pages 0582–0586, April 2019.

[77] J.M. Kahn, R.H. Katz, and K.S.J. Pister. Emerging challenges: Mobile networking for "smart dust". *Journal of Communications and Networks*, 2(3):188–196, September 2000.

[78] S. Kapoor, S. Rao, and G. Singh. Opportunistic spectrum sensing by employing matched filter in cognitive radio network. In *In Proceedings of the 2011 International Conference on Communication Systems and Network Technologies, Katra, Jammu, India*, pages 580–583, 3-5 June 2011.

[79] B. Khalfi, A. Zaid, and B. Hamdaoui. When machine learning meets compressive sampling for wideband spectrum sensing. In *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1120–1125, June 2017.

[80] K. Kim, Y. Xin, and S. Rangarajan. Energy detection based spectrum sensing for cognitive radio: An experimental study. In *In Proceedings of the 2010 IEEE Global Telecommunications Conference GLOBECOM 2010, Miami, FL, USA*, pages 1–5, 6-10 December 2010.

[81] S. Kim, H. Kim, and D. Hong. Joint power allocation and MCS selection in downlink NOMA system. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–4, 2018.

[82] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[83] A. Kliks, D. Triantafyllopoulou, L. De Nardis, O. Holland, L. Gavrilovska, and A. Bantouna. Cross-layer analysis in cognitive radio—context identification and decision making aspects. *IEEE Transactions on Cognitive Communications and Networking*, 1(4):450–463, 2015.

[84] A. Kosta, N. Pappas, and V. Angelakis. *Age of Information: A New Concept, Metric, and Tool*. now Publishers Inc, 2017.

[85] V. I. Kostylev. Energy detection of a signal with random amplitude. In *In Proceedings of the 2002 IEEE International Conference on Communications, New York, NY, USA*, volume 3, pages 1606–1610 vol.3, 28 April-2 May 2002.

[86] V. Kumar, H. Li, J. J. Park, and K. Bian. Crowd-sourced authentication for enforcement in dynamic spectrum sharing. *IEEE Transactions on Cognitive Communications and Networking*, 5(3):625–636, 2019.

[87] Y. LeCun, C. Cortes, and C. Burges. *Mnist handwritten digit database*. at&t labs, 2010.

[88] W. Lee, M. Kim, and D. Cho. Deep cooperative sensing: Cooperative spectrum sensing based on convolutional neural networks. *IEEE Transactions on Vehicular Technology*, 68(3):3005–3009, 2019.

[89] W. M. Lees, A. Wunderlich, P. J. Jeavons, P. D. Hale, and M. R. Souryal. Deep learning classification of 3.5-GHz band spectrograms with applications to spectrum sensing. *IEEE Transactions on Cognitive Communications and Networking*, 5(2):224–236, June 2019.

[90] Dongcheng Li, W Eric Wong, Wei Wang, Yao Yao, and Matthew Chau. Detection and mitigation of label-flipping attacks in federated learning systems with kpca and k-means. In *2021 8th International Conference on Dependable Systems and Their Applications (DSA)*, pages 551–559. IEEE, 2021.

[91] Z. Li, W. Wu, X. Liu, and P. Qi. Improved cooperative spectrum sensing model based on machine learning for cognitive radio networks. *IET Communications*, 12(19):2485–2492, 2018.

[92] F. Liu, Y. Zhou, and Y. Liu. A deep neural network method for automatic modulation recognition in OFDM with index modulation. In *VTC Spring*, pages 1–5. IEEE, 2019.

[93] H. Liu, X. Zhu, and T. Fujii. Ensemble deep learning based cooperative spectrum sensing with stacking fusion center. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1841–1846, November 2018.

[94] H. Liu, X. Zhu, and T. Fujii. Ensemble deep learning based cooperative spectrum sensing with semi-soft stacking fusion center. In *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2019.

[95] J. Liu, W. C. Tang, Y. Chen, M. Li, and M. Guizani. A novel crowd-sourcing inference method. In *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, pages 55–60, 2019.

