Palestine Polytechnic University

Deanship of Graduate Studies and Scientific Research

Master of informatics

Performance Evaluation of Machine Learning-Based Techniques for Spectrum Sensing in Mobile Cognitive Radio Networks

Submitted by:

Sundous Khamayseh

Supervisor:

Dr. Murad Abusubaih

Thesis submitted in partial fulfillment of requirements

of the degree Master of Science in Informatics

September, 2021

The undersigned hereby certify that they have read, examined, and recommended to the Deanship of Graduate Studies and Scientific Research at Palestine Polytechnic University the approval of a thesis entitled: **Performance Evaluation of Machine Learning-Based Techniques for Spectrum Sensing in Mobile Cognitive Radio Networks**, submitted by **Sundous Khamayseh** in partial fulfillment of the requirements for the degree of Master in Informatics.

Graduate Advisory Committee:

Dr. Murad Abusubaih (Supervisor), Palestine Polytechnic University.

Signature:_____ Date:_____

Dr. Alaa Halawani (Internal committee member), Palestine Polytechnic University.

Signature:_____ Date:_____

Dr. Samer Bali (External committee member), Al-Zaytona University of Science and Technology.

Signature:_____ Date:_____

Thesis Approved

Dr. Nafez Naser Aldeen
Dean of Graduate Studies and Scientific Research
Palestine Polytechnic University

Signature:_____ Date:_____

DECLARATION

I declare that the Master Thesis entitled **Performance Evaluation of Machine Learning-Based Techniques for Spectrum Sensing in Mobile Cognitive Radio Networks** is my original work, and hereby certify that unless stated, all work contained within this thesis is my own independent research and has not been submitted for the award of any other degree at any institution, except where due acknowledgement is made in the text.

Sundous Khamayseh

Signature:_____ Date:_____

STATEMENT OF PERMISSION

TO USE

In presenting this thesis in partial fulfillment of the requirements for the master degree in Informatics at Palestine Polytechnic University, I agree that the library shall make it available to borrowers under rules of the library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgement of the source is made.

Permission for extensive quotation from, reproduction, or publication of this thesis may be granted by my main supervisor, or in his absence, by the Dean of Graduate Studies and Scientific Research when, in the opinion of either, the proposed use of the material is for scholarly purposes.

Any coping or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Sundous Khamayseh

Signature:_____ Date:_____

***

الملخص

يتطور قطاع الإتصالات يوما بعد يوم، ولا يخفى على أحد مدى التطور في مجال الإتصالات وتقنياتها. تؤدي هذه التطورات بحكم الضرورة إلى زيادة الطلب على تقنيات الإتصال الذي يؤدي بدوره أيضا إلى تخصيص المزيد والمزيد من قنوات الإتصال بين الجهات المرسلة والمستقبلة. ومع ذلك بدأت تظهر بعض المشاكل كمشكلة نقص الطيف المتاح مع الإستخدام المتسارع لتقنيات الإتصال المختلفة، والإبقاء على طرق الوصول التقليدية في تخصيص قنوات الإتصال، وهو ما يعرف بسوء تخصيص الطيف وأزمة الترددات.

Cognitive Radio (CR) هو حل مبتكر تم تطويره لتجنب العقبات المذكورة أعلاه، حيث أنه يتفاعل مع البيئة المحيطة لتحديد معلمات الإتصال المناسبة، تمر دورة حياة الـ CR بالعديد من المهام ويعد إستشعار الطيف المهمة الرئيسية في هذه الدورة تكتسب أهميتها من أنه يمكن خلالها إكتشاف مناطق الطيف غير المشغولة. قد تحدث مهمة إستشعار الطيف في وضع Non-Cooperative mode أو وضع Cooperative mode يتعاون فيه المستخدمون الثانويون معا لتحديد حالة قناة الإتصال. جذب نشر تقنيات الاستشعار القائمة على التعلم الالي اهتمام الباحثين في السنوات الاخيرة. الفكرة هي تزويد الشبكة ببعض الذكاء لتعزيز عملية استشعار الطيف وبالتالي امكانية الكشف الدقيق عن نشاط المستخدم الاساسي ومن الأمثلة على هذه التقنيات تقنية Energy Detection-based technique، تقنية Covarience Matrix-based technique وتقنية Machine Learning-based technique.

في هذه الرسالة ندرس ونقارن أداء تقنيات الإستشعار غير التعاونية Non-Cooperative spectrum sensing technique، تقنية And-based spectrum sensing technique، تقنية Or-based spectrum sensing technique، وتقنية KMeans-based spectrum sensing technique. على غرار غالبية الأبحاث المنشورة، نختبر كذلك أداء تقنيات إستشعار الطيف في شبكات Mobile CR network (CRN). أيضا، نحاول فهم تاثير نوع قنوات الخبو على أداء عملية الإستشعار. بالإضافة إلى ذلك، نحاول البحث عن المعلمات المثلى التي تعمل على تحسين تقنيات الإستشعار مثل نوع

قناة الخبو وعدد المستخدميين الثانويين وعدد العينات المأخوذة. أخيرا، نحقق في الظروف تجعل تقنيات ال Machine learning-based techniques تتفوق على تقنيات الإستشعار التقليدية.

تمت محاكاة شبكات ال Stationary CRN و Mobile CRN بإستخدام الإصدار الثالث من محاكي الشبكة (ns3 simulation). أظهرت النتائج تأثير قناة الإتصال على أداء تقنيات الإستشعار المختلفة حيث أن الظروف الصعبة كالضوضاء و الخبو الحاصل للإشارات المرسلة أدت إلى إنخفاض الأداء بشكل ملحوظ. كما أظهرت النتائج بشكل عام أن الـ Stationary senario تفوقت على الـ Mobile senario في نفس الظروف. كذلك أظهرت النتائج أننا بحاجة إلى أكثر من ٣ مستخدميين ثانويين و ١٥٠٠ عينية للوصول إلى مستوى أداء إستكشافي مقبول. علاوة على ذلك، أظهرت النتائج أن KMeans-based spectrum sensing technique تتفوق على الـ Or-based spectrum sensing technique في البيئات عالية الضوضاء عندما تتواجد قيم عالية لمستوى الضوضاء و تلاش شديد في قوة اشارة المستخدم الأساسي المستقبلة.

Abstract

Communication technologies evolve drastically in recent years. These recent developments in communication technologies inevitably lead to increasing demand for communication technologies. They also lead to allocating progressively communication channels between transmitting and receiving elements. However, the scarcity of spectrum began to appear with the accelerating rise in the usage of various communication technologies and the preservation of traditional channel access methods. That is what is called the spectrum under-utilization and frequencies crisis.

Cognitive Radio (CR) is an innovative solution developed to avoid these obstacles mentioned above. It deals with the surrounding environment and determines the appropriate communication parameters. The CR life cycle goes through many tasks. Spectrum sensing is a key task in this cycle that gains significance where the spectrum holes can be detected during this task. Spectrum sensing task is needed in both **non-cooperative spectrum sensing (Non-CSS)** or **cooperative spectrum sensing (CSS)** modes, whereby the secondary users (SUs) cooperate to determine channel state. The deployment of machine learning-based spectrum sensing techniques in CR networks has been attracting researchers in recent years. The idea is to provide the network with some intelligence to enhance the spectrum sensing process and thus the possibility of accurately detecting PU activity. Examples of these techniques are the **energy detection-based**, the **covariance matrix-based**, and **Machine learning-based** techniques.

In this thesis, firstly we study and compare the performance of the **Non-CSS**, the **And-based**, the **Or-based** techniques, as well as the **KMeans-based ML technique** in stationary CRNs. In contrast to the majority of published research, we examine the performance of that mentioned spectrum sensing techniques in mobile CRNs. Also, we try to grasp the effect of the fading channels on the sensing performance. Moreover, we try to find the optimal parameters that can improve the performance of various spectrum sensing techniques. Finally, we investigate the circumstances in

which KMeans-based spectrum sensing techniques introduce superior to traditional techniques.

Stationary and mobile CRNs were simulating using the third version of the network simulator (ns3 simulation). The result showed the effect of the communication channel on the spectrum sensing performance. Difficult conditions such as noise and fading effects on the transmitted signals lead to a notable decrease in sensing performance. Also, the results generally revealed that spectrum sensing techniques provide better performance in stationary networks. However, the results showed we need above three SUs and about 1500 samples to reach an acceptable performance level in mobile CRN. In addition, the results showed that the **KMeans-based technique** slightly outperforms the **Or-based technique**, especially in highly-noisy environments and under severe fading channels.

DEDICATION

To my mother and father who gave me all the support and always encouraged me to get my master's degree,
for those who have positive souls and they spread happiness and hope around them,
for all the lively honest people who do their works with sincerity,
for my beautiful country,
I dedicate this work.

. . .

ACKNOWLEDGEMENT

I greatly appreciate the thesis's supervisor Dr. Murad Abusubaih for his unlimited assistance, support, and sincerity throughout his supervision journey on my master thesis.

I would like to thank all of the doctors, Dr. Alaa Halwani, Dr. Ghandi Manasra, and Dr. Liana Tamimi for their willingness to help and their valuable advice in previous stages of my master's program.

I would like to thank the thesis's examiners committee, the administrators, and academics working at the Palestine Polytechnic University.

Also, I would like to thank my colleagues and everyone who directly or indirectly helped me during my master's program.

Table of contents

1 Introduction **1**

1.1 Problem formulation 1

1.2 Thesis Motivation 2

1.3 Contributions 3

1.4 Thesis Organization 3

2 Background **4**

2.1 Propagation of Radio Waves 4

2.1.1 Free space losses 4

2.1.2 Multipath propagation 5

2.2 Path-loss Propagation models 5

2.2.1 Log-normal distribution 6

2.2.2 Nakagami distribution 6

2.2.3 Ricean Distribution 6

2.2.4 $\kappa - \mu$ Distribution 7

2.3 Cognitive radio technology 8

2.3.1 CR life cycle 9

2.3.2 Spectrum sensing task in CRNs 10

2.3.3 Spectrum sensing types 10

2.4 Machine learning 11

2.4.1 Main categories of ML algorithms 11

2.4.2 ML-based CSS: General model 13

3 Literature Review **14**

3.1 ML-based CSS 14

3.1.1 CSS system with Multi-classes 14

3.2 Feature extraction methods 15

3.2.1 ED-based schemes 15

3.2.2 Covariance matrix-based schemes 16

3.2.3 Signal processing-based schemes 17

3.3 Data combining and ML 18

3.3.1 Logical And-based/Or-based CSS 18

3.3.2 Unsupervised-based CCS 19

3.3.3 Supervised-based CSS 21

3.3.4 Reinforcement CCS 23

4 System Modeling & The Experimental Setup **25**

4.1 System modeling 25

4.2 The simulation & noise modeling 26

4.2.1 The general $\kappa - \mu$ fading distribution 27

4.2.2 The noise Model 28

5 Results and & discussion **30**

5.1 Stationary CRN vs. Mobile CRN 30

5.2 $\kappa - \mu$ setup 32

5.3 Noisy environment 35

6 Conclusion & Future Work **41**

Appendices **44**

A Related topics **45**

A.1 Ns3 implementation of general $\kappa - \mu$ fading 45

A.1.1 Introduction 45
A.1.2 Model implementation 45
A.2 Sampling methods 49
A.3 Receiver operating characteristic curve 50

B Code synopsis **52**
B.1 ns3 Code synopsis 52
B.1.1 Source code: The $\kappa - \mu$ general distribution 52
B.1.2 Source code: Rejection-sampling method 53
B.1.3 Certainly factor 54
B.2 Python Code synopsis 54
B.2.1 Source code: Noise Modling 54
B.2.2 Source code: KMeans algorithm 55
B.2.3 Some of CSS techniques source codes 56
B.2.4 ROC Curve source code 57

List of Figures

2.1 Fading distribution comparison [1] 7
2.2 The general $\kappa - \mu$ fading distribution. 8
2.3 CR life cycle [2] 9
2.4 Explanation of the spectrum sensing task [3] 11
2.5 Reinforcement ML approach 12

4.1 System Modeling 26

5.1 StaCR (left) vs. MobCR (right) of Rayleigh fading 31
5.2 The performance of the various technique under the effect of the different fading channels 33
5.3 Comparison the effect of different fading channels on the performance of the KMeans-based technique 34
5.4 The performance of the various technique under the effect of different noisy environments 36
5.5 Data Clustering for different environments, [Rician, M=1500s] 37
5.6 StaCR vs MobCR of KMeans-base CSS for different noisy environments, [Rayleigh, N=5SUs, M=1500s] 39

A.1 The drawn samples from the KappaMuGeneralPropagation-LossModel of different values of κ, μ. [$\kappa \to 0$ with $\mu = 0.5, 1.0, \&3.0$], [$\mu = 1.0$ with $\kappa = 0.75, 1.0, \&4.0$] 46
A.2 The Effect of the power on the yielded data 48

A.3 The reject-accept algorithm 49
A.4 The Good/Bad ROC curve 51

List of source codes

B.1 $\kappa - \mu$ general distribution, GetTypeId 52
B.2 $\kappa - \mu$ general distribution, accept-reject method 53
B.3 Certainly factor 54
B.4 Noise Modling 54
B.5 Classification process of KMeans algorithm 55
B.6 And/Or-based CSS techniques 56
B.7 KMeans-based CSS techniques 56
B.8 ROC Curve 57

List of Tables

4.1 $\kappa - \mu$ values for different fading channel types 27
4.2 The different noise conditions 29

5.1 Summary of numerical results for the noisy environments
Env 1/1 and Env2/1, [N=5SUs, M=1500s] 38

List of Abbreviations

CR	Cognitive Radio
CRN	Cognitive Radio Network
SS	Spectrum Sensing
Non-CSS	Non-Cooperative Spectrum Sensing
CSS	Cooperative Spectrum Sensing
StaCR	Stationary Cognitive Radio
MobCR	Mobile Cognitive Radio
ML-based CSS	Machine Learning-based Cooperative Spectrum Sensing
PU	Primary User
SU	Secondary User
FC	Fusion Center
CH	Cluster Head
H_0	The null hypothesis
H_1	The alternative hypothesis
FA	False Alarm
MD	Missed Detection
P(FA)	Probability of False Alarm
P(MD)	Probability of Missed Detection
P(D)	Probability of Detection

***

Chapter ▷1

Introduction

Cognitive radio technology (CR) is a promising solution for the spectrum scarcity problem. The reconfigurability feature of CR indicates the ability of CR technology to switch between radio access methods, as well as, transmitting in different portions of the radio spectrum. However, the reconfigurability of the CR requires a dynamic spectrum management framework (DSMF) [4]. DSMF framework consist of cognition tasks that, in general, are: sensing the spectrum, analyzing the spectrum, making joint decisions on spectrum selections, and channel clearance when a licensed user need to use it.

Spectrum sensing is the first task of the cognitive radio life cycle that gains significance since the spectrum holes can be detected during this task. There are a plethora of works of spectrum sensing techniques on CR networks (CRNs). Most of these types are the energy detection-based technique, the cyclostationary matrix-based technique, and the covariance-based technique. Machine learning-based techniques are another modern type of innovated spectrum sensing technique. In such methods, the sensing process in detecting the primary user's activities passes through two phases which are: I- the feature extraction phase and II- the decision-making phase.

1.1 Problem formulation

Spectrum sensing is a key task in the CR life cycle. In this task, CR learns and perceives the surrounding environment, then detects the spectrum holes. Spectrum sensing was first formulated as a binary hypothesis test for radar signal detection [5]. Later, the same binary hypothesis test is used to describe the general spectrum sensing problem, as can be seen in equation (1.1),

$$y(k) = \begin{cases} w(k), & H_0 \\ \alpha * s(k) + w(k), & H_1 \end{cases} \tag{1.1}$$

where *s(k)* is the PU transmitted signal and *y(k)* is the PU signal received by the SU side under the ambient noise w*(k)*. *k=1*,..., to *K* are the signal samples that were received by SU and the parameter α denotes the channel's fading coefficient. The SU examines the PU signal existences and the channel is considered **idle** under the null hypothesis (H_0) condition and **busy** under the alternative hypothesis (H_1) condition.

The detection of the PU signal faces two types of detection errors. Type (I) error is a false alarm (FA), the probability of the decision making indicating the PU does exist while it doesn't exist. Type (II) error is the missed detection (MD), which means the probability of the decision making indicates the PU doesn't exist while it does exist. The IEEE 802.22 [6] workgroup recommended that the P(FA) should not exceed 10%. Further, the probability of detection P(D) (the probability of the true detecting the PU signal, i.e., $1-$P(MD)) should be higher than 90%.

1.2 Thesis Motivation

Examining several approaches of spectrum sensing techniques, we noticed that almost in all the spectrum sensing techniques, the authors considered stationary cognitive radio systems, i.e., the primary users (PUs) and the secondary users (SUs) are fixed and not mobile. However, this is not always the case. The only study which addressed the problem of detecting the activity of PUs under the mobility condition is Y. Xu et al.'s study [7]. This study considered the energy vector methods and the non-parametric hidden Markov model that make the ML-based CSS more complex. However, this study doesn't regard the fading effects. **The lack of studies considering the mobility, with the effects of the fading, during the sensing process motivated us to evaluate the performance of the machine learning-based cooperative spectrum sensing techniques (ML-based CSS techniques) in the mobile networks as well.** We wanted to understand and determine scenarios in which ML-based approaches would help in enhancing decision-making processes.

1.3 Contributions

The main goal of this thesis is **to evaluate the performance of detection of ML-based CSS techniques in mobile cognitive radio networks in general κ_μ fading channels.** For that, we used the receiver operating characteristic curve (ROC curve) that measures the probability of detection against the probability of false alarm at various threshold settings. The KMeans clustering approach is considered for its simplicity and practicality of implementation. Further, we try to develop and assess solution ideas that may sustain the performance[1] of the machine learning-based approaches in non-stationary scenarios.

1.4 Thesis Organization

Chapter 2 describes the spectrum sensing problem, theories, and the basic concepts needed to understand the rest of the thesis. Chapter 3 provides a summary of some previous related works. Chapter 4 covers the methodology used in the modeling, simulating, and aggregating data sets to implement and enhance the accuracy. Chapter 5 demonstrates experiments and discusses the results of this work. Finally, Chapter 6 concludes the work and proposes some of the new directions for future research.

[1] From here to the rest of this thesis, we mention the term "performance" alone, but of course, we mean "performance of detection."

Chapter ▷2

Background

This chapter provides a quick overview of radio signal propagation and briefly discusses some proposed fading models. This chapter also explains cognitive radio technology, provides a brief discussion of its life cycle, and discuss spectrum sensing task. Additionally, this chapter will briefly discuss machine learning and related concepts.

