جامعة تونس المنار

Université de Tunis El Manar

المدرسة الوطنية للمهندسين بتونس

école nationale d'ingénieurs de Tunis

Département Génie Électrique

Graduation Project Report

Elaborated by

Aya JENDUBI

To obtain

The National Electrical Engineering Diploma

Machine Learning Techniques for RF Fingerprinting Based Device Identication System

Hosted by

IETR - CentraleSupélec , Rennes Campus

Supported September, 21^{st}, 2022

Before the examination board :

President	: M. Faouzi BOUANI
Reporter	: M. Riadh BOURGUIBA
Supervisor	: M. Amor NAFKHA
Academic Supervisor	: M. Samir SAKRANI

Academic year 2021/2022

Signatures

Dedication

To my dear parents who have an eternal belief in every choice I ever made, have always saw light and hope in my failures and to whom i owe every step of success I would ever acheive,
To my dear friends, the very special ones, to my new family in Rennes, who stood by my side all along this journey and who supported me always,
To all my teachers, who put bricks in the wall i'm building through this career, who marked my life and who taught me the secrets of a successful work,
And finally to Prof.NAFKHA, my incredibly supportive supervisor, who has been always helping me making my own way in a new environment,

Thank you all so much, this would never be possible if you weren't here for me.

Acknowledgements

In the beginning of this document, I would like to pay tribute for people to whom I owe a great part of the success of this work.

I owe my deepest gratitude for my supervisor Prof.Amor NAFKHA, for his expertise, continuous assistance and wise guidance. Without his confidence and encourgement, this work would never have been achieved.

My thanks also go to Prof.Samir SAKRANI, my academic supervisor who provided me with priceless knowledge during my scholar years in the National Engineering School of Tunis, and who was always ready to provide me with help and assistance during my internship.

I extend my thanks to all the hosting team, notably , M.Yves LOUET, Mrs Haifa FARES, M.Ruben Salvador, M.Jean-Christophe PREVOTET, and the rest of IETR team in Rennes, for providing me a supporting and encouraging work environment .

Abstract

The wide spreading of IoT networks, using high-end as much as low-end devices, have awoken the need for robust, efficient and updated security methods.
This work makes use of the imperfections embedded within the phisical layer to implement a radiofrequency fingerprinting (RFF) technique in order to identify devices within a network.
Inspiring from previous works regarding RFF , we developed a CNN-based algorithm that is able to classify devices using the In-phase and Quadrature-phase samples recovered from the signals, and that because of their nature, cannot be completely identical from one device to another. Indeed, this technique theoretically allows the identification of a device just through the analysis of its signal.
First, our work studies the potential features that might be used as 'fingerprints', with respect to their extraction methods. Then we benchmarked machine learning classification algorithms depending on the used features.
We have Thus developed our classifier, whose architecture is influenced by ORACLE, the first CNN-based identification algorithm that have achieved near perfect results with over-the-cable transmission. Neverthless, our work's main contribution is the implemetation of machine learning and data pre-processing techniques to prove the performance of CNN-based classifiers on wirelessly transmitted data and with high-end USRP devices.
Our model has achieved 99.9% of accuracy on data transmitted over-the-air up to 5 devices. Neverthless, to achieve similar performance on a greater number of devices and under extreme circumstances, we have proposed additional techniques to make the classifier robust to the variation of distances between the transmitter and the reciever .

Key Words : Radiofréquency Fingerprinting, I/Q samples, Machine learning, CNN , Pre-processing.