[96] J. Liu, J. Wan, D. Jia, B. Zeng, D. Li, C. Hsu, and H. Chen. High-efficiency urban traffic management in context-aware computing and 5G communication. *IEEE Communications Magazine*, 55(1):34–40, 2017.

[97] Q. Liu, S. Zhou, and G. B. Giannakis. Cross-layer combining of adaptive modulation and coding with truncated ARQ over wireless links. *IEEE Transactions on Wireless Communications*, 3(5):1746–1755, 2004.

[98] X. Liu, Q. Sun, W. Lu, C. Wu, and H. Ding. Big-data-based intelligent spectrum sensing for heterogeneous spectrum communications in 5G. *IEEE Wireless Communications*, 27(5):67–73, 2020.

[99] Yi Liu, Xingliang Yuan, Ruihui Zhao, Yifeng Zheng, and Yefeng Zheng. Rc-ssfl: Towards robust and communication-efficient semi-supervised federated learning system. *arXiv preprint arXiv:2012.04432*, 2020.

[100] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[101] B. F. Lo and I. F. Akyildiz. Reinforcement learning for cooperative sensing gain in cognitive radio ad hoc networks. *Wireless Netw.*, 19:1237–1250, 2013.

[102] J. Lu, L. Li, G. Chen, D. Shen, K. Pham, and E. Blasch. Machine learning based intelligent cognitive network using fog computing. In K. D. Pham and G. Chen, editors, *Sensors and Systems for Space Applications X*, volume 10196, pages 149–157. International Society for Optics and Photonics, SPIE, 2017.

[103] Y. Lu, P. Zhu, D. Wang, and M. Fattouche. Machine learning techniques with probability vector for cooperative spectrum sensing in cognitive radio networks. In *2016 IEEE Wireless Communications and Networking Conference*, pages 1–6, April 2016.

[104] J. Lundén, V. Koivunen, S. R. Kulkarni, and H. V. Poor. Reinforcement learning based distributed multiagent sensing policy for cognitive radio networks. In *Proceedings of the 2011 IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN)*, pages 642–646, Aachen, Germany, 2011.

[105] J. Lundén, M. Motani, and H. V. Poor. Distributed algo rithms for sharing spectrum sensing information in cogni tive radio networks. *IEEE Trans. Wireless Commun.*, 14(8):4667—-78, 2015.

[106] Chuan Ma, Jun Li, Ming Ding, Howard H Yang, Feng Shu, Tony QS Quek, and H Vincent Poor. On safeguarding privacy and security in the framework of federated learning. *IEEE network*, 34(4):242–248, 2020.

[107] X. Ma, S. Ning, X. Liu, H. Kuang, and Y. Hong. Cooperative spectrum sensing using extreme learning machine for cognitive radio networks with multiple primary users. In *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pages 536–540, Oct 2018.

[108] Zhuoran Ma, Jianfeng Ma, Yinbin Miao, Ximeng Liu, Kim-Kwang Raymond Choo, and Robert H Deng. Pocket diagnosis: Secure federated learning against poisoning attack in the cloud. *IEEE Transactions on Services Computing*, 15(6):3429–3442, 2021.

[109] D. Malafaia, J. Vieira, and A. Tomé. Adaptive threshold spectrum sensing based on expectation maximization algorithm. *Physical Communication*, 21:60–69, 2016.

[110] A. Mariani, A. Giorgetti, and M. Chiani. Effects of noise power estimation on energy detection for cognitive radio applications. *IEEE Transactions on Communications*, 59(12):3410–3420, 2011.

[111] Ricardo A Maronna, R Douglas Martin, Victor J Yohai, and Matías Salibián-Barrera. *Robust statistics: theory and methods (with R)*. John Wiley & Sons, 2019.

[112] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[113] A. M. Mikaeil. Machine learning approaches for spectrum management in cognitive radio networks. In H. Farhadi, editor, *Machine Learning*, chapter 6. IntechOpen, Rijeka, 2018.