2.1 Propagation of Radio Waves

In wireless communication, the radio signal emitted by an antenna travels through space (atmosphere and water) towards the receiver. The radio waves, like electromagnetic waves, have physical characteristics when acting with the surrounding environment. Therefore, the wireless radio signal is constantly deteriorating as it travels over long distances [8]. And thus, several versions of the transmitted signal arrive at the receiver at different time delays. Noise is one of the most common causes of signal attenuation. It is imposed by the thermal agitation of electrons of the transmission system itself. Irregular pulses or impulses noise generated from external electromagnetic disturbances. Crosstalk, or any other unwanted signals that interfere with the transmitted signal, are several forms of noise [9]. Free space losses related to the distance traveled; losses due to scattering of radio waves in the atmospheric layer are also causes of attenuation.

2.1.1 Free space losses

All wireless technologies suffer from some level of free space loss. The value of free space losses depends, mainly, on the transmission distance, frequencies, antenna dimensions, and separation [10]. Free space loss, in decibel, is described by the following equation 2.1

$$L_{dB} = -20\log(f) + 20\log(d) - 10\log(A_t A_r) + 169.54 \tag{2.1}$$

where f is the carrier frequency, d is the transmission distance, and A_t, A_r are the effective area of the transmitting and receiving antennas, respectively.

2.1.2 Multipath propagation

As mentioned earlier, radio signals have many electromagnetic behaviors due to the influence of scattered objects in the surrounding environment. Three common electromagnetic properties that describe radio wave propagation are; reflection, diffraction, and scattering. Reflection and diffraction are phenomena when radio waves encounter a relatively large surface compared to the signal wavelength. Some waves are reflected by a value according to the angle of incidence, the reflecting medium, and the polarization of the electric field. Other waves, specifically, those that meet the edges of the obstacles, bend in a different transmission direction [10]. Radio waves are scattered when they hit rough surfaces or obstacles having sizes that are, on the order or less of, the wavelength of the signal.

Signal fading

Due to the multipath effect, several versions of the transmitted signal arrive at the receiver at different time delays. This time variation of the received signal power is known as fading. The types of fading are classified into large-scale fading and small-scale fading. In the frequency domain, fading is classified as flat fading and selective fading. On the other hand, small-scale fading due to transmitter movement and transmitting area size can be classified into fast-scale and slow-scale fading. Fast fading refers to rapid changes in signal strength as it propagates over a short distance of about half the signal wavelength [11]. In contrast, slow fluctuations of signal strength facing different conditions due to its propagation over a long distance are known as slow fading [8, 10, 12].

2.2 Path-loss Propagation models

A large number of distributions have been proposed in an attempt to find the best model that approximates the distributions of different types of fading. In general, the large-scale fading signal is well characterized by the log-normal distribution. While small-scale signal variation is described by several other distributions, such as the Rayleigh, Rice, Nakagami-m, Hoyt, and Weibull

distributions [12]. In most cases, Nakagami-m and Rayleigh fadings are described by the gamma distribution whereas the Rice fading is described by the zero-order modified Bessel function of the first kind [12–14].

2.2.1 Log-normal distribution

The log-normal distribution is usually used to describe a medium-scale fading that is caused when the signal is shaded by local obstacles. The Log-normal indicates that the probability density function of the signal power is taken in the decibel unit. Thus, with Log-normal distribution, the signal power is a Gaussian random variable defined as equation 2.2 depicts

$$f_{(dBm)}(w) = \frac{1}{\sigma\sqrt{2\pi}} \exp\big(-\frac{(w-\mu)^2}{2\sigma^2}\big) \tag{2.2}$$

where w is the signal power in dBm ($10\log(w_{mW})$; w_{mW} is the signal power in Millie Watt). The μ and the σ are the mean value and the standard deviation of w.

2.2.2 Nakagami distribution

Let r be the signal envelop. Then, Nakagami-m distribution is given by the probability density function in the equation 2.3

$$f(r) = \frac{2m^m}{\Gamma(m)w^m} r^{2m-1} \exp\big(-\frac{m}{w}r^2\big), \quad r \geq 0 \tag{2.3}$$

where m is the fading depth parameter. The spread of the distribution can be controlled by the average signal power w [15]. The Rayleigh distribution can be obtained by substituting the parameter m of the equation 2.3 by 1 as the next equation 2.4 depicts

$$f(r) = \frac{2r}{w} \exp\big(-\frac{r^2}{w}\big), \quad r \geq 0 \tag{2.4}$$

2.2.3 Ricean Distribution

The formula of the envelop probability density of Ricean distribution is defined as [10]:

$$f(r) = \frac{r}{\sigma^2} \exp\big(\frac{r^2+A^2}{2\sigma^2}\big) I_0\big(\frac{Ar}{\sigma^2}\big), \quad r \geq 0 \tag{2.5}$$

where A is the peak amplitude of the dominant signal and controls the mean of the distribution. The σ is the standard deviation and $I_0(.)$ is the modified Bessel function of the first kind and zero-order. Figure 2.1 gives a comparative analysis that depicts how much various fading distributions best fit the cellu-

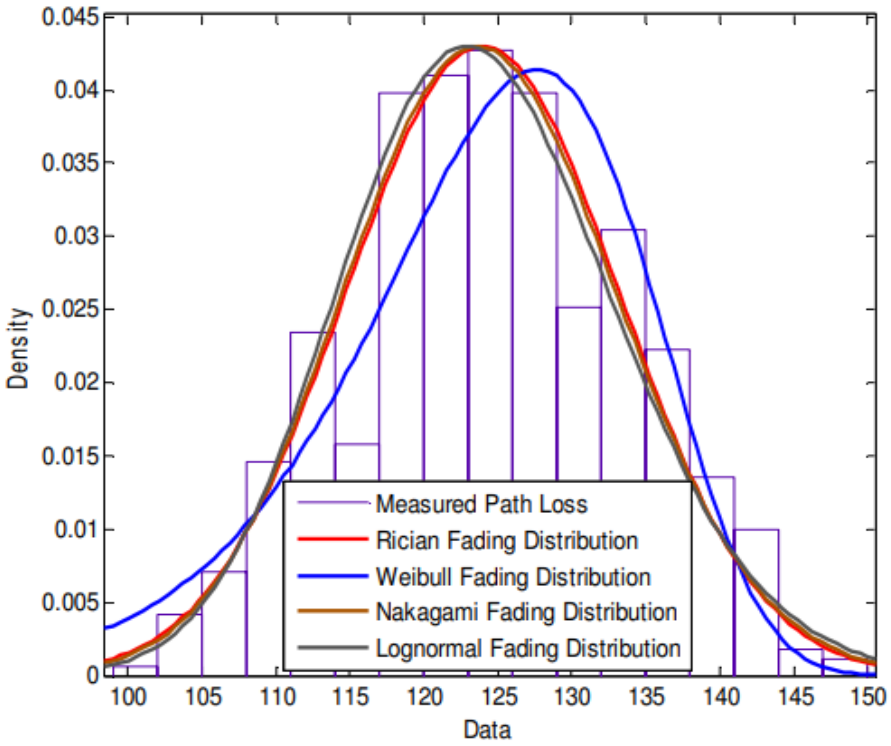


Figure 2.1: Fading distribution comparison [1]

lar signals was collected from the Accra central market [1]. The figure shows that the Rician, Nakagami-m, and log-normal distribution gave good agreement with data but the Weibull distribution gave little agreement. Hence, The quantified shadow loss distribution may follow a log-normal distribution.

2.2.4 $\kappa - \mu$ Distribution

Yacoub et al. [16] introduced a general $\kappa - \mu$ fading model that derives these common types of fading by tuning the κ and μ parameters. Rayleigh fading can be obtained as κ approaches 0 and μ equals 1. Different types of Rician fading can be modeled by fixing the μ parameter and tuning κ. Also, different types of Nakagami fading can be modeled by fixing the κ parameter and tuning the μ parameter. The power probability density function of the general $\kappa - \mu$ fading is described by the following equation 2.6

$$f(w) = \frac{\mu(1+\kappa)^{\frac{\mu+1}{2}}}{\kappa^{\frac{\mu-1}{2}}\exp(\kappa\mu)} w^{\frac{\mu-1}{2}} \exp\big(-\mu(1+\kappa)w\big) \\ I_{\mu-1}\big(\mu\sqrt{\kappa(1+\kappa)w}\big), \quad (\kappa,\mu) > 0 \tag{2.6}$$

where $\kappa > 0$ is the ratio of the total power of the dominant components to the total power of the scattered waves and $\mu = \frac{1+2\kappa}{(1+\kappa)^2}, \mu > 0$. Ω is the normalized power of the fading signal, w is the instantaneous fading signal, and $I_v(.)$ is

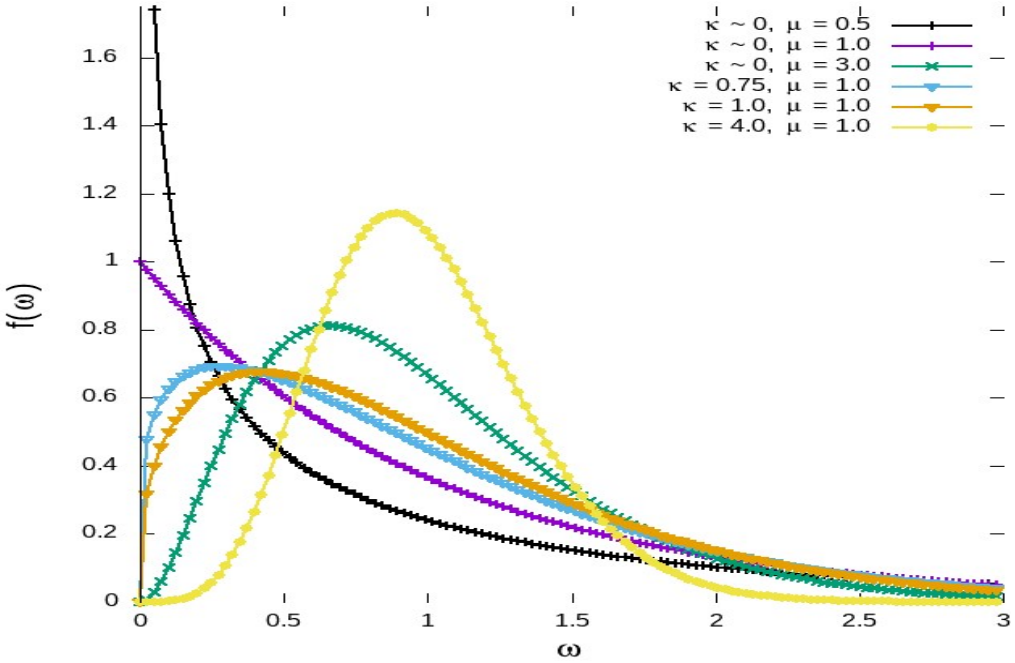


Figure 2.2: The general $\kappa - \mu$ fading distribution.

the modified Bessel function of the first kind and order v. Figure 2.2 shows different power probability density distributions for different $\kappa - \mu$ settings.

2.3 Cognitive radio technology

There are numerous methods of channel access in wireless networks. The foremost methods divide the communication channel according to the band of allocated frequencies (the frequency division multiple access (FDMA)), the geographical regions (space division multiple access (SDMA)), the time (the time division multiple access (TDMA)), or the code that is assigned for each user (the code division multiple access (CDMA)) [17]. Although these methods make bandwidth utilization more efficient, they result in spectrum scarcity.

Cognitive radio (CR) is a promising solution for the spectrum scarcity problem. It is defined as an intelligent radio able to be aware of the surrounding environment and know the used frequencies. One of the most notable features of cognitive radio networks (CRN) is the ability to switch between radio access methods, as well as, the ability to transmit in different portions of the radio spectrum [18]. There are two types of users operate in CRN

- The primary users (PUs) are the licensed users who have a license to use a part of the spectrum. However, PUs are not granted exclusive use of that part of the spectrum, they are rather given a higher priority than

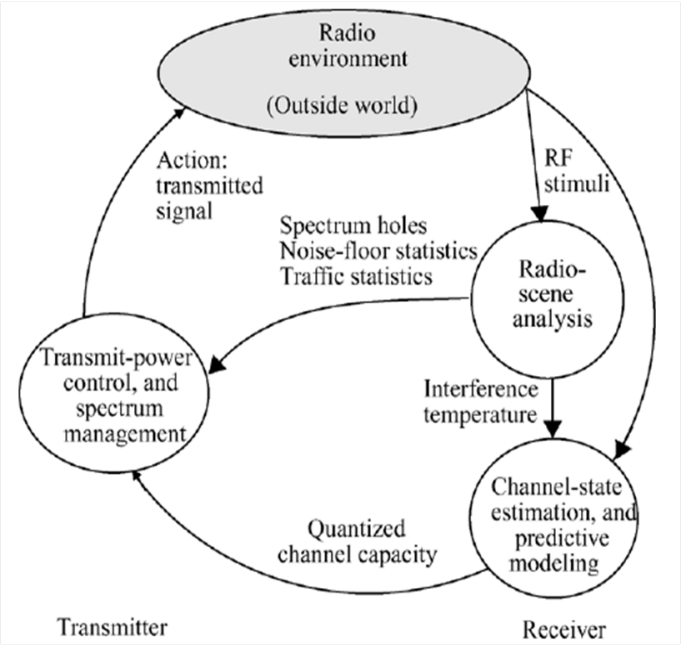


Figure 2.3: CR life cycle [2]

other users as well as safeguards against interference.

- Secondary Users (SUs) are users who have a chance to use part of the spectrum along with the PUs. However, this use is conditional on the main PUs activities, and when they don't temporarily use the spectrum.

2.3.1 CR life cycle

CR aims to make the overall radio spectrum usage more efficient by optimal utilization of the large portions of the licensed frequencies that remain vacant at certain times and locations. CR passes through cycles that provide it several capabilities to act with the surrounding environment and determine the appropriate communication parameters [19, 20].

The CR life cycle [2] encompasses three main tasks which are: radio-scene analysis task, Channel identification task, and Transmit-power control, and dynamic spectrum management task. In general, CR technology is mainly interested in exploring the spectrum holes. It interacts with the surrounding environment, measures the interference temperature[1], then determines the channel's availability states. Figure 2.3 summarizes the main three tasks in the CR life cycle.

[1]The interference temperature is a model to control interference at the receiver through the certain metric called the interference temperature limit

2.3.2 Spectrum sensing task in CRNs

Spectrum sensing is a critical task in the life cycle of a CR. In this task, CR can learn and conscious the surrounding environment, then detect the spectrum holes. The binary hypothesis test was first formulated for radar signal detection [5]. Later, the same test is used to describe the general spectrum sensing problem. Detection of PU signals encounters two types of detection errors. Type I, false alarm (FA); the decision made indicates the PU does exist while it does not exist. Type II, is the missed detection (MD); that the decision made indicates the PU does not exist while it exists. The two possibilities, false alarm probability (P(FA)) and missed detection probability (P(MD)) estimate the two types of detection errors mentioned above (type I and II, respectively). However, if type (I) occurs frequently, SU loses its chance to use the communication channel, whereas when type (II) occurs frequently, the collision is inevitable between PU and SU [5].

2.3.3 Spectrum sensing types

The spectrum sensing task may occur in non-cooperative or cooperative modes. During the sensing duration, each SU individually determines the channel state in the non-cooperative spectrum sensing (Non-CSS). For example, figure 2.4 shows that SU determines that the third and the fourth channels are ideal during the first sensing duration. While the second and the third channels are ideal in the next sensing duration and so on. The Non-CSS mode is considered suitable when nodes are unable to share their spectrum sensing information. In cooperative spectrum sensing (CSS), SUs cooperate to determine the channel state. CSS provides high accuracy, whereby all SUs participate in determining a general decision about the channel state. There are two types of typologies of CSS;

- Centralized CSS: In this typology, spectrum allocation and access can be controlled by a central entity called fusion center (FC). Here, all SUs aggregate the information about PU signal existence via the sensing channel, then send back the sensing information towards the FC over the reporting channel. Later, FC creates a general decision about the channel state. Although this typology is useful in spatial diversity networks,

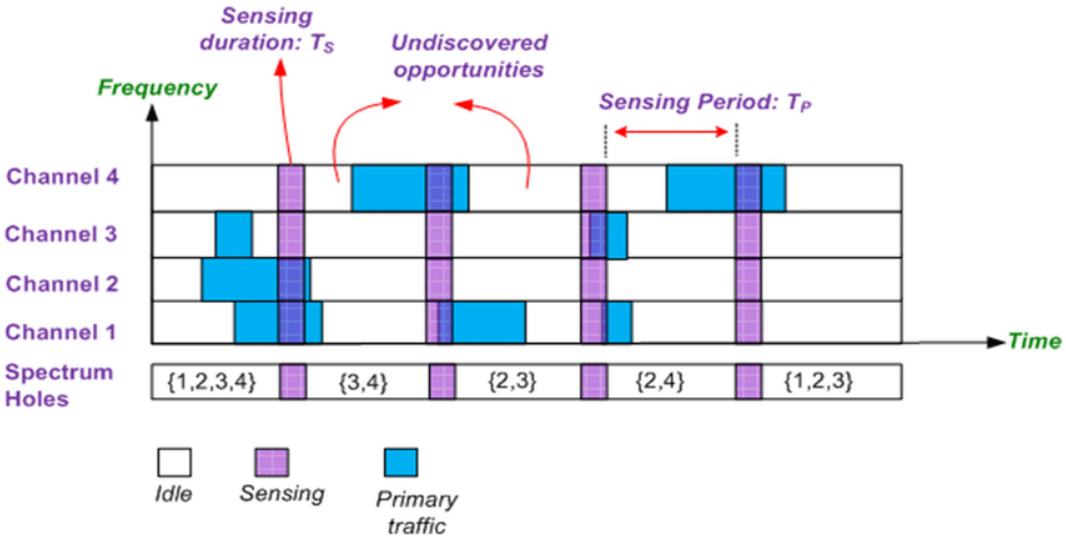


Figure 2.4: Explanation of the spectrum sensing task [3]

FC endures high overhead when determining the final sensing decision. Further, this topology will face the single node failure problem.

- Distributed CSS: In this typology, a set of adjacent SUs form a cluster then, they elect one node as a cluster head (CH). CH is responsible for creating a general decision about the channel state. Several CHs may subsequently exchange the sensing information to derive a general decision about the channel state. An advantage of this mode is that the overall overhead on determining the final sensing decision will be distributed among the various clusters, but the modeling of these systems is too complicated.

2.4 Machine learning

Machine learning (ML) is a branch of artificial intelligence that provides systems automatically learn and improve their performance. ML algorithms are designed, in such a way, that they can improve their behavior through experiments without being explicitly programmed to do so. ML processes involve; collecting data that describes a particular problem, finding hidden patterns beyond that data, and eventually making a decision.