Résumé

La large diffusion des réseaux IoT, utilisant autant des appareils haut de gamme que des appareils bas de gamme, a réveillé le besoin pour des méthodes de sécurité robustes, efficaces et mises à jour.
Ce travail utilise les imperfections intégrées dans la couche physique pour mettre en œuvre une technique d'empreinte radiofréquence (RFF) afin d'identifier les appareils au sein d'un réseau.
En s'inspirant des travaux précédents concernant la RFF , nous avons développé un algorithme basé sur CNN qui est capable de classer les appareils en utilisant les échantillons en phase et en quadrature de phase récupérés à partir des signaux, et qui, en raison de leur nature, ne peuvent pas être complètement identiques d'un appareil à un autre. En effet, cette technique permet théoriquement l'identification d'un appareil uniquement par l'analyse de son signal.
Premièrement, notre travail étudie les caractéristiques potentielles qui pourraient être utilisées comme "empreintes digitales", en foncrion de leurs méthodes d'extraction. Ensuite, nous avons comparé les algorithmes de classification d'apprentissage automatique en fonction des fonctionnalités utilisées.
Nous avons donc développé notre classificateur, dont l'architecture est influencée par ORACLE, le premier algorithme d'identification basé sur CNN qui a obtenu des résultats presque parfaits avec la transmission par câble. Néanmoins, la principale contribution de notre travail est la mise en œuvre de techniques d'apprentissage automatique et de pré-traitement des données pour prouver les performances des classificateurs basés sur CNN sur des données transmises sans fil et avec des dispositifs USRP haut de gamme.
Notre modèle a atteint 99,9% de précision sur les données transmises par voie hertzienne jusqu'à 5 appareils. Cependant, pour obtenir des performances similaires sur un plus grand nombre d'appareils et dans des circonstances extrêmes, nous avons proposé des techniques supplémentaires pour rendre le classificateur robuste à la variation des distances entre l'émetteur et le récepteur.
Mots clés : Empreintes radiofréquence , Echantillons I/Q, Machine learning , CNN , Pré-traitement.

Contents

List of Figures

List of Tables

Introduction

The use of Internet of Things (IoT) is no longer exclusive to intelligent industries, smart establishments and sophisticated healthcare application. Now-days, the blooming of research in the IoT area imposes a connected reality concept, where spoofing risks can't be tolerated and applications' safeguard is no more optional.
The 'globalization' of IoT comes with two facts regarding informations' security : the first is that more applications results in more sensitive data being transferred, more protocols, and more communication vulnerabilities, and the second is the expansion of low cost IoT devices, which presents more weaknesses in term of security.
Besides, a great portion of more complex applications require communication architecture's reconfigurability for which Software Defined Radios (SDR) are used considering that they allow cross-layer reprogrammability .

The rising need for communication's security encouraged the research within device authentification's techniques, beyond traditional identification relying on MAC and IP addresses , which are actually susceptible of great spoofing risks. These researches are increasingly oriented in physical-layer as it is becoming a promising and more efficient alternative. Radiofrequency fingerprinting for instance, is more interested in identifying devices' 'signatures' (aka "fingerprints") within their emitted signals. A lot of work has been done during the recent years in order to build models that profit from these small electromagnetic imperfections to distinguish a device out of a group.
And just like almost any other technology-related area, the exponential development of machine learning has inspired various researchers to apply such tool, that is proven efficient in other domains, for radiofrequency fingerprinting purposes.
It's in this perspective that this work finds its basis as it aims to study the state of the art of machine learning techniques for radiofrequency fingerprinting. Then, inspires from a CNN architecture that has shown 85 to 99% of accuracy for device identification, to build a model that is trained on the publishers' dataset first, then on the local testbed's dataset, and shows nearly perfect accuracy for non-variant channel and 96% to 99% for over-the-air transmission.

Chapter 1

Hosting establishment and project overview

1.1 Introduction

This chapter includes a presentation for the internship's environment, the hosting team and institution, as well as the context and the objectives of the project. It also presents an overview of the report's organisation citing all the major parts for clearer and more understandable view.

1.2 Internship's environment

This project was proposed as a Master project by the Institut d'Electronique et des Technologies du numéRique (IETR) and localized in CentraleSupélec, Rennes Campus, where a part of the hosting team 'Architecture, Systems, Infrastructure and electroniCs' ASIC is conducting their research. The following sections presents each of these entities.