[114] A. M. Mikaeil, B. Guo, and Z. Wang. Machine learning to data fusion approach for cooperative spectrum sensing. In *In Proceedings of the 2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Shanghai, China*, pages 429–434, 13-15 October 2014.

[115] A. M. Mikaeil, B. Guo, and Z. Wang. Machine learning to datafusion approach for cooperative spectrum sensing. In *Proceedings of the 6th International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, CyberC '14*, pages 429–434, Shanghai, China, 2014.

[116] J. Mitola and G. Q. Maguire. Cognitive radio: making software radios more personal. *IEEE Personal Communications*, 6(4):13–18, 1999.

[117] M. A. Munoz, M. Rodriguez, J. Favela, A. I. Martinez-Garcia, and V. M. Gonzalez. Context-aware mobile communication in hospitals. *Computer*, 36(9):38–46, 2003.

[118] R. Mustafa, R. R. Jaglan, and S. Agrawal. Decision-fusion-based reliable CSS scheme in CR networks. *IET Communications*, 13(7):947–953, 2019.

[119] A. C. Müller and S. Guido. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media, Sebastopol, CA, USA, 2016.

[120] M. Nazzal, A. R. Ektí, A. Görçín, and H. Arslan. Exploiting sparsity recovery for compressive spectrum sensing: A machine learning approach. *IEEE Access*, 7:126098–126110, 2019.

[121] Solmaz Niknam, Harpreet S Dhillon, and Jeffrey H Reed. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Communications Magazine*, 58(6):46–51, 2020.

[122] O. Omotere, J. Fuller, L. Qian, and Z. Han. Spectrum occupancy prediction in coexisting wireless systems using deep learning. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–7, 2018.

[123] F. Paisana, A. Selim, M. Kist, P. Alvarez, J. Tallon, C. Bluemm, A. Puschmann, and L. DaSilva. Context-aware cognitive radio using deep learning. In *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pages 1–2, March 2017.

[124] Nazanin Parhizgar, Ali Jamshidi, and Peyman Setoodeh. Defense against spectrum sensing data falsification attack in cognitive radio networks using machine learning. In *2022 30th International Conference on Electrical Engineering (ICEE)*, pages 974–979. IEEE, 2022.

[125] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[126] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE Communications Surveys Tutorials*, 16(1):414–454, 2014.

[127] John W Pratt, Jean D Gibbons, John W Pratt, and Jean D Gibbons. Kolmogorov-smirnov two-sample tests. *Concepts of nonparametric theory*, pages 318–344, 1981.

[128] L. Pucker. Review of contemporary spectrum sensing technologies. Report for IEEE-SA P1900.6 Standards Group, 2010.

[129] Zhuwei Qin, Fuxun Yu, Chenchen Liu, and Xiang Chen. How convolutional neural network see the world-a survey of convolutional neural network visualization methods. *arXiv preprint arXiv:1804.11191*, 2018.

[130] Shuming Qiu, Ding Wang, Guoai Xu, and Saru Kumari. Practical and provably secure three-factor authentication protocol based on extended chaotic-maps for mobile lightweight devices. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[131] H. N. Qureshi and A. Imran. Optimal bin width for autonomous coverage estimation using MDT reports in the presence of user positioning error. *IEEE Communications Letters*, 23(4):716–719, 2019.

[132] S. Ramjee, S. Ju, D. Yang, X. Liu, A. E. Gamal, and Y. C. Eldar. Fast deep learning for automatic modulation classification. *arXiv preprint arXiv:1901.05850*, 2019.

[133] Leon Reznik. Adversarial machine learning. *Intelligent Security Systems*, pages 315–335, 2021.

[134] A. O. P. Ribas and U. S. Dias. On the double threshold energy detection-based spectrum sensing over κ-μ fading channel. In *In Proceedings of the 2015 IEEE Radio and Wireless Symposium (RWS), San Diego, CA, USA*, pages 82–85, 25-28 January 2015.