2.4.1 Main categories of ML algorithms

ML algorithms are generally classified into three main categories,

- Unsuppervised ML: Clustering algorithms aim to organize the samples of data into several clusters according to the features that distinguish them from the other [21]. In this type, the ML classifier is fed blind samples without declaring their distinct labels.
- Supervised ML: Unlike the previous type, the samples are fed to the ML classifier along with their labels to produce a general decision [22].
- Reinforcement ML: In the standard reinforcement ML approaches, the agent observes and acts with the environment and predicates the suitable action that should be taken according to the actual state [23]. Therefore, it receives a reward or a punishment that suits the action that was taken as shown in figure 2.5. After considerable training, the agent has the experience to handle certain states.

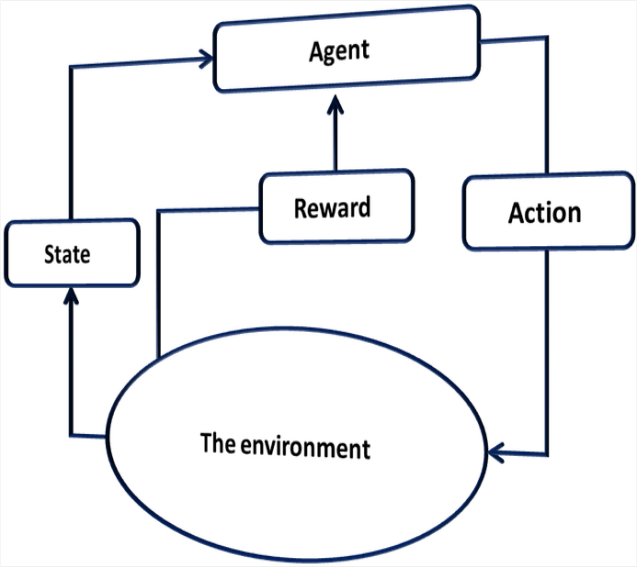


Figure 2.5: Reinforcement ML approach

Several ML algorithms are covered under each type. kMeans, Gaussian mixture model (GMM), and Bayesian classifier are some of the most common examples of unsupervised ML. Different types of neural networks (such as artificial neural network (ANN), convolutional neural network (CNN)), and support vector machines (SVM) are the most famous examples of supervised ML. TD learning, Q-Learning as well as Deep Q-Learning are addressed within a reinforcement learning framework.

2.4.2 ML-based CSS: General model

Three types of conventional CSS schemes are energy detection (ED), cyclo-stationary detection, and matched filter (MF) detection. SU checks for the presence of a PU signal and then sends back a local decision about the channel availability status toward the decision-making entity [5, 20, 24]. In contrast, the decision-making in CSS schemes which is based on ML schemes (ML-based CSS) includes two types of classifier algorithms are: **I- feature extraction and II- machine learning classifier**. In these schemes, the system aggregates the sensing information by all SUs. After that, the features that characterize the PUs' status are extracted in the first classifier using the mathematical algorithms. Subsequently, these features are fed into the ML classifier to generate the possible clusters. Later, the final sensing decision is made based on the result that is generated in the ML classifier.

Chapter ▷3

Literature Review

This chapter discusses how can be reformulated various machine learning algorithms in terms of the CSS problem. Then, the general model of the ML-based CSS is presented. Further, we review several techniques for feature extraction schemes proposed in the literature. Finally, this chapter examines several types of ML-based CSS.

3.1 ML-based CSS

Classification of ML-based CSS schemes, based on the type of ML algorithm, include: I- unsupervised-based CSS, II- supervised-based CSS, and III- reinforcement-based CSS [25]. In the unsupervised-based CSS, the features are fed into the classifier without declaring their distinct labels [7, 26–34]. In the supervised-based, the features are filled along with their labels within the classifier to construct the final decision [35–43]. Reinforcement-based addresses specific problems such as power consumption level, throughput, energy efficiency, etc. The agent is given rewards when evaluating its behavior. These rewards are evaluated based on the problems intended to be solved [44, 45].

3.1.1 CSS system with Multi-classes

In a large-scale CRN with multiple PUs, multiple hypotheses must be formulated within the framework of binary hypothesis testing. Assuming CRN with P$'$ active PUs of P PUs and possible classes of active P$'$ (i.e. $\{C_1, C_2, ..., C_i\}$). Each ith class, has possible combinations $\binom{P}{i} = \frac{P!}{(P-i)!i!}$. Herein the null hypothesis, H_0, indicate that all PUs are inactive [38, 43] while there are ith alternative hypotheses, H_i, construct the general alternative hypothesis, H_1, as shown in equation 3.1,

$$H_1 = \bigcup_{i=1} H_i \tag{3.1}$$

In another case, multi-class hypotheses can be formulated with one PU, whereby several classes represent an assurance level of PU presence. For example, [30] initiates three clusters C_1, C_2, and C_3 indicating the power is high, weak, and the absence of the received PU signal, respectively. However, in our work, we consider small-scale CRN with one PU.

3.2 Feature extraction methods

Several mathematical algorithms have been devised to extract the features of the data samples. These algorithms fall mainly into three categories. The first category is energy detection-based (ED-based). In this type, features are the sensed PU signal power. The second is schemes based on signal processing. This type focuses on finding the circular characteristic that helps to distinguish between transmitted signals and noise. The third is the covariance matrix-based schemes, which initially take the sensed PU signal power. Then, they apply the mathematical calculations to derive some scales used as feature vectors.

3.2.1 ED-based schemes

Let the channel's availability state is *'A'*, then, *'A'* is defined as in equation 3.2

$$A = \begin{cases} 1, & for \quad H_0 \\ 0, & for \quad H_1 \end{cases} \tag{3.2}$$

When the n^{th} SU receives *K* samples from the transmitted PU signal, it calculates the energy of that signal by equation 3.3,

$$Z_n(k) = [A * \alpha_n * s(k)] + w(k) \tag{3.3}$$

where *Z(k)* is the energy value corresponding *k* samples and $w(k)$ is the noise-usually considered to be the additive white Gaussian noise (AWGN) with the zero mean and 1 variance. The total energy (*E*) is estimated by the n^{th} SU as shown in equation 3.4

$$E_n = \frac{1}{K} \sum_{k=1}^{K} |Z_n(k)|^2 \tag{3.4}$$

SUs monitor the communication channel in the sensing period and then estimate the value of E. The column vector from energy values $\{E_1, E_2, .., E_n, ..., E_N\}^T$ is then used to determine the channel state.

Probability vectors

This method learns about the properties of the distribution models that underlay the hypotheses H_0 and H_1. So, it aims to reduce the multidimensional energy vectors to two-dimensional probability vectors [27, 35]. After collecting a sufficiently large number of sensing samples over an appropriate sensing period, the new probability vectors for the two hypotheses $P^G_{H_i}$ are derived from the probability density function as given in equation 3.5

$$P^G_{H_i} = \phi(E|\mu_{E|H_i}, R_{|H_i}), i \in \{0, 1\} \tag{3.5}$$

where $\mu_{|H_i}$ and $R_{|H_i}$ are respectively the mean vector and the covariance matrix of the multivariate Gaussian distribution ϕ.

3.2.2 Covariance matrix-based schemes

The sensing matrix (*Si*) can be generated from the received PU signal as given in equation 3.6,

$$Si = \begin{bmatrix} y_1(1) & y_1(2) & y_1(3) & \dots & y_1(K) \\ y_2(1) & y_2(2) & y_2(3) & \dots & y_2(K) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_N(1) & y_N(2) & y_N(3) & \dots & y_N(K) \end{bmatrix} \tag{3.6}$$

where *N* is the number of SUs. *K* represents the number of samples. $y_N(.)$ is the received signal estimated by N^{th} SU under the binary hypotheses. Therefore, the covarience matrix (*R*) can be reconstructed from *Si* matrix using equation 3.7

$$R = \frac{1}{K} \sum_{k=1}^{K} Si * Si^T \tag{3.7}$$

Si^T; The transpose of a matrix- Si

Many of the proposals in the literature follow this method. We mention from them

Eigenvalue, eignvector

Let the $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$ be the eigenvalues of the covariance matrix R. $(\vec{\nu_n})$ and I be the eigenvector and the identity matrix of R respectively. Then, the eigenvalue can be calculated using equation 3.8 as,

$$|R - \lambda I|\vec{\nu_n} = 0 \tag{3.8}$$

Many researchers adopt the eigenvalue/eigenvector in their work. For example, the ratio of the maximum and minimum eigenvalue (MME) and the difference between the maximum and the mean eigenvalue (MSE) are used as feature vectors in [27, 31]. The improved versions of MME and MSE (IMME and IMSE respectively) are used as features in [28] after calculating the principal component (PC) of the eigenvector of the matrix *Si*.

Geodesic distance

In the geodesic distance, the matrix *Si* is sequentially split upon the order and the intervals into sub-signal vectors *s* of length *q*, i.e., *s* = *K* / *q*. Then, two new covariance metrics Si^O and Si^I are derived from *s* and *q* respectively [29]. The Riemann means ($\overline{Si}^O$) and ($\overline{Si}^I$) which represent the noise environment are calculated for both matrices Si^O and Si^I. After that, the geodesic distance (*Gd*) between Si^O, Si^I and their $\overline{Si}^O$, $\overline{Si}^I$ can be used as feature vectors, given in equation 3.9,

$$Gd(Si^a, \overline{Si}^a) = \sqrt{\frac{1}{2}\sum_{j=1}^{Nq} \log^2 \lambda_j} \quad , a \in \{O, I\} \tag{3.9}$$

Where λ_j is the i^{th} eigenvalue indirectly derived from the Si^O and Si^I matrices.

3.2.3 Signal processing-based schemes

The idea of methods based on signal processing came from the fact that there are characteristics that distinguish transmitted signals and noise. Such characteristics can easily be detected in the frequency domain [39, 46]. Waveform-based and cyclostationarity-based types, which are the most types addressed under this method, will be briefly explained here;

Waveform-based schemes

In many situations, the PU signal pattern such as the regular pilot, preambles, spreading sequence, etc. can be known and used efficiently. Therefore, the binary hypothesis test can be formulated by the waveform-based method as given in equation 3.10

$$y(k) = \begin{cases} Re\Big[\sum_{k=1}^{K} w(k)s^*(k)\Big], & H_0 \\ \sum_{k=1}^{K} |s(n)|^2 + Re\Big[\sum_{k=1}^{K} w(k)s^*(k)\Big], & H_1 \end{cases} \tag{3.10}$$

where *Re* indicates the real part of the complex received signal and * is the complex conjugate operator.

Cyclostationarity-based schemes

cyclostationarity refers to the periodic characteristics of a certain signal. The received PU signal usually has a cyclostationary characteristic such as the mean and the autocorrelation. In contrast, the noise signal does not have such periodic characteristics. Therefore, we can utilize the differences between the transmitted and the noise signals. The determination of the presence of the cyclic pattern of the PU signal can be examined using the so-called spectral density function (SCD). Here, the binary hypothesis can be reformulated by this method as given in equation 3.11,

$$S^w_{y(k)}[k] = \begin{cases} S^w_{W(k)}[k], & H_0 \\ |\alpha|^2 S^w_{s(k)}[k] + S^w_{W(k)}[k], & H_1 \end{cases} \tag{3.11}$$

where $S^w_{y(k)}[k]$ is the SCD of the PU transmitted signal at some cyclic frequency w and α is the channel gain coefficient.

3.3 Data combining and ML

After extracting the features in the CSS mode, the classifier can apply soft or hard combining schemes for decision making. The determined type depends on the nature of the feature vectors. SUs in hard combining schemes implement a mechanism to digitize their local observation, while they explicitly exchange their local decisions in soft combining schemes (i.e., the SUs exchange the real values of their local decisions).

3.3.1 Logical And-based/Or-based CSS

And/Or-based CSS techniques are the most examples of hard combining schemes. In the And-based CSS, the channel is considered occupied, if and only if all SUs determine that the PU is active. Unlike the And-based CSS, in the Or-based CSS, the channel is considered busy if at least one SU determines that the PU is active [47]. Therefore, if all SUs are independent then the two probabilities, the probability of false alarm P(FA) and the probability of detection P(D) of the And-based technique can be calculated as shown in

equation 3.12

$$P(I) = \prod_{n=1}^{N} P(i)_n, \qquad I, i \in \{FA, D\} \tag{3.12}$$

P(FA) and the P(D) of the Or-based technique can be calculated as shown in equation 3.13

$$P(I) = 1 - \prod_{n=1}^{N} 1 - P(i)_n, \qquad I, i \in \{FA, D\} \tag{3.13}$$

N; is the number of SUs

The appendixB.2.3 in the appendices shows the source code of the And-based and the Or-based CSS techniques.

3.3.2 Unsupervised-based CCS

After collecting a sufficiently large number of training feature vectors, the ML classifier constructs the suit decision. In unsupervised-based methods, these blind features are populated within the classifier to produce the global decision about the channel state. Various types of unsupervised algorithms have been proposed in the literature to solve CSS problems in CRNs [7, 33, 34].

KMeans-based model

KMeans algorithm maps the feature vectors to non-overlapping clusters, at the nearest Cartesian distance. Each possible cluster represents a set Ψ and is indexed by *j*. So, KMeans-based methods try to find the *J*s of clusters corresponding to various channel states.[1] Each cluster has its centroid C_j that represents the cluster arithmetic mean. Therefore, the objective function of the KMeans-based CSS technique (*distortion function,* Θ) is to find the minimum squared distance of overall clusters from their corresponding centroid as shown in equation 3.14,

$$\Theta_{(\{\Psi_j\},\{C_j\})} = argmin \sum_{j=1}^{J} \sum_{l \in \Psi_j} \eta_{lj} ||l - C_j||^2 \tag{3.14}$$

where l is a feature vector and $\{.\}$ reflects the cardinality of Ψ_j and C_j sets. $||.||$ is the ℓ^2-norm, η_{lj} takes 1 if l is belong to Ψ_j and 0 otherwise.

After training the clusters, the items and the centroid of each cluster become known. In the testing phase, the classifier becomes able to make a

[1] J here is the same K in the KMeans algorithm that indicates the number of clusters. The experiments of this thesis are based on K=2 corresponding two states (channel un/available states)

suitable decision about the channel state, i.e., the channel is available or not. Let *l'* denotes the test vector, then the decision making can be defined as shown in equation 3.15,

$$\frac{||l' - C_i||}{min_{j=1,...,J}||l' - C_j||} \geq \zeta \tag{3.15}$$

$C_i < min(C_j)$ **which usually represents the noise cluster**

The test vector *l'* is classified to the cluster C_i if equation 3.15 satisfied. Otherwise, it is classified to the cluster C_j. Appendix B.2.2 and appendix B.1.3 in the appendices respectively show the source codes of the KMeans algorithm and the KMeans-based CSS technique.

Gaussian mixture model

Gaussian mixture model (GMM) provides a smooth classification in the opposite of the KMeans algorithm. The clusters in GMM-based CSS are an overlay of Gaussian densities. The statistical parameters are first randomly initialized for each cluster (i.e., for various Gaussian densities). Let $\mathcal{N}(x|\mu_k, cov_k)$ is a Gaussian cluster k with the parameters μ_k and cov_k (x represents the set of items that belong to cluster k). Then, these parameters can be estimated by maximizing the likelihood function [27] given in equation 3.16,

$$\ln p(l|\pi, \mu, cov) = \sum_{m=1}^{M} \ln \Big\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(l|\mu_k, cov_k) \Big\} \tag{3.16}$$

where l is the training vector, M is the number of training vectors, and π_j is the mixing coefficient with $\sum_{k=1}^{K} \pi_k$ equals 1.

After training the model for several iterations, the parameters μ_k and cov_k converge. In the testing phase, the testing vectors (l') are then classified to the corresponding cluster the classifier rule given equation 3.17 (assuming the binary hypotheses test is adopted)

$$\ln \frac{\pi_1 \mathcal{N}\Big(l'|\mu_1, cov_1\Big)}{\pi_2 \mathcal{N}\Big(l'|\mu_2, cov_2\Big)} \geq \zeta \tag{3.17}$$

The unsupervised-based CSS techniques are widely adopted in a lot of researches. KMeans with ED-based scheme was adopted in [26] whereas the geodesic distance was proposed for extracting the feature vectors with the fuzzy C-mean in [29]. [30] proposes filtering energy vector to get the max (Emax) as features of the kernel fuzzy C-means. The eigenvalue/eigenvector

as features with the KMeans were adopted in [28] while with the GMM is was adopted in [27]. The improvement IMSE, IMME were used in [27, 28, 33]. Non-parametric Byasien learning models, the Hierarchical Dirichlet Process, and the beta process sticky hidden Markov model were proposed in [7, 32] respectively. The hierarchical Dirichlet Process converts the received signal to the frequency domain to find the Fourier coefficients. Then, it uses these coefficients as feature vectors. Beta process sticky hidden Markov model is ED-based.

Additionally, [31] introduces a signal processing scheme (WEMD) which is combined the empirical mode decomposition algorithm and the wavelet threshold algorithm to remove the noise components and thus reduce their effects. WEMD adopts KMeans as an ML classifier with two types of feature vectors which are: I- the difference between maximum eigenvalue and the average energy, and II- the difference between the maximum and minimum eigenvalue. Last but not least, a blind continuous hidden Markov model scheme was proposed in [34]. This algorithm can recognize the PU transmit power level. It uses the KMeans algorithm to estimate the channel state as well as the continuous wavelet transform method for features extraction. This study proposes two strategies. The first strategy applies the ED method to build the observation sequence before computing the continuous wavelet transform, while the latter strategy uses the MME as feature vectors.

3.3.3 Supervised-based CSS

In the supervised-based CSS, the features are extracted from the received signal. Then, they are labeled in a certain way. Labels are the correct individual sensing decisions that each SU gets. The binary labels, due to the binary hypothesis, can be represented as follows; label 1 indicates that the PU is absent and the channel is available. Label [0] indicates that the PU is present and the channel is unavailable. The signaling process (mapping the feature with the appropriate label) increases the sensing accuracy, but it also increases the system complexity and training overhead [37, 38]. Support vector machine (SVM) and neural network (NN) are examples of supervised-based CSS techniques.