1.2.1 Institute of Electronics and Digital Technologies (IETR)

IETR is a public research laboratory founded in 2002 and implanted in Brittany and Pays-de-la-Loire regions of France. Its structure is based on 6 main departments, within which work 13 research teams on a diversity of field in the area of electronics, signals and digital technologies.
The research work of more than 350 researcher within IETR is carried in five establishments INSIS of CNRS , INS2I of CNRS, CentraleSupélec, INSA Rennes, The university of Nantes and the University of Rennes 1, and it hosts as well work collaborations with uni-

versities and institutes from around the world and therefore welcomes visit and contracted researchers.

1.2.1.1 Architecture and working fields

As shown in the following diagram in figure 1.1 of IETR, the six teams (ADH , AUT, IMAGE, OS, OASiS, SCEE), are actually devided into 13 sub-teams (FunMAT, STAR, SUMIT, AUT, AIMAC, VAADER, MULTIP, eWAVES, POLARIS, SurfWave, OASiS, SIGNAL, and ASIC) working on more specific fields.
Their main themes are :

- Electronic micro-technologies and transconducters
- Antennas and Complex radiating systems in frequency spectrum from MHz to sub-THz
- Control of EM waves in random environments
- Remote sensing and propagation (detection and localisation)
- Digital communication systems and associated processing
- Smart, high-performance, efficient and adaptable embedded systems.
- Image processing, video coding and artificial intelligence
- Automation

1.2.1.2 Publications and achievements

The research annual activities of IETR amounts to an average of 75 project per year , 10 patent families, and the production of 270 publications and 40 PhDs per year.
IETR is constantly in interaction with the research and entreprenerial environment via more than 150 partnerships in the context pf CIFRE PhD Thesis, RD collaborations..etc. It is to mention, as well, that the organism commits in the creation of start-ups who are based on the research results of IETR. This orientation have resulted in the creation of 5 start-ups during the last 5 years.
According to the Scimago Institutions Ranking, a classification of 8089 academic and research-related institutions around the world based on research performance, innovation outputs and societal impact measured by their web visibility, IETR is the world's 485th in terms of research ranking 314th in terms of innovation ranking and that, refers to the

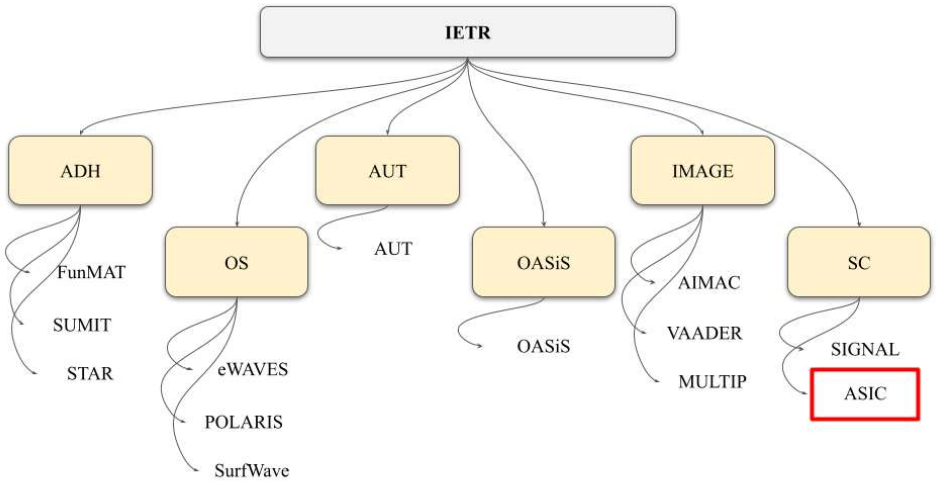

Figure 1.1: IETR teams diagram

volume, impact and quality of the institution's research output as well as the number of patent applications of the institution and the citations that its research output receives from patents.

1.2.1.3 ASIC team

The team ASIC (Architectures Systems, Infrastructures and electroniCs) part of the SCEE team is addressing problems of the transmission chain in its entity through subjects regarding , modeling, digital and analog architectures, embedded systems and IoT applications notably their design, efficiency and security.