[135] Charles Richards, Sofia Khemani, and Feng Li. Evaluation of various defense techniques against targeted poisoning attacks in federated learning. In *2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, pages 693–698. IEEE, 2022.

[136] T. Riihonen, S. Werner, and R. Wichman. Optimized gain control for single-frequency relaying with loop interference. *IEEE Transactions on Wireless Communications*, 8(6):2801–2806, 2009.

[137] D. Roy, T. Mukherjee, M. Chatterjee, and E. Pasiliao. Primary user activity prediction in DSA networks using recurrent structures. In *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pages 1–10, 2019.

[138] H. Rutagemwa, A. Ghasemi, and S. Liu. Dynamic spectrum assignment for land mobile radio with deep recurrent neural networks. In *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6, 2018.

[139] Sabra Ben Saad, Bouziane Brik, and Adlen Ksentini. Toward securing federated learning against poisoning attacks in zero touch b5g networks. *IEEE Transactions on Network and Service Management*, 20(2):1612–1624, 2023.

[140] U. Salama, P. L. Sarker, and A. Chakrabarty. Enhanced energy detection using matched filter for spectrum sensing in cognitive radio networks. In *2018 Joint 7th International Conference on Informatics, Electronics Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision Pattern Recognition (icIVPR)*, pages 185–190, 2018.

[141] Q. M. Salih, M. A. Rahman, F. Al-Turjman, and Z. R. M. Azmi. Smart routing management framework exploiting dynamic data resources of cross-layer design and machine learning approaches for mobile cognitive radio networks: A survey. *IEEE Access*, 8:67835–67867, 2020.

[142] Pedro Miguel Sánchez Sánchez, Alberto Huertas Celdrán, Timo Schenk, Adrian Lars Benjamin Iten, Gérôme Bovet, Gregorio Martínez Pérez, and Burkhard Stiller. Studying the robustness of anti-adversarial federated learning models detecting cyberattacks in iot spectrum sensors. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[143] Pedro Miguel Sánchez Sánchez, Alberto Huertas Celdrán, Timo Schenk, Adrian Lars Benjamin Iten, Gérôme Bovet, Gregorio Martínez Pérez, and Burkhard Stiller. Studying the robustness of anti-adversarial federated learning models detecting cyberattacks in iot spectrum sensors. *IEEE Transactions on Dependable and Secure Computing*, 21(2):573–584, 2022.

[144] Purushothaman Saravanan, Shreeram Suresh Chandra, Akshay Upadhye, and Sanjeev Gurugopinath. A supervised learning approach for differential entropy feature-based spectrum sensing. In *2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 395–399, 2021.

[145] B. N. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *IEEE Network*, 8(5):22–32, 1994.

[146] S. Sequeira, R. R. Mahajan, and P. Spasojević. On the noise power estimation in the presence of the signal for energy-based sensing. In *In Proceedings of the 2012 35th IEEE Sarnoff Symposium, Newark, NJ, USA*, pages 1–5, 21-22 May 2012.

[147] H. A. Shah and I. Koo. Reliable machine learning based spectrum sensing in cognitive radio networks. *Wireless Communications and Mobile Computing*, 2018:5906097, 2018.

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

[149] B. S. Shawel, D. H. Woldegebreal, and S. Pollin. Convolutional lstm-based long-term spectrum prediction for dynamic spectrum access. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2019.

[150] C. She, C. Yang, and T. Q. S. Quek. Cross-layer optimization for ultra-reliable and low-latency radio access networks. *IEEE Transactions on Wireless Communications*, 17(1):127–141, 2018.

[151] Alex Shenfield, Zaheer Khan, and Hamed Ahmadi. Deep learning meets cognitive radio: Predicting future steps. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–5. IEEE, 2020.

[152] George W. Snedecor and William G. Cochran. *Statistical Methods*. Iowa State University Press, 1989.

[153] Lingyang Song and Jia Shen. *Evolved cellular network planning and optimization for UMTS and LTE*. CRC press, 2010.