Support vector machine (SVM)

SVM is an algorithm that aims to find the optimal hyperplanes that leave the maximum margin from all possible classes. SVM for linearly separable data uses the linear kernel and its formula is defined in equation 3.18

$$Kr(l) = w^T l + bis \tag{3.18}$$

where w is the hyperplane rotation weight and bis is the hyperplane bias. Given a training vector l which may be labeled either 0 (l_0) or 1 (l_1), the optimization problem of SVM is then defined as shown in equation equation 3.19,

$$\begin{aligned} &minimize \quad \frac{1}{2}||w||^2 + c * \sum_{l=1}^{L} \zeta^l \\ &s.t. \quad l_i(w^T * l + bis) \geq 1 - \zeta^l, \quad i \in \{0,1\}, \quad \zeta^l > 0 \end{aligned} \tag{3.19}$$

where ζ is a tolerance limit that represents the number of vectors that misclassified into an incorrect cluster. The constant c controls the magnitude of ζ. The value of c depends on the trade-off between achieving the lowest misclassification or the maximum margin. Minimizing w achieves the equation, but on other hand, minimizing w is a nonlinear task. However, this convex optimization can be solved under the Karush-Kuhn-Tucker (KKT) condition. After finding the optimal boundary, a new test vector *l'* can be classified using equation 3.20,

$$Classf(l') = sign(\sum \lambda^\iota * Kr(l, l') + bis) \tag{3.20}$$

λ^ι; The Lagrange multiplier

Lots of researches based on SVM were proposed. A probability vector with linear and polynomial kernel was proposed in [35]. [36] proposed applying two phases of SVM algorithms with ED-based methods. In the first phase, each energy vector is mapped to a predicated label, whereby the unsuccessful ones are propagated during the second phase to alleviate the misclassification errors. Multi-class SVM (M-CSVM) with ED-based and multiple PUs was adopted in [38, 42], whereas the supervised beamformer-based technique was proposed in [43]. To implement M-CSVM, an approach of classes coding is needed. For example, the one versus rest approach characterizes a particular class as a positive class and the rest as negative classes.

Neural network (NN)

Neural network algorithms are a set of interconnected virtual neurons that operate in a manner somewhat similar to biological neurons or electronic structures. Few NN-based CSS techniques have been proposed. Deep CSS based on Convolutional Neural Network (CNN) was proposed in [37]. The CNN structure consists of convolution and fully connected parts (FC). The convolution part includes three layers which are: (I) the 3*3 convolution layer (3*3 Conv.), (II) rectifier linear unit (ReLU) layer, and (III) max-pooling layer.

[37] studies modeling a small-scale CRN with a PU that operates several sub-bands. Each SU examines the presence of a PU on sub-bands then produces a 2D set of the labeled features. After giving the 2D data set during the sensing process, it is fed to the first layer to extract the spatial correlation of the sensing data. The residual layers deal with the nonlinearity and reduce the size of the sensing data. Finally, FC multiplies the weights and adds biases to generate the final decision.

Ensemble classifier was introduced in [40]. This unsupervised scheme adopted the cyclostationary-based features extraction method and uses the decision tree and AdaBoost for decision making. Finally, [41] suggested unsupervised CSS based on an artificial neural network (ANN). This study makes a 2D set of individual SUs decisions and uses Zhang statistics to train ANN to make a decision.

3.3.4 Reinforcement CCS

Reinforcement learning is concerned with taking actions that maximize the reward. Reinforcement-based CSS techniques are rare. Sensing policy based on ϵ-*greedy* policy was proposed in [44]. The reward in this study represents the immediate throughput corresponding to the selected sub-band *sd*. Under this policy, the authorized SU that has access to *sd* will inform the FC of the information about the achieved throughput. Later, the FC compares the Q-value of a particular SU with the prior decision and updates the Q-value based on the following rule given in equation 3.21

$$r_{t+1}(SU, sd) = \begin{cases} dn_{t+1}(SU, sd), & dn_{t+1}(FC, sd) = 1 \\ Q_t(SU, sd), & dn_{t+1}(FC, sd) = 0 \end{cases} \tag{3.21}$$

where dn_{t+1}*(SU , sd)* indicates the decision that made by the SU on *sd*. While dn_{t+1}*(FC , sd)* is the global decision made by the FC accordingly. When FC truly estimates the sd state, it is granted a reward equal to the immediate throughput. However, it is granted the last Q-value when it misestimates the sd state. After updating all Q-values based on this rule, the FC exploits its knowledge and informs SUs to sense the sd that has the maximum Q-values.

Finally, an efficient sub-band selection policy based on replicated Q-learning was proposed in [45]. This technique introduces a partially observable Markov decision process that awards high rewards for a large number of idle channels while otherwise being awarded smaller rewards.

Chapter ▷4

System Modeling & The Experimental Setup

This chapter will examine system modeling and network simulation. It also explains the challenges encountered while preparing for the experimental part and the ideas used to overcome these challenges.

4.1 System modeling

We consider a small-scale CRN that consists of a PU network placed in the center and N multiple SUs around it. Two mobility scenarios are considered the stationary and the mobile CRNs, abbreviated as StaCR and MobCR, respectively. The PU network in the two scenarios is the same and consists of a fixed PU transmitter (PU-Tx) and a fixed PU receiver (PU-Rx) placed at 15 meters far from the PU-Tx. SUs in the first scenario are also fixed and positioned 120 meters away from the PU network.[1] In the second scenario, SUs are mobile and move randomly in the area (i.e., moving in a random direction at a random velocity. The reasonable setting of these values has been taken into account). However, both scenarios start with the initial configuration as shown in figure 4.1

Different levels of cooperation are considered too. The Non-CSS is firstly considered, in which only one node participates in the sensing process. Then, different cooperative SUs (i.e., 2SUs, 5SUs, and 10SUs) are participating in the sensing process. In addition, the effect of various fading channels is examined. We adopt the general $\kappa - \mu$ fading model. What makes this model preferable is that completely different fading channels can be controlled and modeled by two parameters of the distribution, the κ and the μ. A comprehensive guide of the general $\kappa - \mu$ fading model implementation and the test

[1] These locations are sensitive and carefully considered where if they change low according to mobility, the SUs sometimes become able to detect the presence of PU and sometimes not.

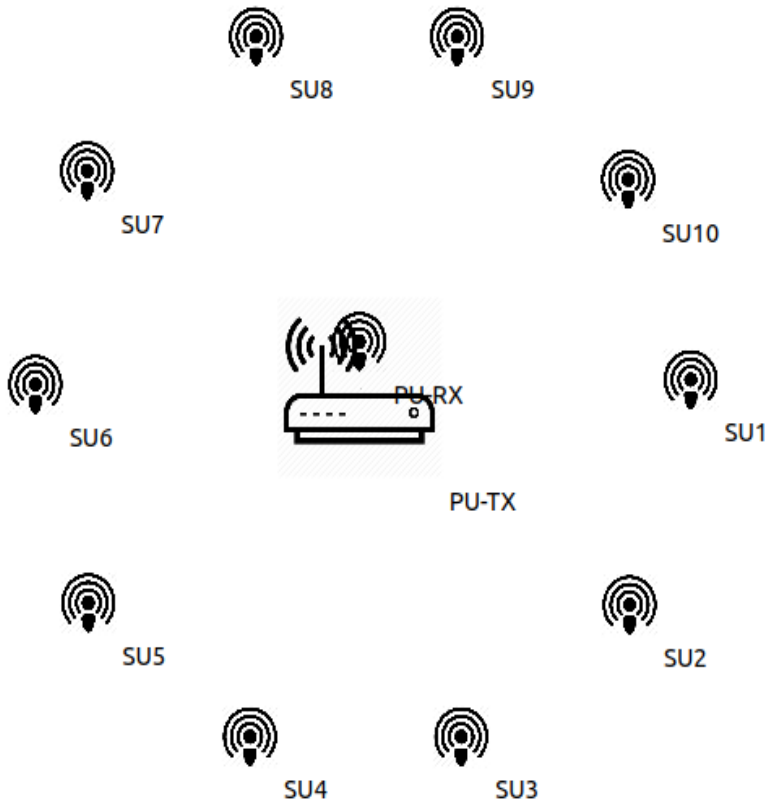


Figure 4.1: System Modeling

is available in the appendix A.1.

4.2 The simulation & noise modeling

We used the well-known third version of the discrete-event network simulator (ns3.30) to model a small-scale CRN and; to generate datasets.[2] In this model, we have assumed that the PU network always operates at channel 36 of the IEEE802.11n-5GHz wireless technology. The Wifi mode for unicast data frames is indexed to 'HtMcs6' value which is a metric to several parameters of the wifi connection such as the 64-QAM modulation type, the 3/4 coding rate, one spatial stream, and etc.[3]

During the simulation time, the PU-Tx randomly broadcasts 1500 byte-length UDP packets to the PU-RX with a data rate of 5Mbps. SUs must monitor and estimate the instantaneous signal-to-noise ratio (SNR) of each packet for overall decision making. In the simulation experiments, SUs listen to channel 36 for 5 milliseconds per second, then estimate the normalized

[2]This study was developed using version 3.30 documented on the web page `https://www.nsnam.org/releases/ns-3-30/documentation/`

[3]Full MCS table is available in `http://mcsindex.com/`

SNR over the entire simulation time. After that, SUs determine their local decision about the channel state, busy or ideal, according to the estimated SNR value.

4.2.1 The general $\kappa - \mu$ fading distribution

A group of studies searched for the optimal distribution that suits the level of the signal power propagated over communication channels. The general $\kappa - \mu$ fading model was first proposed and examined in [16]. This study presents a general $\kappa - \mu$ distribution that can be controlled via the two parameters, κ, and μ. By controlling these parameters, many types of fading channels can be modeled. Table 4.1 below shows the parameters' values for different types of fading channels.

Fading Channels		
Fading Channel types	Parameter κ	Parameter μ
Nakagami-m	$\to 0$	μ (or m)
Rayleigh	$\to 0$	1
Rician	κ	1
One-sided Gaussian	$\to 0$	0.5

Table 4.1: $\kappa - \mu$ values for different fading channel types

In our work, we aim at investigating the performance of several CSS techniques in the general $\kappa - \mu$ fading channel. Unfortunately, the ns3 package has no such model of this type of fading. Therefore, we developed our $\kappa - \mu$ fading model for the ns3 simulator. Here, we employed the well-known rejection sampling method to directly sample random variables from the $\kappa - \mu$ distribution [48]. The subsection 2.2.4 provides more information about the κ-μ fading, while appendix A.2 explains the rejection sampling method. Appendices A.1 and B.1.1 introduces the the implementation of this model within the ns3 simulation platform. **In our experiments, we assume that the probability of PU-Tx activity is 0.5. While the experiments for various types of fading channels were carried out for the $\kappa \to 0, \mu = 3.5$ (Nakagami) and the $\kappa = 2.65, \mu = 1$ (Rician).** In the page below, brief information on the configuration of a Rayleigh fading channel experiment is provided.

```
Stationary CR with Rayleigh fading
(brief info.)
==================================

Time:2400s;         TxPower:3dBm

Signal(dBm)     Noi+Inf(dBm)      SNR(dB)
=============================================
-84.7             -93.9             9.28

Kappa = 1.18e-38; Mu = 1; (Rayleigh)

Distance from, to(m)...
PU -->> SU         Distance
        0-         15 (PU-Rx)
        1-         120
        2-         120
        3-         120
        4-         120
        5-         120
        6-         120
        7-         120
        8-         120
        9-         120
        10-        120

Channel is avaliable for // 1.22e+03s // time long
Propability of the PU is inactive: 51%
Channel is not avaliable for // 1.18e+03s // time long
Propability of the PU is active: 49%
```

```
Mobile CR with Rayleigh fading
(brief info.)
==============================

Time:2400s;         TxPower:3dBm

Signal(dBm)     Noi+Inf(dBm)      SNR(dB)
=============================================
-83.9             -93.9               10

Kappa = 1.18e-38; Mu = 1; (Rayleigh)

Distance from, to(m)...
PU -->> SU         Distance
        0-         15 (PU-Rx)
        1-         149
        2-         377
        3-         184
        4-         54.7
        5-         195
        6-         111
        7-         87.5
        8-         131
        9-         106
        10-        122

Channel is avaliable for // 1.21e+03s // time long
Propability of the PU is inactive: 50.2%
Channel is not avaliable for // 1.19e+03s // time long
Propability of the PU is active: 49.8%
```

Configure a Rayleigh fading channel simulation.

4.2.2 The noise Model

While ns3 is in nature a discrete-event simulator, we can only track and extract the data when the PU becomes active (i.e., during the ON-intervals).[4] To

[4]That because in the off-intervals the PU become silent and there are no more events to track. See `https://www.nsnam.org/docs/release/3.30/manual/html/events.html` for more explanation

overcome this problem, we developed a noise model using the python language and embedded it in the ns3. The noise model follows the Gaussian distribution and can be controlled using the mean of the distribution (μ) as well as the standard deviation (σ). A higher value for μ and σ indicates a very-noisy environment, while a lower value indicates a low-noise environment.

We started the experiments by assuming the parametric noise model has -89.75 dBm and μ and the σ parameters = 1.0 dBm. The low-noise environment, abbreviated as Env1/1, the coded as given in appendix B.2.1. Then, we raised these values to model different channel conditions. Table 4.2 below depicts the various noise conditions that were considered and their abbreviations.

**	low-noise environment		High-noise environment	
μ	1.0 dBm		1.5 dBm	
σ	-89.75 dBm	-89.25 dBm	-88.25 dBm	-87.5 dBm
Abb.	Env1/1	Env1/2	Env2/1	Env2/2

Table 4.2: The different noise conditions

The resulting data was then combined with that of ns3 to obtain the complete dataset that is in the experiments.[5] While the Modulation and Coding Index (MCS Index) was set to 'HtMcs6'. The packet size and the application data rate were also set to 1500-byte and 5Mb/s, respectively. We found the best configuration of the noise model to generate 2-samples per time key (time key indicates the second's index, i.e., 2-samples/s). The best number of samples taken during the sensing duration can be calculated by dividing the sensing interval by the packet size and the application data rate. This already suits the data that was produced from the ns3 and reflects the probability of the PU-Tx being active that was configured. Therefore the accurate selection of the MCS Index, the size of the packet, and the application data rate has a direct effect on the performance measurement.

[5]There is a note to be mentioned when FC loses a sample of SU at a certain instant, it randomly compensate this sample with a low value to get feature vectors with the same length.

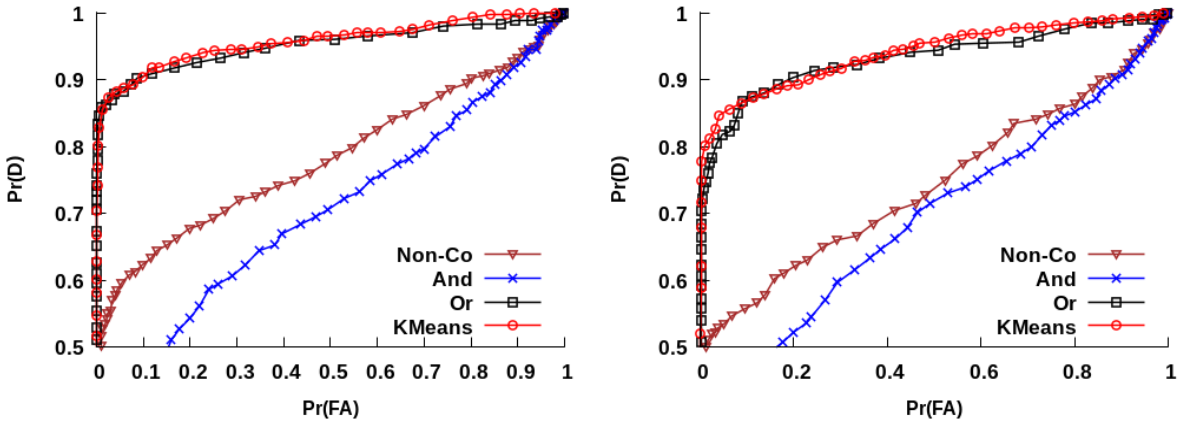
Chapter ▷5
Results and & discussion

This chapter discusses the most important findings of this work. The performance for Non-CSS/CSS techniques is measured by the receiver operating characteristic (ROC) curve. This curve plots the P(D) versus the P(FA) for different SNR settings. High P(D) versus low P(FA) indicates good detection, and the opposite indicates bad detection. See appendix A.3 for more information on the ROC curve. The experiments were conducted for the Non-CSS, the and/Or-based CSS, and the KMeans-based CSS techniques.

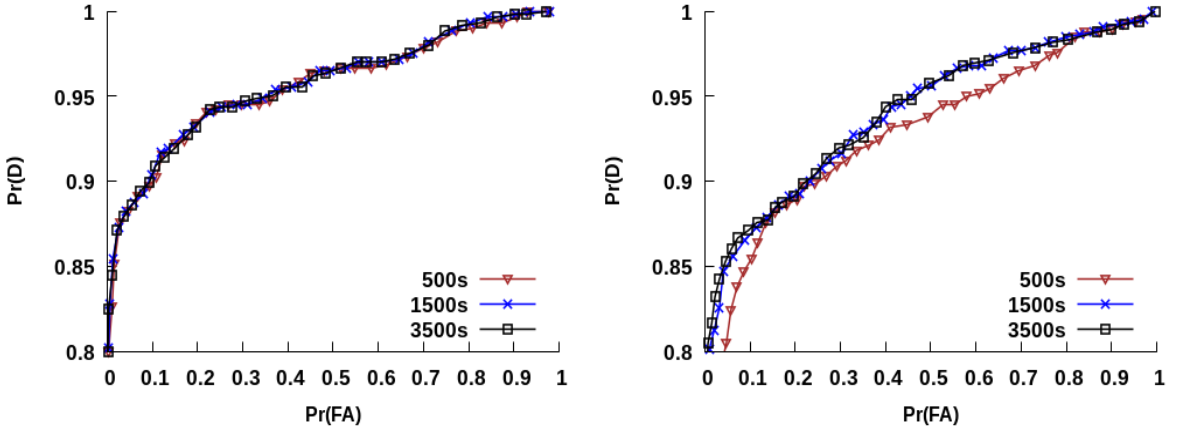
5.1 Stationary CRN vs. Mobile CRN

We started the experiments by assuming that the parametric noise model has -89.75 dBm and 1.0 dBm values for the μ and the σ parameters, i.e., for Env1/1. Then, we raised these values to model different channel conditions. Table 4.2 above depicts the various noise conditions that were considered. In the addition, the experiments were initially carried out for Rayleigh fading (i.e., $\kappa \to 0$ & $\mu = 1$).