1.2.2 CentraleSupélec (CS)

CentraleSupélec is one of the 204 French engineering schools, ranked 3rd in france according to *L'Etudiant* and a part of Paris-Saclay university.It is located in three campuses in France, Paris, Metz and Rennes Campus where are hosted research teams within two different laboratories, IRISA and IETR.
The internship project is hosted within the ASIC team of IETR in CentraleSupélec where the SCEE's testbeds are located.

1.3 Project overview

1.3.1 Context

Radiofrequency fingerprinting have recently been the heart of research work since, to our knowledge, it's on of the most reliable authentication techniques considering it presents a near perfect immunity against risks of forging or spoofing.

In such techniques, authentication is achieved by recognizing the transmitting device directly through device identifiable physical imperfections (i.e statistical moments of I/Q amplitude, phase and frequency prole, turn-on transients behaviours, carrier frequency offset, power amplifier non-linearity, and I/Q offset, . . .). Therefore, RF Fingerprinting is a process that identifies an unique transmitter through discriminating features extracted from its intrinsic physical properties.

Thus, these features are referring to the imperfections that are inherent to the transmitter's hardware makeup and are, hence, very hardly replicated. Due to the advantages these features provide, a lot of statistical and machine learning methods have been applied to track the finest patterns that would make the most accurate device identification. The outstanding performance of machine and deep learning in image and speech processing and recognition was a motivational element to deploy these approaches on radiofrequency fingerprinting problems.
The continuous search for the ideal algorithm, has led us to a variety of options shifting between pure statistical methods and deep neural networks. Yet, the performance of these algorithms is also question of the features they're making using of.

For instance, I/Q imbalance is a feature is a very neural network-friendly feature , due to its aptitude to be stored in complex format which opens doors of possibilities of utilization of the values. It's also produced using SDR and USRP platforms and has shown relatively high accuracy in application with deep neural network especially.

1.3.2 Project objectives

The ultimate goal of this project is to propose machine learning / deep learning algorithm that is capable of recognizing a device out of a group of devices using a time series sequence of the emitted signal, with regards to performance (accuracy in our case) and complexity.

- Study the existing state of the art of physical-layer identification and radiofrequency fingerprinting techniques. Evaluate the techniques in terms of complexity and performance.

- Focus on the published work : ORACLE, a CNN model that achieved 99% of accuracy on cabled transmission. Afterwards, we implement the architecture of ORACLE's CNN , train it on the ORACLE published dataset and reproduce the paper's results.

- Propose data processing techniques and modifications on the model's parameter with respect to the architecture, in order to optimize its performance , i.e raise the accuracy for over-the-air-transmission .

- Suggest methods of complexity reducing in order to optimize the computational efforts and the training time.

1.4 Document Structure

The purpose of this report is to provide a walk through the theory and the steps behind the machine learning part of this project as it treats the problem of classification of the data, which is concretely the I/Q samples stored in SigMF files, process it and classify it to achieve authentication as precisely as possible.
The second chapter of this report will present the problem statement in more precise aspects as it digs into the details of radiofrequency fingerprinting and underlay the important concepts. It also provides a walk through the commonly used technique for machine learning in the RF fingerprinting area. The third chapter is consecrated for the state-of-the-art of RF fingerprinting using machine learning , and the supporting works. We present an overview of the work ORACLE, its origins, its performance, and the vulnerabilities to improve.
The final chapter presents the adopted approaches, techniques and architecture. We justify the choice of features, the algorithm and the classifiers' parameters in reference to ORACLE's papers. We also explain the pre-processing steps and evaluate the model's performance under different circumstances, all while citing the limitations and the potential improvements.

1.5 Conclusion

This chapter presents a walk through the projects' environment , objectives and and the structure of this document. In the next chapter, the attention will be drawn to the key concepts and techniques of radiofrequency fingerprinting as a preamble to the state of the art which we will present later in this document.