[154] D. D. Z. Soto, O. J. S. Parra, and D. A. L. Sarmiento. Detection of the primary user's behavior for the intervention of the secondary user using machine learning. In *Future Data and Security Engineering*, pages 200–213, Cham, 2018. Springer International Publishing.

[155] Suvrit Sra, Sebastian Nowozin, and Stephen J Wright. *Optimization for machine learning*. Mit Press, 2012.

[156] M. Subbarao and P. Samundiswary. Automatic modulation recognition in cognitive radio receivers using multi-order cumulants and decision trees. *International Journal of Recent Technology and Engineering (IJRTE)*, 7:61–67, 11 2018.

[157] A. Subekti, H. F. Pardede, R. Sustika, and Suyoto. Spectrum sensing for cognitive radio using deep autoencoder neural network and svm. In *In Proceedings of the 2018 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET), Serpong, Indonesia, Indonesia*, pages 81–85, 1-2 November 2018.

[158] X. Sun, L. Gao, X. Luo, and K. Su. RBM based cooperative Bayesian compressive spectrum sensing with adaptive threshold. In *2016 IEEE/CIC International Conference on Communications in China (ICCC)*, pages 1–6, July 2016.

[159] Yuwei Sun, Hideya Ochiai, and Jun Sakuma. Semi-targeted model poisoning attack on federated learning via backward error analysis. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

[160] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[161] M. Tang, Z. Zheng, G. Ding, and Z. Xue. Efficient TV white space database construction via spectrum sensing and spatial inference. In *Proceedings of the 34th IEEE International Performance Computing and Communications Conference, IPCCC 2015*, pages 1–5, Nanjing, China, December 2015.

[162] Y. J. Tang, Q. Y. Zhang, and W. Lin. Artificial neural network based spectrum sensing method for cognitive radio. In *Proceedings of the 2010 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*, pages 1–4, Chengdu, China, 23–25 September 2010.

[163] C. H. A. Tavares and T. Abrão. Bayesian estimators for cooperative spectrum sensing in cognitive radio networks. In *2017 IEEE URUCON*, pages 1–4, Oct 2017.

[164] K. M. Thilina, K. W. Choi, N. Saquib, and E. Hossain. Machine learning techniques for cooperative spectrum sensing in cognitive radio networks. *IEEE J. Sel. Areas Commun.*, 31:2209–2221, 2013.

[165] J. Tian, P. Cheng, Z. Chen, M. Li, H. Hu, Y. Li, and B. Vucetic. A machine learning-enabled spectrum sensing method for OFDM systems. *IEEE Transactions on Vehicular Technology*, 2019.

[166] A. Tiwari, H. Chenji, and V. Devabhaktuni. Comparison of statistical signal processing and machine learning algorithms for spectrum sensing. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec 2018.

[167] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In *Computer security–ESORICs 2020: 25th European symposium on research in computer security, ESORICs 2020, guildford, UK, September 14–18, 2020, proceedings, part i 25*, pages 480–501. Springer, 2020.

[168] Matthew Troglia, Jordan Melcher, Yao Zheng, Dylan Anthony, Alvin Yang, and Thomas Yang. Fair: Federated incumbent detection in cbrs band. In *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pages 1–6, 2019.

[169] John W Tukey. Comparing individual means in the analysis of variance. *Biometrics*, pages 99–114, 1949.

[170] Aashma Uprety and Danda B Rawat. Mitigating poisoning attack in federated learning. In *2021 IEEE symposium series on computational intelligence (SSCI)*, pages 01–07. IEEE, 2021.

[171] P. Verma and B. Singh. Simulation study of double threshold energy detection method for cognitive radios. In *In Proceedings of the 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India*, pages 232–236, 19-20 February 2015.

[172] M. R. Vyas, D. K. Patel, and M. Lopez-Benitez. Artificial neural network based hybrid spectrum sensing scheme for cognitive radio. In *Proc. IEEE Int. Symp. Personal, Indoor, and Mobile Radio Commun. (PIMRC 2017)*, pages 1–7, Montreal, Canada, October 2017.