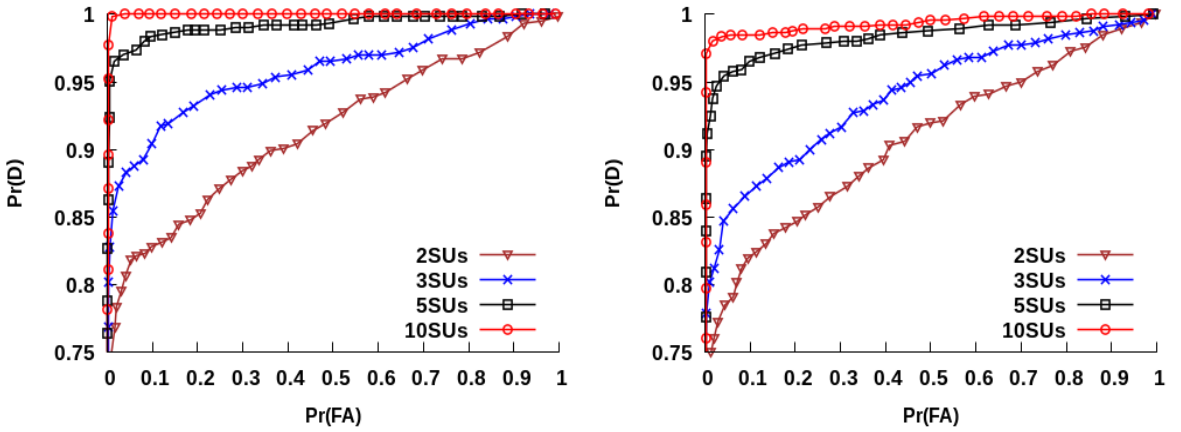
Figure 5.1 and figure 5.2 show that the performance is generally better for the stationary CR (StaCR) as compared to the mobile CR (MobCR). Figure 5.1a-left shows that the And-based technique in the StaCR and under Rayleigh fading seems to give the worst-case and proximate to the Non-CSS performance. This is because when a SU wrongly detects the presence of the PU, it affects the global decision made by all SUs. Figure 5.1a-left also shows that the Or-based and the KMeans-based techniques have perfect and comparable detection performance. In MobCR under Rayleigh fading, the KMeans-based technique is very slightly superior to the Or-based technique, as clearly appears in figure 5.1a-right. However, the latter gives good performance while the Non-CSS and the And-based techniques give a degraded



(a) ROC of different CSS techniques, [Rayleigh, N=3SUs, M=1500s]



(b) KMeans-based CSS with different training samples, [Rayleigh, N=3SUs]



(c) KMeans-based CSS with different number of collaborative SUs, [Rayleigh, M=1500s]

Figure 5.1: StaCR (left) vs. MobCR (right) of Rayleigh fading

performance.

Figure 5.1b and figure 5.1c show that the number of samples, M, has no clear effect while the number of SUs, N, have an obvious effect on the performance of the KMeans-based technique, regardless of the mobility nature of SUs. Figure 5.1b and figure 5.1c generally show that we need about 1500 samples and at least 3 SUs to reach acceptable performance.

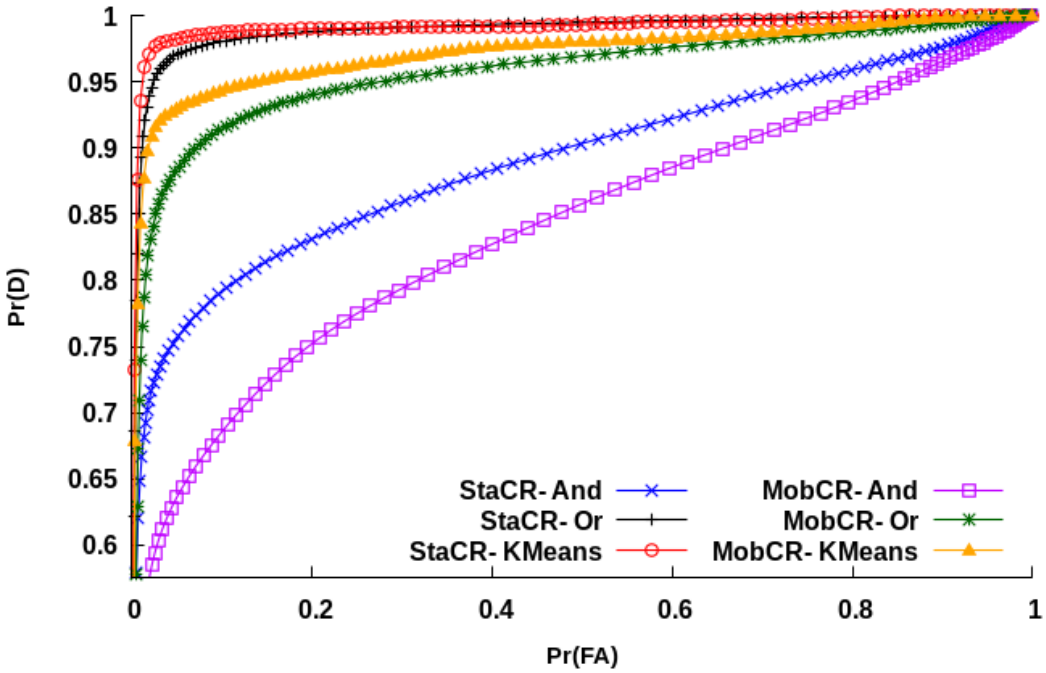
Figure 5.1 confirms that employing more than 5SUs for cooperative sensing as well as using more than the 1500s don't massively improve the efficiency or accuracy of the system. The results also show that all techniques provide slightly better performance in the StaCR as compared to MobCR. This is due to the more dynamic nature of mobile channels, which introduces difficulties in identifying the presence of PU. This leads to a higher probability of F(A).

5.2 $\kappa - \mu$ setup

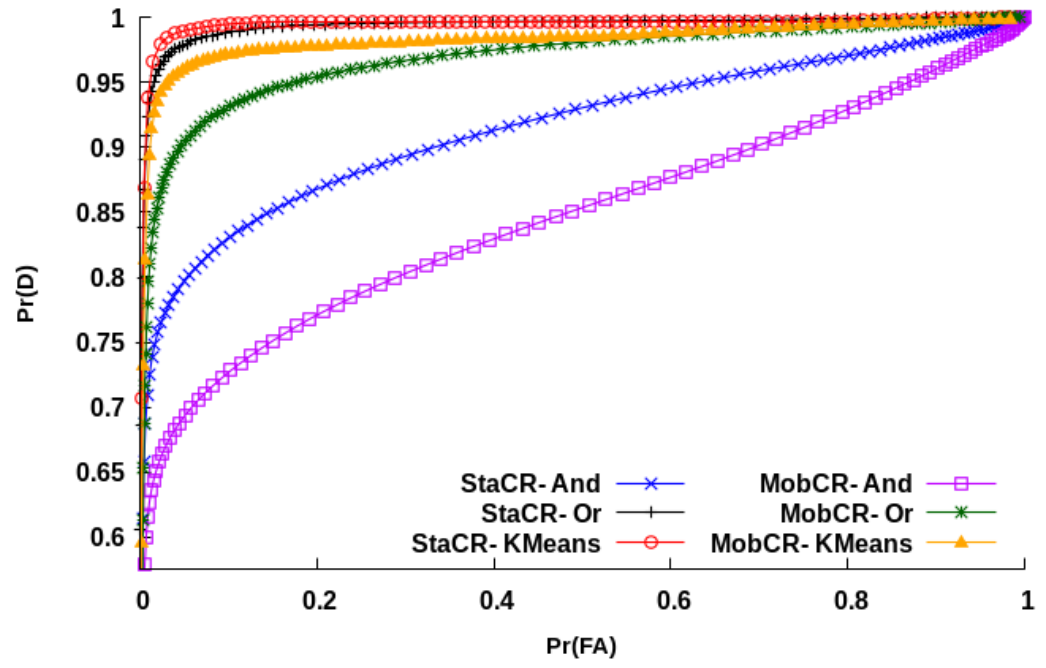
Returning to table 4.1, we easily conclude that the Rayleigh fading combines the Rician fading set and the Nakagami fading set. Rayleigh fading can be simulated as κ approaches 0 and μ equals 1. Different types of Rician fading can be modeled by fixing the μ parameter and tuning the κ. Also, different types of Nakagami fading can be modeled by fixing the κ parameter and tuning the μ parameter. The experiments for other fading channel types were carried out for $\kappa \to 0, \mu = 3.5$ (Nakagami) and $\kappa = 2.65, \mu = 1$ (Rician).

In Nakagami fading channels, figure 5.2a, and Rician fading, figure 5.2b, it is clear that the KMeans-based technique and the Or-based technique almost provide comparable performance. That is superior to the And-based technique in the StaCR. On the other hand, the KMeans-based technique outperforms other techniques in MobCR. In general, the performance is better under the Nakagami and Rician fading as compared to Rayleigh fading for all CSS techniques, the And-based technique, the Or-based technique, and the KMeans-based techniques. Thus, we conclude that the characteristic of the fading environment is highly affecting the performance of all CSS techniques as shown in the figures, figure 5.2 and figure 5.3.

Figure 5.3 highlights the performance of the KMeans-based technique of Rayleigh fading compared to other fading distributions. The results show that the performance of the KMeans is best in the Nakagami, figure 5.3a, and

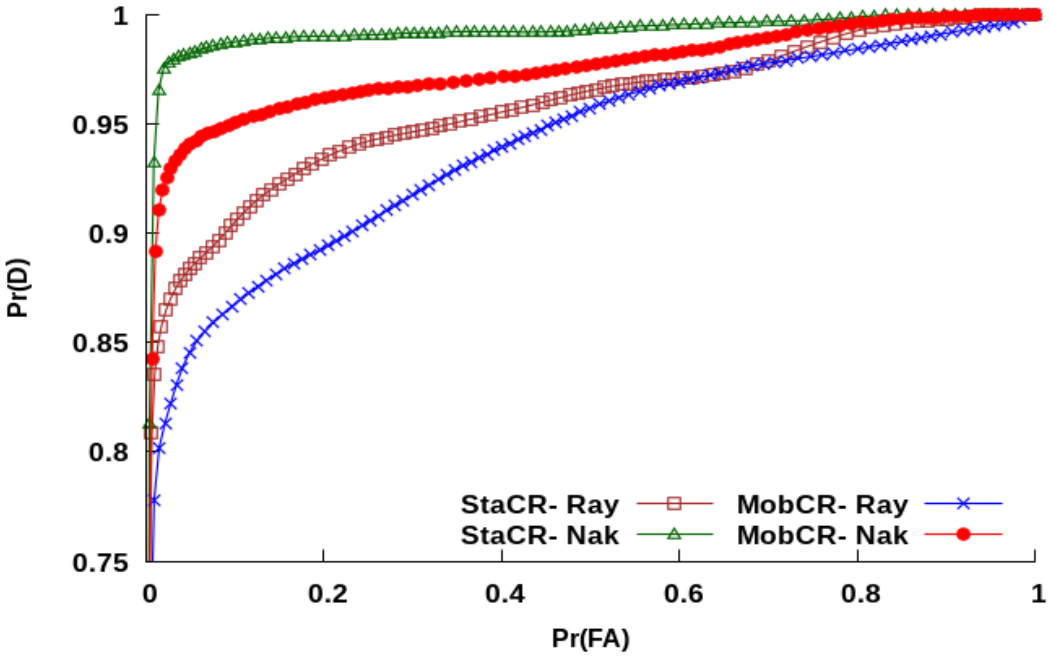


(a) Nakagami, [N=3SUs, M=3500s]

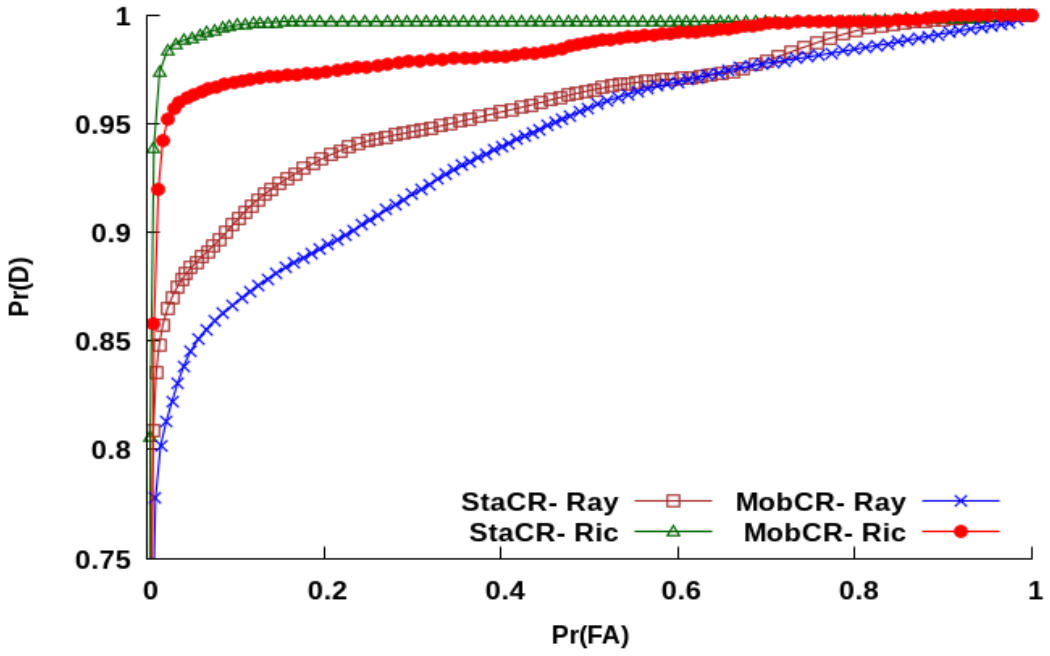


(b) Rician, [N=3SUs, M=3500s]

Figure 5.2: The performance of the various technique under the effect of the different fading channels



(a) Rayleigh vs. Nakagami, [N=3 SUs, M=1500]]



(b) Rayleigh vs. Rician, [N=3 SUs, M=1500]

Figure 5.3: Comparison the effect of different fading channels on the performance of the KMeans-based technique

Rician fading channels, figure 5.3b. This is because when the value of κ or μ increases, the dispersion of the faded signal decreases. As a result, the distance between the clusters' centroids increases.

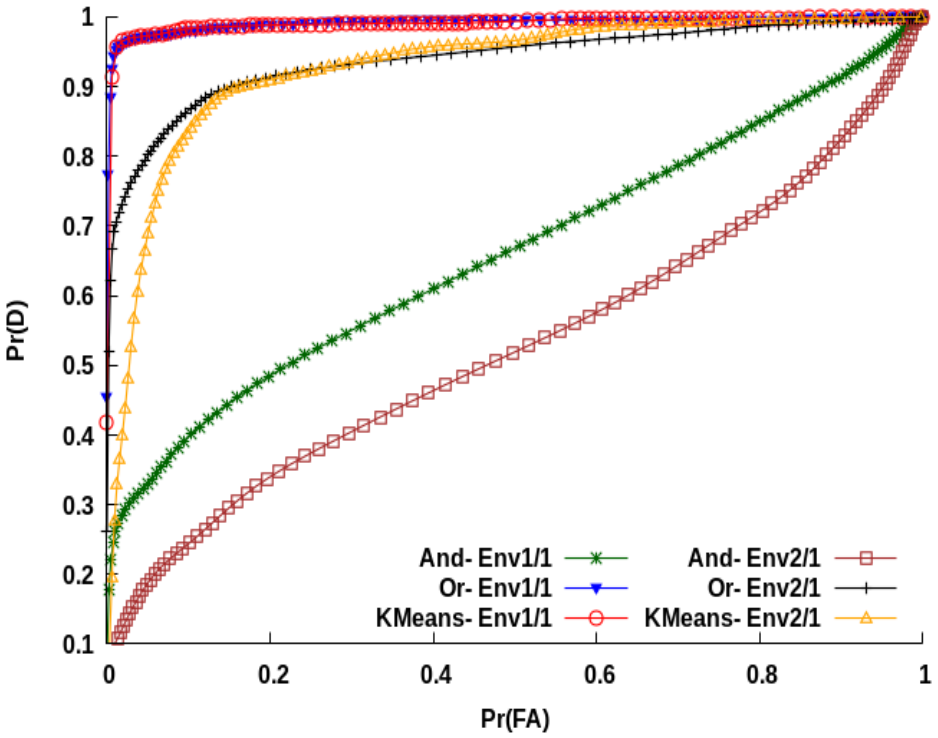
Comparing the performance of the KMeans-based technique for StaCR and MobCR for all types of fading channels as also depicted in figure 5.3, we found that mobility causes performance degradation. While increasing the parameters κ and μ leads to a decrease in the dispersion of the fading signal, the mobility increases the dispersion of the fading signal due to the spatial diversity effects of mobile nodes. Thus, the signal dispersion has a significant impact on the sensing performance.

5.3 Noisy environment

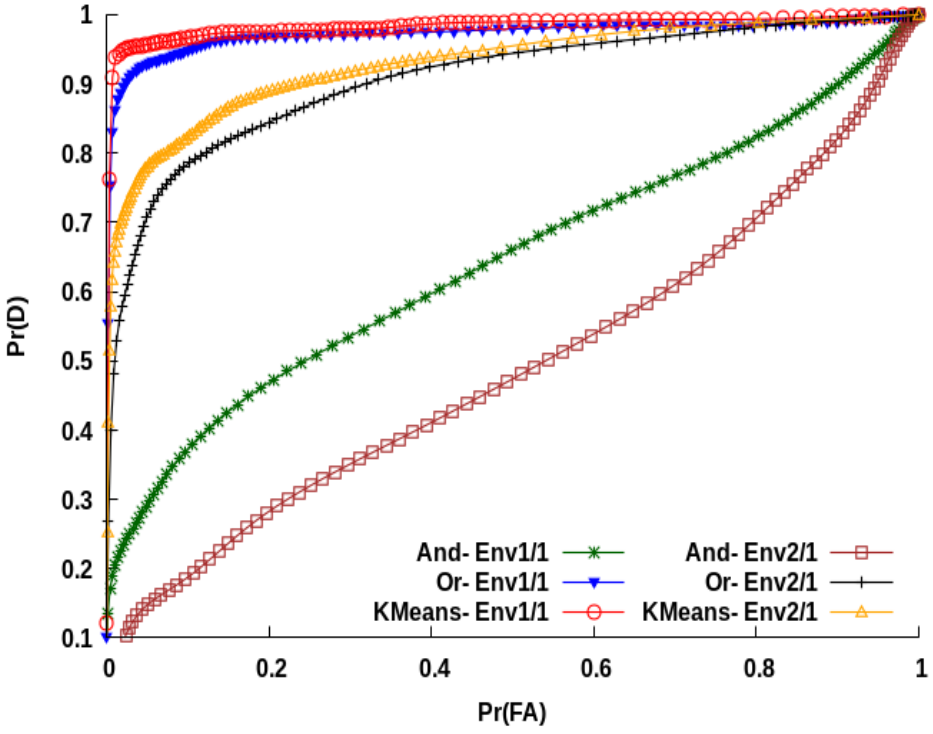
In a noisy environment, it becomes difficult for SUs to truly detect the presence of the PU signal. This is because they become unable to determine the nature of the captured signal. Figure 5.4 compares different CSS techniques in two different noisy environments, Env1/1 (μ= - 89.75, σ= 1.0) and Env2/1 (μ= - 88.25, σ= 1.5). The performance of CSS deteriorates in a high-noise environment, especially for the And-based technique.