Chapter 2

Techniques of radiofrequency fingerprinting (RFF)

2.1 Introduction

Before exploring the techniques of fingerprinting, it's essential to introduce the origins of this concept, its development through the previous two decades and the need for integrating deep learning in this area of research. This option comes with other problematics such as the choice of features, the model's complexity and architecture. These concepts will all be covered in this chapter in the first section along with a taxonomy of the most used machine learning techniques in the rest of the chapter.

2.2 Radiofrequency fingerprinting

As wireless communication is prone to malicious attacks, the innovation in terms of security has brought out several schools of device authentication. The conventional type of protection for IoT networks is relying on multiple-protocol layers : MAC-layer , network-layer and transport-layer. Yet, these methods do not provide a perfect immunity against impersonation attacks.
A relatively new school proposes the alternative that is built on physical-layer identification, as it defines the process of extracting features that are resulted of hardware imperfections in the analog circuitry, and use these, to 'fingerprint' devices. The efficiency of this method goes back to the fact that despite the enormous effort that manufacturers invest in processing an ideal device, they can never prevent the hardware impairments.

Notably due to these impairments , we can not only distinguish different devices from different models but also different devices within the same model just as we will be citing

in the state of the art. Better yet, algorithms have proven efficiency in distinguishing bit-similar devices in a shared spectrum environment.

2.2.1 Principle, history and applications

Despite that the research work in the area of radiofrequency fingerprinting has bloomed recently due to the security challenges imposed by the viral spread of internet of things, the history of physical-layer use goes back to the nineties, when Choe *et al* [1] have analysed communication signals in order to distinguish equipments in 1995. Then, it was introduced as a concept in 2004 by Hall *et al* [2], and a growing number of techniques have been launched ever since.

This concept is built on 3 stages :

- Capturing signals
- Feature extraction : make use of the captured signal (or a part of it) to extract unique features having the characteristic information of the corresponding device.
- Classification : the stage where an algorithm is proposed to "learn" these information and therefore classify devices accordingly.

Each of these steps will be explained later in the chapter.

Furthermore, It is to note , that these emerging techniques have been recently deployed in several domains, such as ADS-B systems in Air Traffic Control [3] to identify aircraft, in Bluetooth devices [4], RFID [5] and push-to-talk-transmitters [6]. And it's here that low-end devices interfere as a preference for several research work, and a challenge in terms of the practical limits they present for RFF.

2.2.2 Features

Various studies have explored different types of the emitted signal, as it's the kernel of radiofrequency fingerprinting, in order to find the features that give the best accuracy.

2.2.2.1 Features properties

We can classify features according to several criterias, yet there are essential properties that these features or so called 'fingerprints' should satisfy:

1. Uniqueness: The features should be specific to devices. Between two devices, a difference within these features must be present.

2. Universality: They should exist in all devices

3. Robustness: These features should be environment proof, *i.e* their performance should be evaluated with regard to external environment device-related aspects

4. Permanence: The fingerprints should be time and environment invariant.

5. Collectability: These features should be measurable.

2.2.2.2 Taxonomy of features

A great way to categorize features is by what part of the signal is targeted during feature extraction. According to [7] we can recognized 3 schools of feature extraction as cited below.

1. **Transient-Based RF fingerprinting**
   This type of techniques spot the transition between the turn-off and the turn-on of the transmitter device, which consists in the phase before the transmission of the real information of a signal. Hence, these approaches require precise transient extraction. It is also to mention , that these techniques are also subjects of channel noise and hardware impact, which relates the efficiency of these techniques to the accuracy of the transition extraction.
   There are several types of transient signals and so multiple transient extraction methods to detect the start point as precisely as possible, such as Bayesian Step Change Detection (BSCD), Bayesian Ramp Change Detection (BRCD), Variance Fractal Dimension Threshold Detection, Phase Detection (PD), Mean Change Point Detection (MCPD),Permutation Entropy (PE)and Generalized Likelihood Ratio Test (GLRT) Detector, *etc*...