[173] Ding Wang and Ping Wang. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE transactions on dependable and secure computing*, 15(4):708–722, 2016.

[174] Qun Wang, Haijian Sun, Rose Qingyang Hu, and Arupjyoti Bhuyan. When machine learning meets spectrum sharing security: Methodologies and challenges. *IEEE Open Journal of the Communications Society*, 3:176–208, 2022.

[175] Weizheng Wang, Fida Hussain Memon, Zhuotao Lian, Zhimeng Yin, Thippa Reddy Gadekallu, Quoc-Viet Pham, Kapal Dev, and Chunhua Su. Secure-enhanced federated learning for ai-empowered electric vehicle energy prediction. *IEEE Consumer Electronics Magazine*, 12(2):27–34, 2021.

[176] M. Wasilewska, H. Bogucka, and A. Kliks. Federated learning for 5g radio spectrum sensing. *Sensors*, 22(1):1—-15, 2022.

[177] Małgorzata Wasilewska and Hanna Bogucka. Machine learning for LTE energy detection performance improvement. *Sensors*, 19(19):4348, 2019.

[178] Małgorzata Wasilewska and Hanna Bogucka. Space-time-frequency machine learning for improved 4G/5G energy detection. *International Journal of Electronics and Telecommunications*, 66(1):217–223, 2020.

[179] Małgorzata Wasilewska, Hanna Bogucka, and Adrian Kliks. Spectrum sensing and prediction for 5g radio. In *Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings 10*, pages 176–194. Springer International Publishing, 2021.

[180] Małgorzata Wasilewska, Adrian Kliks, Hanna Bogucka, Krzysztof Cichoń, Julius Ruseckas, Gediminas Molis, Aušra Mackutė-Varoneckienė, and Tomas Krilavičius. Artificial intelligence for radio communication context-awareness. *IEEE Access*, 9:144820–144856, 2021.

[181] Mark Weiser. The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, 3(3):3–11, July 1999.

[182] C. Y. Wong, R. S. Cheng, K. B. Lataief, and R. D. Murch. Multiuser OFDM with adaptive subcarrier, bit, and power allocation. *IEEE Journal on Selected Areas in Communications*, 17(10):1747–1758, 1999.

[183] Geming Xia, Jian Chen, Chaodong Yu, and Jun Ma. Poisoning attacks in federated learning: A survey. *IEEE Access*, 11:10708–10722, 2023.

[184] S. Xie and L. Shen. Double-threshold energy detection of spectrum sensing for cognitive radio under noise uncertainty environment. In *In Proceedings of the 2012 International Conference on Wireless Communications and Signal Processing (WCSP), Huangshan, China*, pages 1–5, 25-27 October 2012.

[185] B. Xu, X. Wen, and X. Wang. Research on modulation recognition technology based on machine learning. *International Journal of Computer Applications Technology and Research*, 8:102–106, 04 2019.

[186] G. Xu, S. Gao, M. Daneshmand, C. Wang, and Y. Liu. A survey for mobility big data analytics for geolocation prediction. *IEEE Wireless Communications*, 24(1):111–119, 2017.

[187] Jie Xu and Heqiang Wang. Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective. *IEEE Transactions on Wireless Communications*, 20(2):1188–1200, 2021.

[188] Y. Xu, P. Cheng, Z. Chen, Y. Li, and B. Vucetic. Mobile collaborative spectrum sensing for heterogeneous networks: A Bayesian machine learning approach. *IEEE Trans. Signal Process.*, 66(21):5634–5647, 2018.

[189] H. Xue and F. Gao. A machine learning based spectrum-sensing algorithm using sample covariance matrix. In *Communications and Networking in China (ChinaCom), 2015 10th International Conference on.*, pages 476–480, 2015.

[190] H. Yang, X. Xie, and R. Wang. SOM-GA-SVM detection based spectrum sensing in cognitive radio. In *Wireless Communications, Networking and Mobile Computing (WiCOM), 2012 8th International Conference on*, pages 1–7, 2012.