In StaCR, figure 5.4a, the performance of the KMeans-based and the Or-based technique are comparable in a low-noise environment. However, their performance is degraded in a high-noise environment. The performance of the Or-based techniques is slightly superior compared to the performance of the KMeans-based technique. But, the KMeans-based technique is superior and more stable as compared to other techniques in the MobCR, figure 5.4b. Further, the results show that the performance of the And-based technique is degraded.

Figure 5.5 depicts the KMeans-based CSS performance for Rician fading in two noisy environments, Env1/2 and Env2/1, which are parametrized in the previous table 4.2. Figure 5.5b and figure 5.5d show that the mobility clearly affects the sensing data. The data is more dispersed in the mobility scenario, leading to altering the place of the centroids (i.e. the Euclidean distance in MobCR > the euclidean distance in StaCR). This means that mobility inevitably presents low performance. Figure 5.5c and figure 5.5d show the effects of the noise level on the sensing data. Clearly, in a noisy environ-

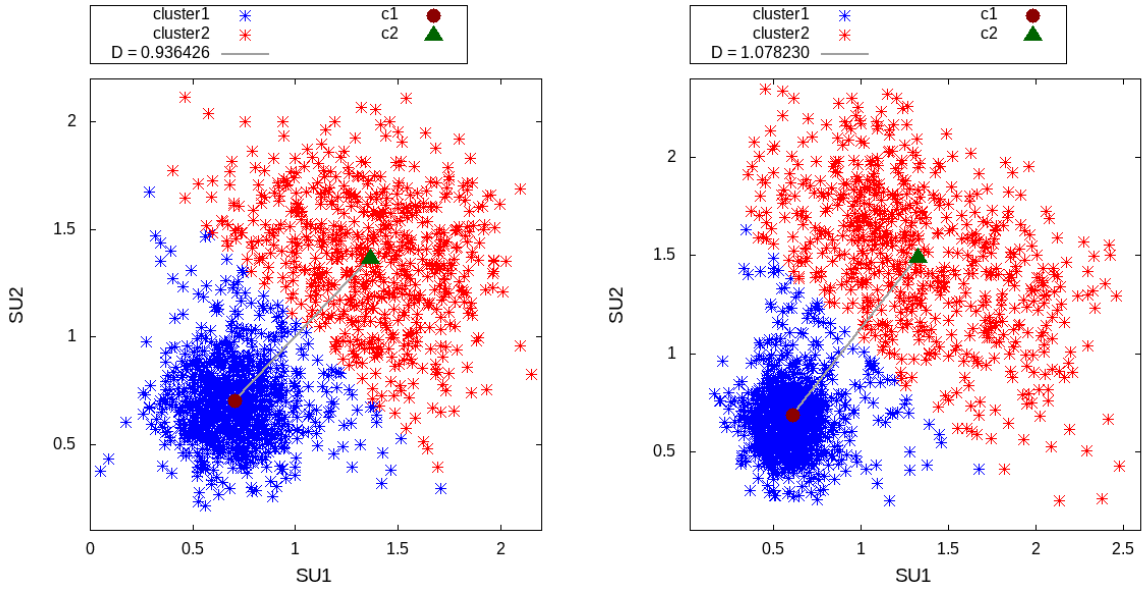


(a) StaCR of Rayleigh fading, [N=5SUs, M=1500s]



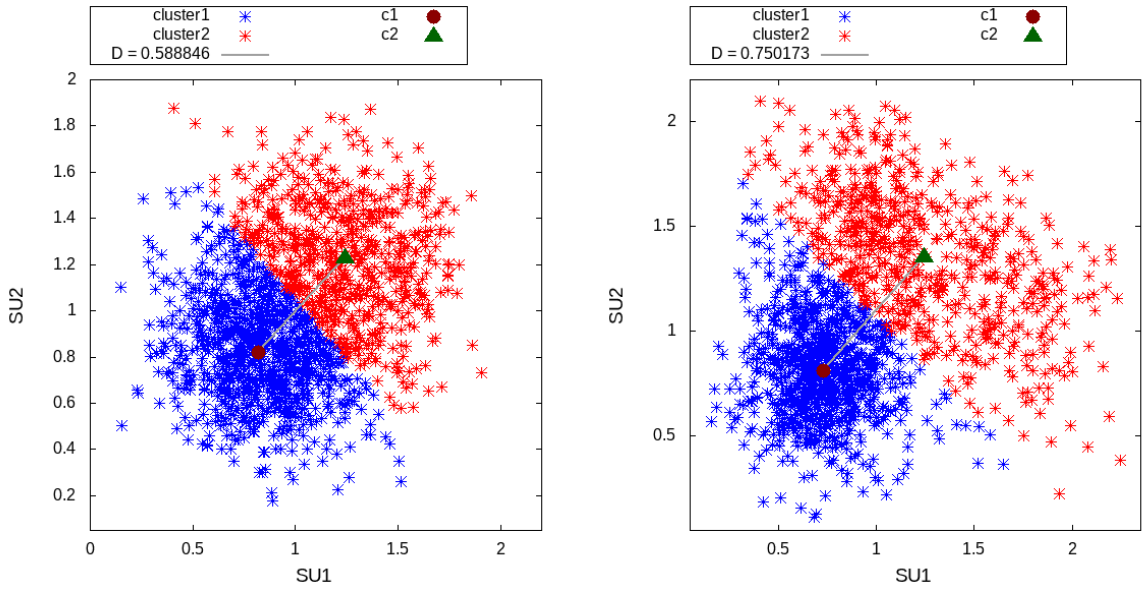
(b) MobCR of Rayleigh fading, [N=5SUs, M=1500s]

Figure 5.4: The performance of the various technique under the effect of different noisy environments



(a) StaCR, Env1/2

(b) MobCR, Env1/2



(c) StaCR, Env2/1

(d) MobCR, Env2/1

Figure 5.5: Data Clustering for different environments, [Rician, M=1500s]

ment, the clusters become more condensed. Thus, clusters' centroids become more close, causing difficulties for the ML techniques to accurately classify the sensed data.

Table 5.1 summaries the numerical results for the noisy environments, Env 1/1 and Env2/1, under the effect of different fading channels. The table depicts the probability of detection, Pr(D), verses different reference points of probability of false alarm, Pr(FA), $r1 = 5\%$, $r2 = 10\%$, and $r3 = 15\%$. By comparing the measurements, it is clear that the Pr(D) decreases when the noise level is high, and thus, the performance in a high-noise environment

Low-noise environment, Env1/1

Row's order	Ch. type	Spectrum sensing techniques	StaCR $r1 = 0.05$	StaCR $r2 = 0.1$	StaCR $r3 = 0.15$	MobCR $r1 = 0.05$	MobCR $r2 = 0.1$	MobCR $r3 = 0.15$
1^{st}	Nakagami	Non-CSS	0.8281	0.8612	0.8827	0.6218	0.6838	0.7428
2^{nd}		And	0.7207	0.7537	0.7702	0.4660	0.5340	0.5810
3^{rd}		Or	0.9950	0.9950	0.9983	0.9682	0.9773	0.9773
4^{th}		KMeans	0.9967	0.9966	0.9966	0.9788	0.9803	0.9818
1^{st}	Rician	Non-CSS	0.8397	0.8777	0.9025	0.7318	0.7712	0.7909
2^{nd}		And	0.7719	0.8033	0.8182	0.5362	0.5756	0.6373
3^{rd}		Or	0.9967	0.9999	1.0000	0.97575	0.9803	0.9818
4^{th}		KMeans	1.0000	1.0000	1.0000	0.9863	0.9879	0.9909
1^{st}	Rayleigh	Non-CSS	0.5917	0.6247	0.6528	0.5348	0.5620	0.5954
2^{nd}		And	0.3272	0.4165	0.4500	0.2954	0.3848	0.4439
3^{rd}		Or	0.9719	0.9802	0.9851	0.9303	0.9450	0.9681
4^{th}		KMeans	0.9736	0.9851	0.9868	0.9560	0.9670	0.9700

Height-noise environment, Env2/1

Row's order	Ch. type	Spectrum sensing techniques	StaCR $r1 = 0.05$	StaCR $r2 = 0.1$	StaCR $r3 = 0.15$	MobCR $r1 = 0.05$	MobCR $r2 = 0.1$	MobCR $r3 = 0.15$
1^{st}	Nakagami	Non-CSS	0.4850	0.5966	0.6650	0.4030	0.4810	0.5160
2^{nd}		And	0.4285	0.5040	0.5487	0.2239	0.2904	0.3237
3^{rd}		Or	0.8615	0.9005	0.9250	0.7700	0.8623	0.8950
4^{th}		KMeans	0.8876	0.9388	0.9586	0.8850	0.9107	0.9319
1^{st}	Rician	Non-CSS	0.5370	0.6430	0.7055	0.4136	0.4879	0.5530
2^{nd}		And	0.4770	0.5480	0.6100	0.2015	0.2712	0.3303
3^{rd}		Or	0.8980	0.9370	0.9640	0.7651	0.8742	0.9015
4^{th}		KMeans	0.9322	0.9669	0.9802	0.9045	0.9469	0.9560
1^{st}	Rayleigh	Non-CSS	0.3867	0.4297	0.4743	0.3820	0.4409	0.4636
2^{nd}		And	0.1965	0.2380	0.3060	0.1515	0.1742	0.2409
3^{rd}		Or	0.7960	0.8644	0.9000	0.7166	0.7924	0.8196
4^{th}		KMeans	0.6810	0.8331	0.8942	0.8181	0.8697	0.9010

Table 5.1: Summary of numerical results for the noisy environments Env 1/1 and Env2/1, [N=5SUs, M=1500s]

(Env2/1, the second part of the table) is degraded. The second rows of various fading channels show that the performance of the And-based technique is the lowest. The performance of this technique is far from reaching the value of 90% for Pr(D) versus the value of 10% for Pr(FA). For example, while the Pr(D) of stationary Rician fading was approximately equal 80% versus 10% of Pr(FA), it decreases too much less in most experiments (i.e., The Pr(D) is even lower 20% with the MobCR).

The performance of the Non-CSS techniques is low as shown in table 5.1. The Pr(D) was around forty percent versus 10% of Pr(FA) of the Rayleigh fad-

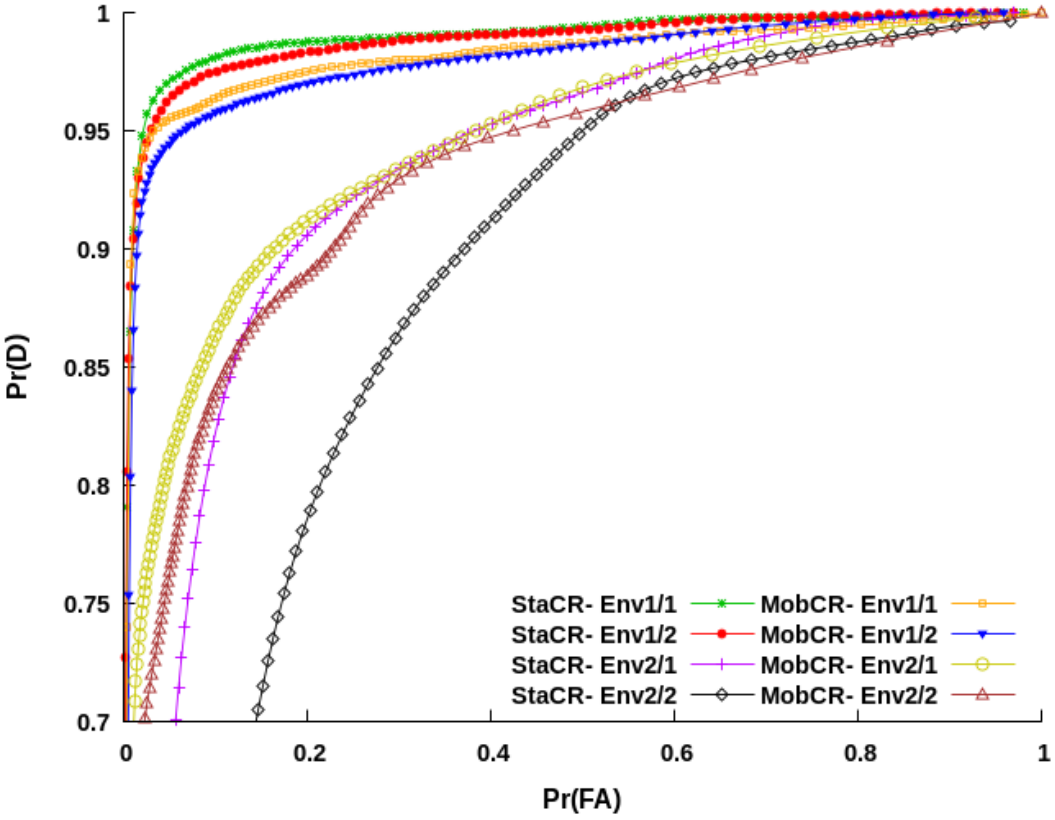


Figure 5.6: StaCR vs MobCR of KMeans-base CSS for different noisy environments, [Rayleigh, N=5SUs, M=1500s]

ing with Env2/1. Table 5.1 also shows great convergence in the performance of the Or-based and the KMeans-based techniques. They achieve high performance that reaches 100 percent versus 10% of Pr(FA) in the KMeans-based techniques under stationary Rician fading and Env1/1.

Finally, figure 5.6 illustrates the mobility and noise effects on the performance of the KMeans-based technique for Rayleigh fading. As seen, the best performance corresponds to a lowest-noise environment, while the worst performance corresponds to a highest-noise environment. An exception here to the mobility conditions, the MobCR introduces superior performance compared to StaCR in high-noise environments. Actually, this is not an exception in the true sense. The good performance comes from the approximation between the parameters of the Gaussian distribution and the parameters of the $\kappa - \mu$ distribution while this was already achieved here with the high-noise environments.

From the above results, we can deduce that the KMeans-based and the Or-based techniques with the stationary scenarios provide the best comparable

performance. The And-based and the Non-CSS techniques provide the worst performance in stationary scenarios. In mobile CR, the And-based and the Non-CSS techniques provide highly degraded performance. Also, the performance of the KMeans-based and the or-based techniques is better as compared to the And-based and the Non-CSS techniques, but at the same time not better than the stationary case. Further, the results show that at least 3+ collaborative SUs and about 1500 samples are needed to improve the performance of the KMeans-based and the Or-based techniques. Finally, we found the performance of the KMeans-based technique is stable in the high-noise environment as compared to the And-based and the Or-based techniques.

Chapter ▷6
Conclusion & Future Work

CR has been proposed as a promising solution to the problem of spectrum scarcity. The life cycle of the CR goes through spectrum sensing task, spectrum analysis task, and making joint decisions on spectrum selection task. SS is the first task of the cognitive radio life cycle during which spectral holes are detected. The most well-known SS techniques are energy detection-based, covariance matrix-based, and cyclostationary-based. Machine learning-based techniques are other modern types of innovative spectrum sensing. In these methods, the sensing process to detect the PUs activities passes through two phases, which are: The feature extraction phase, and the decision-making phase.

Literature studies conclude that the performance of the CSS schemes can be affected by many factors. First, the number of collaborating SUs. Second, the PU transmit power. Increasing the number of collaborating SUs and the PU transmit power can improve the performance. Third, the number of active PUs. Increasing the number of active PUs can deteriorate the performance because of the high interference. Finally, the number of training samples. Increasing the number of training samples increases the classification time and computational complexity. However, these studies are conducted for stationary CR. As nodes in wireless networks are normally mobile, studying the performance of CSS techniques is needed. Throughout this work, we focused on the KMeans-based technique.

A small-scale CRN was adopted in this thesis and simulated using the well-known ns3 simulation platform. The general $\kappa - \mu$ fading channel is considered. The $\kappa - \mu$ fading signal was sampled using the well-known rejection sampling method. As ns3 is a discrete-event network simulator, there was a need to develop a noise model using python language. Non-CSS and different

types of CSS techniques were considered.

This thesis concludes that:

- There are two main reasons for the degradation of CSS performance, which are: mobility and noise level. The nodes' movement and the high noise complicate the process of classifying data.
- The strength of ML techniques appears in highly faded channels.
- The And-based technique has the lowest performance in stationary scenarios, as well as in mobile CR for a small number of samples and SUs.
- Non-CSS has low performance in stationary scenarios. It also shows highly degraded performance in mobile CR. However, the Or-based and the KMeans-based techniques provide the best performance in stationary scenarios.
- Although the KMeans-based and the Or-based techniques have slightly lower performance in mobile scenarios, their performance can be improved by increasing the number of cooperative SUs and the training samples. We noticed that increasing the number of SU and number of training samples above certain values, does not lead to an increase in performance.
- In a high-noise environment, the performance of the KMeans-based techniques outperforms other techniques, whereby the clusters that represent the channel state have similar densities. This occurs when the received PU signal is weak and the noise level is high. These are the key parameters that affect CSS performance.

In the following, **ideas for future research are presented:**

- To improve the performance in mobile networks, spatial diversity may provide useful information that can be utilized to improve the accuracy of estimating the PU states. This spatial diversity results from the different SUs positions and their movement. The accurate estimate of the channel states inevitably leads to massive improvement in the sensing performance. A certainty factor that represents how much estimates of SUs are trusted can be utilized. Appendix B.1.3 shows a simple idea of

the proposed factor. Finding new methods for extracting the sensing data features can be very useful for this task.

- Further research can be devoted to other ML techniques, such as the GMM.
- In our simulations, we used the rejection sampling approach for modeling the fading. Other methods can be studied. See appendix B.1.2 and appendix A.2 for more information on the rejection sampling method.
- Correct tuning of sensing parameters, such as sensing duration and periods, probability of primary user activity, number of samples taken per sensing period, etc., has a direct impact on sensing performance. The selection of the exact values of these parameters is mainly modified based on the type of simulated networks and their protocols. Here, some limitations regarding network technologies and protocols are visual and are worth considering as directions for future research. The standards that govern these parameters constitute another aspect worth considering as well.
- Whereas different levels of SUs' cooperation are considered, we assume that all SUs that participate in the sensing process provide honest information about what has been observed from their point of view. However, the security aspect may be considered. Of course, if we assume scenarios when some adversary nodes deliberately manipulate the sensing data to falsy determine the channel state.

Appendices

Appendix ▷A

Related topics

A.1 Ns3 implementation of general $\kappa - \mu$ fading

A.1.1 Introduction

In the wireless network, when the signal travels from the transmitter to the receiver, it suffers from several types of attenuation such as path loss and multipath [9]. The value of the attenuation depends on several factors such as the distance from the transmitter to the receiver, the time, the working radio frequency, and scattered objects along the path from the transmitter to the receiver. Due to the multipath effect, several versions of the transmitted signal arrive at the receiver at different time delays. However, this time variation of the received signal power is known as fading.

The types of fading are divided into large-scale fading and small-scale fading. A large number of distributions have been proposed in an attempt to find the best model that approximates the distributions of different types of fading. In general, the large-scale fading signal is well characterized by the lognormal distribution. While small-scale signal variation is described by several other distributions, such as the Rayleigh, Rice, Nakagami-m, Hoyt, and Weibull distributions [12]. In most cases, Nakagami-m and Rayleigh fadings are described by the gamma distribution whereas the Rice fading is described by the zero-order modified Bessel function of the first kind. Yacoub et al. [16] introduced a general $\kappa - \mu$ fading model that derives these common types of fading by tuning the κ and μ parameters.

A.1.2 Model implementation

In our work, we aimed at investigating the performance of several CSS techniques in the general $\kappa - \mu$ fading channel. Therefore, there was a need to develop our $\kappa - \mu$ fading model for the ns3 simulator. Here, we employed the

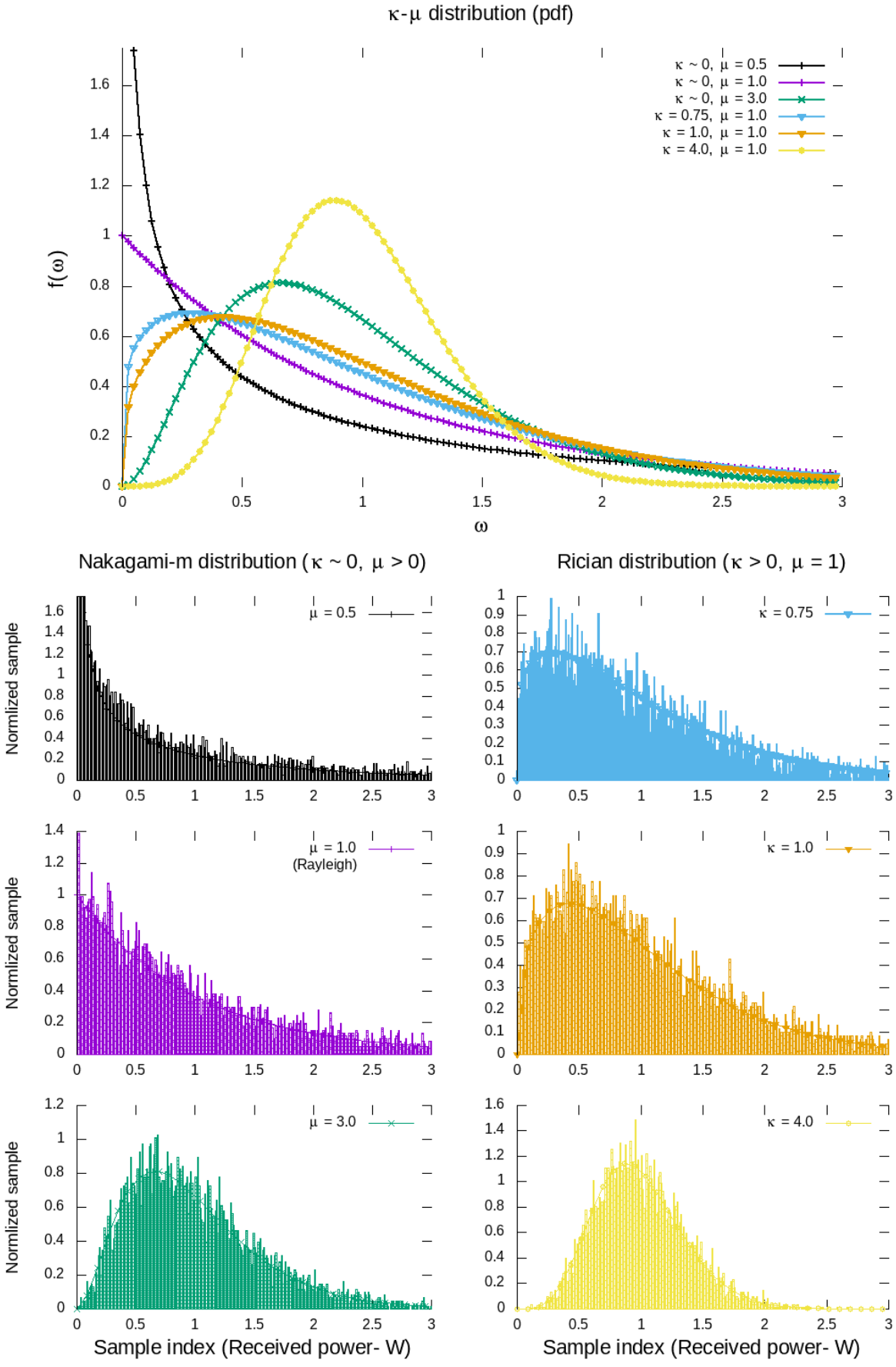


Figure A.1: The drawn samples from the KappaMuGeneralPropagationLossModel of different values of κ, μ.
$[\kappa \to 0$ with $\mu = 0.5, 1.0, \&3.0], [\mu = 1.0$ with $\kappa = 0.75, 1.0, \&4.0]$

well-known rejection sampling method to directly sample random variables from the $\kappa - \mu$ distribution [48].

Our $\kappa - \mu$ fading model for ns3 is developed as the "ns3::KappaMuGeneral-PropagationLossModel" class. The newly Ns3 developed propagation loss class must be registered to the generic interface, namely "PropagationLossModel" via the TypeId system. TypeId is a general class that records a lot of meta-information about the ns3 classes. However, we firstly model a random number class which we called "ns3::KappaMuRandomVariable".

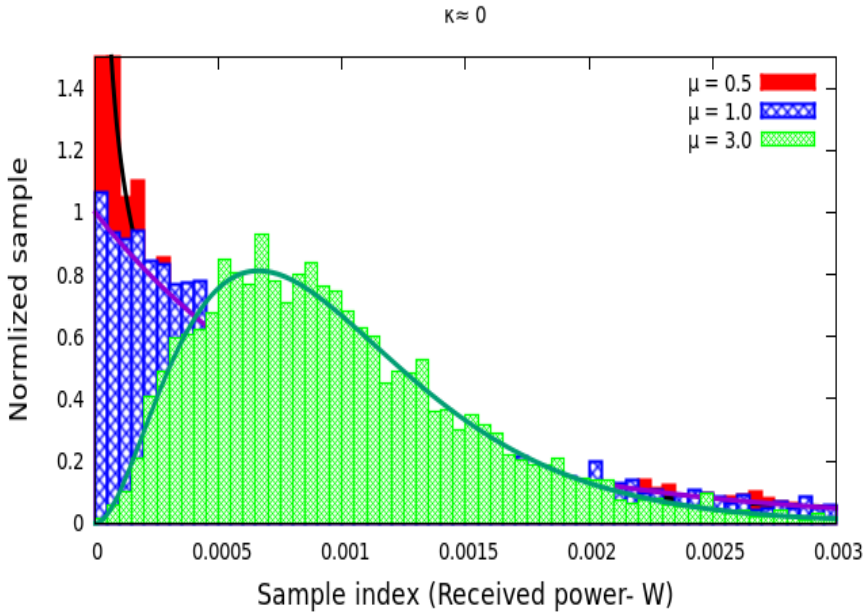
The KappaMuRandomVariable class is provided via instances of the "ns3::-RandomVariableStream" abstract class. It should implement virtual "Get-Integer (void)" and "GetValue (void)" member functions which use the rejection method to randomly draw numbers. Later, a specific member function returns these random numbers to the KappaMuGeneralPropagationLoss-Model's member function via requesting. Further, the KappaMuGeneralPropagationLossModel class should implement a "CalcRxPower (double txPowerDbm, ...)". CalcRxPower takes a transmitted power value as a parameter, then returns a floating value representing the received power toward the requesting member function.[1]

Virtual member functions GetInteger (void) and GetValue (void) are overloaded to new versions that pass three parameters corresponding to the κ and μ parameters, as well as the transmitted power. Therefore, we can control the nature of the data yielded from our propagation model.

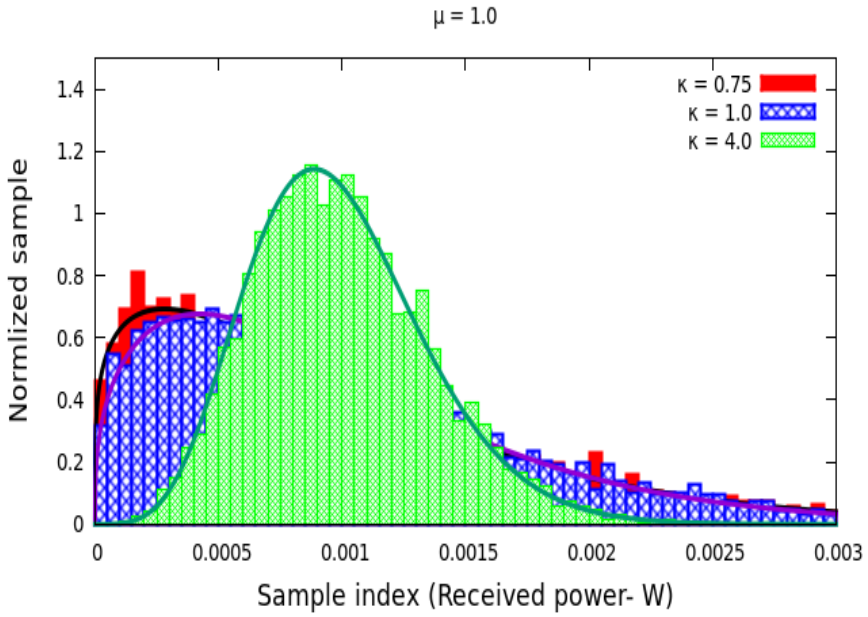
We test the histogram for more than 6000 samples that are randomly drawn from the KappaMuGeneralPropagationLossModel for different values of κ, and μ ([$\kappa \to 0$ with μ =0.5, 1.0, & 3.0], [μ =1.0 with κ =0.75, 1.0, & 4.0]). Figure A.1 depicts that the drawn random numbers that fit the $\kappa - \mu$ power probability density function. The transmission power was set to 30dbm. The developed model provides samples that are close to the exact distribution function. Further, the figure shows that Rayleigh and Rician distributions can be obtained as special cases by setting proper values for the parameters, κ and μ.

[1] `https://www.nsnam.org/docs/doxygen/classns3_1_1_random_variable_stream.html` lists the ns3::RandomVariableStream Class' APIs while `https://www.nsnam.org/docs/doxygen/classns3_1_1_propagation_loss_model.html` lists the ns3::PropagationLossModel Class' APIs

Figure A.2 depicts the effect of the transmitted power on the yielded data. Here, the transmission power was set to 0dbm. As seen, the κ and the μ parameters affect the shape of the distribution, while the power influences the spread of the distribution.



(a) The drawn samples from the KappaMuGeneralPropagationLoss-Model for different values of μ and $\kappa \to 0$



(b) The drawn samples from the KappaMuGeneralPropagationLoss-Model for different values κ and $\mu = 1$

Figure A.2: The Effect of the power on the yielded data

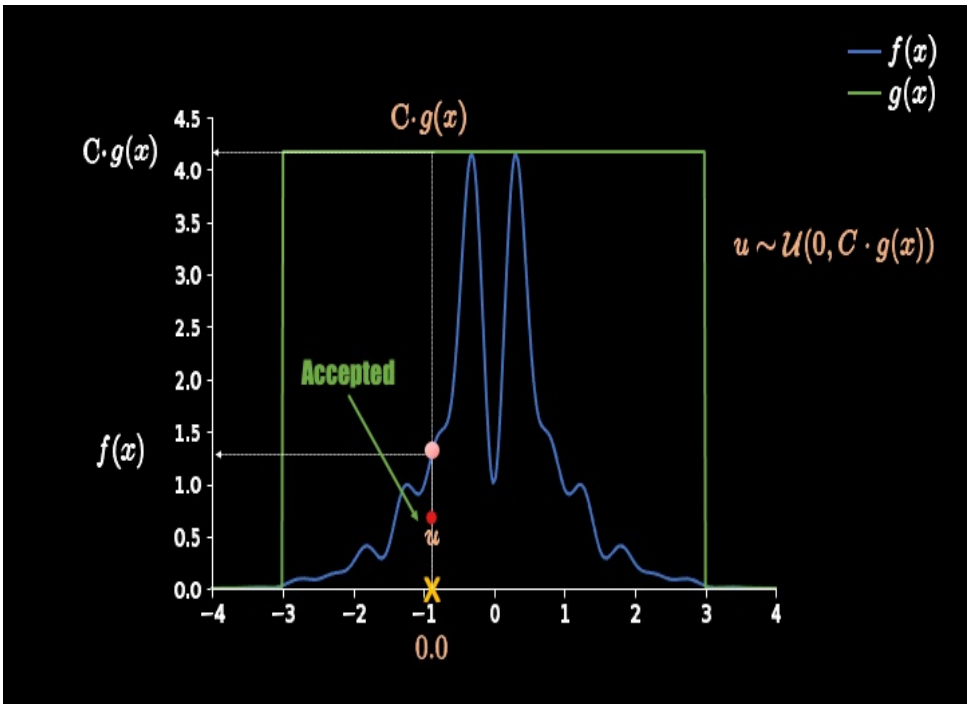


Figure A.3: The reject-accept algorithm [2]

A.2 Sampling methods

A random number generator (RNG) is hardware or computer software that generates a series of numbers or symbols that cannot reasonably be expected. There are two types of RNG are true-RNG and Pseudo-RNG (PRNG). True-RNG took its name from the assumption that it can be used to generate real sets of random numbers. Whereas PRNG algorithms conceptually provide a large number of independent streams. Therefore determining the next random number is guided by the seed. The importance of the PRNG emerges from the fact that it can be used to construct random number generators for different statistical distributions. However, PRNGs cannot be used for applications that worry about really unpredictable numbers, since constructing real RNG from deterministic things is considered impossible.

Some probability distributions can be easy to sample, while others with a complex density function are more complex. Therefore, some proposals offer ideas to facilitate this task. For example, the well-known rejection-sampling algorithm. Rejection sampling, acceptance-rejection, or accept-reject is a type of Monte Carlo random sampling method used to generate observation directly from density functions [48]. The idea behind this method is to use a well-

[2]Source: `https://towardsdatascience.com/what-is-rejection-sampling-1f6aff92330d`

defined distribution easy to sample (the proposal distribution) for sampling a more complex distribution (the target distribution).

Let $f(x)$ and $g(x)$ are respectively the target and the proposal distribution. $f(x) \leq c * g(x)$ where c is constant, then, we can use $c * g(x)$ to envelope the target distribution. Thus when a new sample comes in, it is either rejected or accepted based on its position regarding $f(x)$. Therefore, the new sample will be accepted if it falls under the target distribution while it will be rejected if it is above the target distribution. The figure A.3 shows that a redpoint drawn at random is acceptable because it lies under the target distribution. The the set of acceptance points x then follow the target distribution $x \sim f(x)$.
The algorithm works as follow:

- **Start:**
- Repeat for $\forall i$,

  (i represents the required number of samples)

  - Draw $x \sim g(x)$;
  - Draw $u \sim U(0,1)$; U: uniform distribution
  - Then, $\begin{cases} accept(x_i), & if \quad u_i \leq \frac{f(x_i)}{c*g(x_i)} \\ reject(x_i), & otherwise \end{cases}$
- **End**

A.3 Receiver operating characteristic curve

Receiver operating characteristics (ROC) is one of the primary measures of performance of classification algorithms. The Roc curve is a probability curve that plots the true positive rate (TPR)- x-axis, versus the true positive rate (FPR)- y-axis. To truly plot the ROC curve, we first need to construct a confusion matrix by comparing the actual classes with the predicated ones as follows,

		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN
*	Predicated		

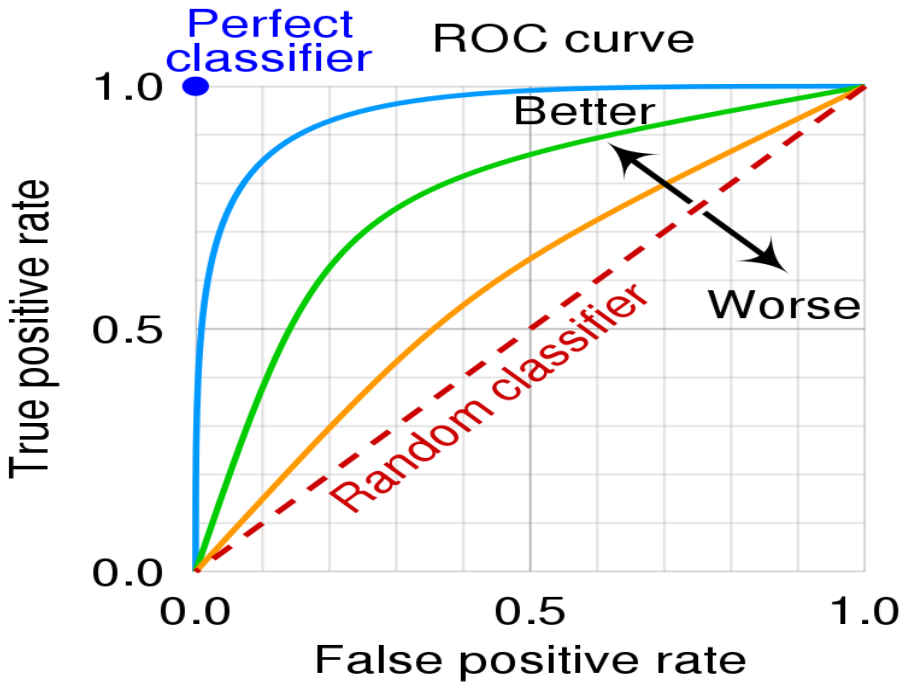


Figure A.4: Good/Bad ROC curve [3]

Then TPR (Sensitivity) can be calculated from the confusion matrix as follow, equation A.1

$$TPR = \frac{TP}{TP + FN} \tag{A.1}$$

While FPR (1 - Specificity) can be calculated from the confusion matrix as follow, equation A.2

$$FPR = \frac{FP}{TN + FP} \tag{A.2}$$

In this curve, high TPR, sensitivity, recall, or high probability of detection P(D) versus low FPR or probability of alarm P(FA) indicates the good detection while the opposite indicates the bad detection. Figure A.4 depicts the best and the worst ROC curves.

[3]Source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Appendix ▷B

Code synopsis

B.1 ns3 Code synopsis

B.1.1 Source code: The $\kappa - \mu$ general distribution

Each class in ns3 must implement a static public member function called GetTypeId (void) which registers a unique string into the TypeId system. TypeId system consists of important metadata of classes so ns3 can derive objects from the correct class.