[191] K. Yang, Z. Huang, X. Wang, and X. Li. A blind spectrum sensing method based on deep learning. *Sensors*, 19:2270, 2019.

[192] P. Yang, M. Di Renzo, Y. Xiao, S. Li, and L. Hanzo. Design guidelines for spatial modulation. *IEEE Communications Surveys Tutorials*, 17(1):6–26, 2015.

[193] Zhaohui Yang, Mingzhe Chen, Walid Saad, Choong Seon Hong, and Mohammad Shikh-Bahaei. Energy efficient federated learning over wireless communication networks. *IEEE Transactions on Wireless Communications*, 20(3):1935–1949, 2021.

[194] L. Yu, J. Chen, and G. Ding. Spectrum prediction via long short term memory. In *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pages 643–647, 2017.

[195] L. Yu, J. Chen, G. Ding, Y. Tu, J. Yang, and J. Sun. Spectrum prediction based on Taguchi method in deep learning with long short-term memory. *IEEE Access*, 6:45923–45933, 2018.

[196] L. Yu, Y. Guo, Q. Wang, C. Luo, M. Li, W. Liao, and P. Li. Spectrum availability prediction for cognitive radio communications: A dcg approach. *IEEE Transactions on Cognitive Communications and Networking*, 6(2):476–485, 2020.

[197] L. Yu, Q. Wang, Y. Guo, and P. Li. Spectrum availability prediction in cognitive aerospace communications: A deep learning perspective. In *2017 Cognitive Communications for Aerospace Applications Workshop (CCAA)*, pages 1–4, 2017.

[198] T. Yucek and H. Arslan. A survey of spectrum sensing algorithms for cognitive radio applications. *IEEE Communications Surveys Tutorials*, 11(1):116–130, First 2009.

[199] K. Zhang, J. Li, and F. Gao. Machine learning techniques for spectrum sensing when primary user has multiple transmit powers. In *2014 IEEE International Conference on Communication Systems*, pages 137–141, November 2014.

[200] L. Zhang, H. Huang, and X. Jing. A modified cyclostationary spectrum sensing based on softmax regression model. In *2016 16th International Symposium on Communications and Information Technologies (ISCIT)*, pages 620–623, Sep. 2016.

[201] 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 Transactions on Communications*, E101.B(12):2435–2444, 2018.

[202] M. Zhang, Y. Zeng, Z. Han, and Y. Gong. Automatic modulation recognition using deep learning architectures. In *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5, June 2018.

[203] N. Zhang, X. Sun, C. Guo, and L. Gao. PU probability prediction based Bayesian compressive spectrum sensing. In *2015 IEEE/CIC International Conference on Communications in China (ICCC)*, pages 1–5, Nov 2015.

[204] Zhongyuan Zhao, Chenyuan Feng, Wei Hong, Jiamo Jiang, Chao Jia, Tony Q. S. Quek, and Mugen Peng. Federated learning with non-iid data in wireless networks. *IEEE Transactions on Wireless Communications*, pages 1–1, 2021.

[205] Haibin Zheng, Tao Liu, Rongchang Li, and Jinyin Chen. Poe: Poisoning enhancement through label smoothing in federated learning. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 70(8):3129–3133, 2023.

[206] S. Zhou, Z. Yin, Z. Wu, Y. Chen, N. Zhao, and Z. Yang. A robust modulation classification method using convolutional neural networks. *EURASIP Journal on Advances in Signal Processing*, 2019(1):21, Mar 2019.

[207] Z. Zhou, G. Ma, M. Dong, K. Ota, C. Xu, and Y. Jia. Iterative energy-efficient stable matching approach for context-aware resource allocation in D2D communications. *IEEE Access*, 4:6181–6196, 2016.

[208] Q. Zhu, Z. Han, and T. Başar. No-regret learning in collaborative spectrum sensing with malicious nodes. In *Proceedings of the 2010 IEEE International Conference on Communications*, pages 1–6, Cape Town, South Africa, 2010.