Listing B.1: $\kappa - \mu$ general distribution, GetTypeId

```
61  TypeId
62  KappaMuGeneralPropagationLossModel::GetTypeId (void)
63  {
64    static TypeId tid =
65                          TypeId ("ns3::KappaMuGeneralPropagationLossModel")
66        .SetParent<PropagationLossModel> ()
67        .SetGroupName ("Propagation")
68        .AddConstructor<KappaMuGeneralPropagationLossModel> ()
69
70        .AddAttribute ("Kappa",
71                       "the ratio of the total power of the line-of-sight
                        ↪  (LOS) components to the total power of the
                        ↪  reflected waves kappa should be > 0",
72                       DoubleValue (DBL_MIN),
73                       MakeDoubleAccessor
                        ↪  (&KappaMuGeneralPropagationLossModel
74                       ::SetKappa,
75                       &KappaMuGeneralPropagationLossModel
76                       ::GetKappa),
77                       MakeDoubleChecker<double> ())
78
79        .AddAttribute ("Mu",
80                       "mu is a value related to multipath fading, should
                        ↪  be > 0",
81                       DoubleValue (DBL_MIN),
82                       MakeDoubleAccessor (
83                       &KappaMuGeneralPropagationLossModel::SetMu,
```

```
84                          &KappaMuGeneralPropagationLossModel::GetMu),
85                          MakeDoubleChecker<double> ())
86
87          .AddAttribute ("KappaMuRv",
88                          "Access to the underlying KappaMuRandomVariable",
89                          StringValue ("ns3::KappaMuRandomVariable"),
90                          MakePointerAccessor
                             ↪  (&KappaMuGeneralPropagationLossModel
91                          ::m_kappaMuRandomVariable),
92                          MakePointerChecker<UniformRandomVariable> ())
93      ;
94      return tid;
95    }
```

B.1.2 Source code: Rejection-sampling method

Listing B.2: $\kappa - \mu$ general distribution, accept-reject method

```
96  // Get a uniform value from KappaMuRandomVariable.
97
98  double
99  KappaMuRandomVariable::GetUniformValue (double min, double max)
100 {
101     NS_LOG_FUNCTION (this << min << max);
102     double v = rand();
103     v = v/RAND_MAX;
104     v = min + v * (max - min);
105     return v;
106 }
107
108 /**
109  * The code for the following generator functions was developed
110  * using the Rejection sampling method.
111  */
112
113 double
114 KappaMuRandomVariable::GetValue
115 (double kappa, double mu, double power)
116 {
117 NS_LOG_FUNCTION (this << kappa << mu);
118     while (1)
119     {
120         double u1 = KappaMuRandomVariable::GetUniformValue (0.0, 3.0);
121         // Set the envolop (the proposal function time constant = 100.0).
122         double u2 = KappaMuRandomVariable::GetUniformValue(0.0, 2.5);
123         double value = KappaMuRandomVariable::ComputeKappaMuPdf (u1,
             ↪  kappa, mu);
124
125     NS_LOG_DEBUG ("The power before the fading " << power <<
```

```
126     "W; \t The power to be estimate after fading " << power * u1);
127
128     // Sampling from a kappa-mu pdf, See Rejection sampling:
129     // http://en.wikipedia.org/wiki/Rejection_sampling
130
131     if (u2 <= value)
132     {
133               return u1*power;
134               break;
135     }
136     }
137 }
```

B.1.3 Certainly factor

Listing B.3: Certainly factor

```
138 double ComputeCertianly (double receivedSNR)
139 {
140         int16_t highRxPwr = -80.0; // dBm.
141         int16_t lowRxPwr = -100.0; // dBm.
142
143         double res;
144
145         if (receivedSNR > highRxPwr)
146                 receivedSNR = highRxPwr;
147         if (receivedSNR < lowRxPwr)
148                 receivedSNR = lowRxPwr;
149
150         res = 5 * (receivedSNR + 100);
151
152         return res / 100;
153 }
```

B.2 Python Code synopsis

B.2.1 Source code: Noise Modling

Listing B.4: Noise Modling

```
154 def get_truncated_normal(mean, sd, low, upp):
155     a=(low - mean) / sd
156     b=(upp - mean) / sd
157     return truncnorm(a, b, loc=mean, scale=sd)
158
159 nf = -93.9660 # noise floor
160 nmax = abs(nf)
161 nsd = 1.0
```

```
n_su = int(input("Please, input number of SUs: ")) # number of STAs
total = [0] * n_su

for i in range(l): # number of samples
    for j in range(n_su):
        # replace '0.000000' with random value follow normal distribution
        if lines[i][j] == '0.000000':
        X = get_truncated_normal(mean=89.75, sd=nsd, low=82.0, upp=nmax)
        tmp = X.rvs()
        sig = -tmp
    else:
        tmp = float(lines[i][j])
        sig = tmp
    lines[i][j] = sig-nf #recivied_pwr
    total[j] += lines[i][j]

norm = [total[i]/l for i in range(len(total))]

# normalizing the received power to the total received power
lines = [['{:07.6f}'.format(round(float(lines[i][j])/norm[j], 12)) for j
↪  in range(n_su)] for i in range(l)]
```

B.2.2 Source code: KMeans algorithm

Listing B.5: Classification process of KMeans algorithm

```
# Belonging samples to a certain cluster by means of the set of centroid
def clustering(num, clus, initCen, par):
    lagacy = par
    dis = [0 for i in range(num)]
    newClusters = [[] for i in range(num)]

    for i in clus:
        for j in range(len(dis)):
            dis[j] = CalcEuclideanDis(i,initCen[j])
        index = dis.index(min(dis))
             # append the sample feature to the closest cluster
        if index == 0:
            newClusters[0].append(i)
        else:
            newClusters[index].append(i)

    means = CalcMean(newClusters)
    variences = CalcVar(means, newClusters)

    if (variences != lagacy):
        means, newClusters = clustering(num, clus, means, variences)

    return means, newClusters;
```

B.2.3 Some of CSS techniques source codes

Listing B.6: And/Or-based CSS techniques

```
############ The and tec.
def AndRule(file1, upper, snrth):
    a = ReadContent(str(file1))
    b = []

    for i in range(len(a)):
        ld = a[i][1] >= snrth # observing hight snr
        for j in range(2, upper):
            ld &= (a[i][j] >= snrth)
        if ld == 1:
            b.append([a[i][0], -1]) # channel unavailable cluster
        else:
            b.append([a[i][0], 1]) # channel available cluster

    return b

############ The or tec.
def OrRule(file1, upper, snrth):
    a = ReadContent(str(file1))
    b =[]

    for i in range(len(a)):
        ld = a[i][1] >= snrth # observing hight snr
        for j in range(2, upper):
            ld |= (a[i][j] >= snrth)
        if ld == 1:
            b.append([a[i][0], -1]) # channel unavailable cluster
        else:
            b.append([a[i][0], 1]) # channel available cluster

    return b
```

Listing B.7: KMeans-based CSS techniques

```
# The KMeans tec.
def ML_Decision(file1, file2, upper, threshold, thr=thr):
    sum1 = sum2 = 0
    result = [];
    decision = []
    centriods = []
    a = ReadContent(str(file1))
    infile = open(str(file2), 'r')
    centriods = infile.readline()
    centriods = ast.literal_eval(centriods)
```

```
246     infile.close()
247     c1 = centriods[0]
248     c2 = centriods[1]
249     d1 = d2 = 0
250 
251     for i in range(upper-1) :
252         d1 += (c1[i]**2)
253         d2 += (c2[i]**2)
254     d1 = math.sqrt(d1); d2 = math.sqrt(d2)
255 
256     if (d1 < d2) :
257         centriods[0] = c1
258         centriods[1] = c2
259     else :
260         centriods[1] = c1
261         centriods[0] = c2
262 
263     for i in range(len(a)) :
264         for j in range(1, upper) :
265             sum1 += math.pow(a[i][j] - centriods[0][j-1], 2)
266             sum2 += math.pow(a[i][j] - centriods[1][j-1], 2)
267         result.append([math.sqrt(sum1), math.sqrt(sum2)])
268         sum1 = sum2 = 0
269 
270     for i in range(len(result)) :
271         res = result[i][0]/result[i][1]
272         if (res >= threshold) :
273             decision.append([a[i][0], -1]) # channel unavailable cluster
274         else :
275             decision.append([a[i][0], 1]) # channel available cluster
276 
277     return decision
```

B.2.4 ROC Curve source code

Listing B.8: ROC Curve

```
278 # Generating the ROC curve
279 def ROCcurve(actual, predicted):
280     tn = fn = fp = tp = 0
281 
282     # -1/ PU is exist, channel does not available
283     # +1/ PU is not exist, channel does available
284 
285     for i in range(len(actual)):
286         if actual[i][1] == +1 and predicted[i][1] == +1:
287             tn +=1
288         elif actual[i][1] == -1 and predicted[i][1] == +1:
289             fn +=1
```

```
290         elif actual[i][1] == +1 and predicted[i][1] == -1:
291             fp +=1
292         else:
293             tp +=1
294 
295     tpr = tp/(tp+fn) # True positive rate
296     fpr = fp/(fp+tn) # False positive rate
297 
298     return  fpr, tpr, fn, tp, tn, fp
```

Bibliography

[1] KA Bonsu, KO Boateng, JK Oppong, and Koffi A Dotche. Small-scale fading characteristics in cellular networks in ghana. *International Journal of Interdisciplinary Telecommunications and Networking (IJITN)*, 5(3):23–33, 2013.

[2] Simon Haykin et al. Cognitive radio: brain-empowered wireless communications. *IEEE journal on selected areas in communications*, 23(2): 201–220, 2005.

[3] Joseph Tlouyamma and Mthulisi Velempini. Channel selection algorithm optimized for improved performance in cognitive radio networks. *Wireless Personal Communications*, pages 1–18, 2021.

[4] Abdelmohsen Ali and Walaa Hamouda. Advances on spectrum sensing for cognitive radio networks: Theory and applications. *IEEE communications surveys & tutorials*, 19(2):1277–1304, 2016.

[5] Alexander M Wyglinski, Maziar Nekovee, and Thomas Hou. *Cognitive radio communications and networks: principles and practice*. Academic Press, 2009.

[6] Carlos Cordeiro, Kiran Challapali, Dagnachew Birru, and Sai Shankar. Ieee 802.22: the first worldwide wireless standard based on cognitive radios. In *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.*, pages 328–337. Ieee, 2005.

[7] Yizhen Xu, Peng Cheng, Zhuo Chen, Yonghui Li, and Branka Vucetic. Mobile collaborative spectrum sensing for heterogeneous networks: A bayesian machine learning approach. *IEEE Transactions on Signal Processing*, 66(21):5634–5647, 2018.

[8] Jue Wang, Haifeng Zhou, Ye Li, Qiang Sun, Yongpeng Wu, Shi Jin, Tony QS Quek, and Chen Xu. Wireless channel models for maritime communications. *IEEE Access*, 6:68070–68088, 2018.

[9] William Stallings. *Wireless communications & networks*. Pearson Education India, 2009.

[10] VK Garg. Radio propagation and propagation path-loss models. *Wireless Communications & Networking*, pages 47–84, 2007.

[11] Joram Walfisch and Henry L Bertoni. A theoretical model of uhf propagation in urban environments. *IEEE Transactions on antennas and propagation*, 36(12):1788–1796, 1988.

[12] Tapan K Sarkar, Zhong Ji, Kyungjung Kim, Abdellatif Medouri, and Magdalena Salazar-Palma. A survey of various propagation models for mobile communication. *IEEE Antennas and propagation Magazine*, 45(3):51–82, 2003.

[13] Masaharu Hata. Empirical formula for propagation loss in land mobile radio services. *IEEE transactions on Vehicular Technology*, 29(3):317–325, 1980.

[14] Arturas Medeisis and Algimantas Kajackas. On the use of the universal okumura-hata propagation prediction model in rural areas. In *VTC2000-Spring. 2000 IEEE 51st Vehicular Technology Conference Proceedings (Cat. No. 00CH37026)*, volume 3, pages 1815–1818. IEEE, 2000.

[15] Chintha Tellambura and A Dhammika S Jayalath. Generation of bivariate rayleigh and nakagami-m fading envelopes. *IEEE Communications Letters*, 4(5):170–172, 2000.

[16] Michel Daoud Yacoub. The $\kappa - \mu$ distribution and the $\eta - \mu$ distribution. *IEEE Antennas and Propagation Magazine*, 49(1):68–81, 2007.

[17] Williant Stallings. *Wireless Communications and Networking*. Upper Saddle River, NJ 07458 - Library of Congress, 2005.

[18] Yan Zhang, Jun Zheng, and Hsiao-Hwa Chen. *Cognitive radio networks: architectures, protocols, and standards*. CRC press, 2016.

[19] José Marinho and Edmundo Monteiro. Cognitive radio: survey on communication protocols, spectrum decision issues, and future research directions. *Wireless networks*, 18(2):147–164, 2012.

[20] Vishakha Ramani and Sanjay K Sharma. Cognitive radios: A survey on spectrum sensing, security and spectrum handoff. *China Communications*, 14(11):185–208, 2017.

[21] Nizar Grira, Michel Crucianu, and Nozha Boujemaa. Unsupervised and semi-supervised clustering: a brief survey. *A review of machine learning techniques for processing multimedia content*, 1:9–16, 2004.

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

[23] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4: 237–285, 1996.

[24] Abdelmohsen Ali and Walaa Hamouda. Advances on spectrum sensing for cognitive radio networks: Theory and applications. *IEEE communications surveys & tutorials*, 19(2):1277–1304, 2017.

[25] Sundous Khamayseh and Alaa Halawani. Cooperative spectrum sensing in cognitive radio networks: A survey on machine learning-based methods. *Journal of Telecommunications and Information Technology*, 2020.

[26] Vaibhav Kumar, Deep Chandra Kandpal, Monika Jain, Ranjan Gangopadhyay, and Soumitra Debnath. K-mean clustering based cooperative spectrum sensing in generalized $\kappa - \mu$ fading channels. In *2016 Twenty Second National Conference on Communication (NCC)*, pages 1–5. IEEE, 2016.

[27] Gounou Charles Sobabe, Yuhui Song, Xuemei Bai, and Bin Guo. A cooperative spectrum sensing algorithm based on unsupervised learning. In *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 1–6. IEEE, 2017.

[28] Chenhao Sun, Yonghua Wang, Pin Wan, and Yiqi Du. A cooperative spectrum sensing algorithm based on principal component analysis and k-medoids clustering. In *2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 835–839. IEEE, 2018.

[29] Shunchao Zhang, Yonghua Wang, Jiangfan Li, Pin Wan, Yongwei Zhang, and Nan Li. A cooperative spectrum sensing method based on information geometry and fuzzy c-means clustering algorithm. *EURASIP Journal on Wireless Communications and Networking*, 2019(1):17, 2019.

[30] Anal Paul and Santi P Maity. Kernel fuzzy c-means clustering on energy detection based cooperative spectrum sensing. *Digital Communications and Networks*, 2(4):196–205, 2016.

[31] Yonghua Wang, Yongwei Zhang, Pin Wan, Shunchao Zhang, and Jian Yang. A spectrum sensing method based on empirical mode decomposition and k-means clustering algorithm. *Wireless Communications and Mobile Computing*, 2018, 2018.

[32] Xin-Lin Huang, Fei Hu, Jun Wu, Hsiao-Hwa Chen, Gang Wang, and Tao Jiang. Intelligent cooperative spectrum sensing via hierarchical dirichlet process in cognitive radio networks. *IEEE Journal on Selected Areas in Communications*, 33(5):771–787, 2015.

[33] Ke-jun Lei, Yang-hong Tan, Xi Yang, and Han-rui Wang. A k-means clustering based blind multiband spectrum sensing algorithm for cognitive radio. *Journal of Central South University*, 25(10):2451–2461, 2018.

[34] Boyang Liu, Zan Li, Jiangbo Si, and Fuhui Zhou. Blind continuous hidden markov model-based spectrum sensing and recognition for primary user with multiple power levels. *IET Communications*, 9(11):1396–1403, 2015.

[35] Yingqi Lu, Pai Zhu, Donglin Wang, and Michel 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. IEEE, 2016.

[36] Elham Ghazizadeh, Dariush Abbasi-moghadam, and Hossein Nezamabadi-pour. An enhanced two-phase svm algorithm for cooperative spectrum sensing in cognitive radio networks. *International Journal of Communication Systems*, 32(2):e3856, 2019.

[37] Woongsup Lee, Minhoe Kim, and Dong-Ho Cho. Deep cooperative sensing: Cooperative spectrum sensing based on convolutional neural networks. *IEEE Transactions on Vehicular Technology*, 68(3):3005–3009, 2019.

[38] Olusegun Peter Awe and Sangarapillai Lambotharan. Cooperative spectrum sensing in cognitive radio networks using multi-class support vector machine algorithms. In *2015 9th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–7. IEEE, 2015.

[39] Yasmin Hassan, Mohamed El-Tarhuni, and Khaled Assaleh. Learning-based spectrum sensing for cognitive radio systems. *Journal of Computer Networks and Communications*, 2012, 2012.

[40] Hassaan Bin Ahmad. Ensemble classifier based spectrum sensing in cognitive radio networks. *Wireless Communications and Mobile Computing*, 2019, 2019.

[41] Maunil R Vyas, DK Patel, and Miguel Lopez-Benitez. Artificial neural network based hybrid spectrum sensing scheme for cognitive radio. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–7. IEEE, 2017.

[42] Sana Jan, Van-Hiep Vu, and Insoo Koo. Throughput maximization using an svm for multi-class hypothesis-based spectrum sensing in cognitive radio. *Applied Sciences*, 8(3):421, 2018.

[43] Olusegun Peter Awe, Anastasios Deligiannis, and Sangarapillai Lambotharan. Spatio-temporal spectrum sensing in cognitive radio networks using beamformer-aided svm algorithms. *IEEE Access*, 6:25377–25388, 2018.

[44] Jan Oksanen, Jarmo Lundén, and Visa Koivunen. Reinforcement learning based sensing policy optimization for energy efficient cognitive radio networks. *Neurocomputing*, 80:102–110, 2012.

[45] Mohamed A Aref, Stephen Machuzak, Sudharman K Jayaweera, and Steven Lane. Replicated q-learning based sub-band selection for wideband spectrum sensing in cognitive radios. In *2016 IEEE/CIC International Conference on Communications in China (ICCC)*, pages 1–6. IEEE, 2016.

[46] Tevfik Yucek and Huseyin Arslan. A survey of spectrum sensing algorithms for cognitive radio applications. *IEEE communications surveys & tutorials*, 11(1):116–130, 2009.

[47] Khalid L and Anpalagan A. Principles and challenges of cooperative spectrum sensing in cognitive radio networks. in: Zhang w. (eds) handbook of cognitive radio. *Springer*, 2017.

[48] Bernard D Flury. Acceptance–rejection sampling made easy. *SIAM Review*, 32(3):474–476, 1990.