2. **Steady state-Based RF Fingerprinting**
   These approaches target the modulated part of the signal, since it's the non-common part from a transmitter to another, and pull out directly the fingerprints such as frequency error, SYNC correlation, I/Q origin offset, modulation shape, Power Spectral Density coefficients, etc.. Some work that is based on these techniques will be

presented later in the state of the art section. It is to mention that these techniques were mostly used earlier, due to the difficulties in the transient extraction task.

3. **Other Approaches**

   This type of approaches utilizes a specific wireless technology and extract other attributes of the signal and logical layer. In this area, researchers might combine multiple features(PSD, timing, modulation-shape..) or even tend to use random parts of the signal (for example preamble, near-transient part).

   Ergo, in this concept, the main purpose defines the tools (features in this case) i.e fingerprints are specified according to the device catalogue that constructs the database (same model, same manufacturer; various manufacturers, different model;..) and the classification problem .

Another way to define the taxonomy of features is categorizing them into location-dependant and location-independant or radiometrics.

- **Location-dependant features**
  The point of these features isn't only to define the device, but also, to define its location. Radio Signal Strength (RSS) is highly frequent in these related studies, since it's subject to variations due to the attenuation of the channel and the transmission power at the transmitter. Another very common feature is the Channel State Information at the Receiver (CSIR).
  Yet, despite the performance that these features might show from one study to another, they are not reliable for fingerprinting since they are environment-sensitive.

- **Location-Independant Features**
  Radiometrics are features that only depend on the hardware imperfections present in the physical layer of a device. Therefore, they are the most reliable fingerprints. Obviously, these impairments have tiny to null effect on communication process, yet they allow a loyalty to the device fingerprint. For instance, the coefficients of Power Spectral Density (PSD) and normalized PSD were used to extract fingerprints. Also, techniques such as Discrete Wavelet Transform (DWT), were used to extract features like phase, amplitude , phase angle and frequency from the turn-on transient portion of signals. Other researchers have shown interest to power amplifier, as it is the last element in the transmitter's circuit and it is hard for attackers to modify with software, for feature extraction. Phase errors and I/Q origin offset also have shown

performance in this field of research.
The radiometric's catalogue remains so diverse and a lot of them will be cited in publications later in this chapter.

2.3 Classification algorithms

In this phase, an algorithm is proposed to make use of the extracted features and classify devices according to their fingerprints.

2.3.1 Types of Algorithms

Supervised algorithms and unsupervised algorithms are the two categories into which classification algorithms can be subdivided. For supervised algorithms the classifier is built on a set of labeled data. The algorithm is then trained on these informations and learns to predict a device based on a sample of the informations present within the 'fingerprints'.

For unsupervised algorithms, there is no training, the algorithm learns to directly classify data via a function that it defines. We can, thus, foresee that, these algorithms would prove efficiency rather on similar data, i.e devices from the same manufacturer and same model.

2.3.2 Machine learning

Machine learning uses statistical learning algorithms to build smart systems. These systems can automatically learn and improve without explicitly being programmed. It is, thus, an artificial intelligence branch that is built on statistical methods in order to handle certain tasks without following the conventional path of traditional programming i.e providing data and rules for the computer to calculate the answers. ML works in a quite revised way where we provide data and answers and it's the computer's job to come up with rules or functions that can be generalized on a variety of other data.

2.3.2.1 Categorization of machine learning algorithms

Machine learning algorithms can be categorized in 3 main categories:

- Supervised Learning : Where they perform mostly either classification or regression. Classification is the process of sorting data into categories or pre-provided labels. Regression on the other hand, is all about predicting continuous variables.

- Unsupervised Learning : It consists in 2 main concepts: clustering , which means that the computer acquires the ability of grouping data together, and dimensionality reduction, which consists in condensing the features.
- Semi-supervised Learning : it's where the reinforcement learning comes in which means that we train the model to act in a certain way given certain agents and environment via a reward concept.

We cite here some of the most frequent machine learning algorithms (from the three categories) in solving RFF problems.

2.3.2.2 Support Vector Machine (SVM)

One of the very common tasks in machine learning is to classify a group of objects into categories, SVM is one of the simplest and most efficient methods that an algorithm could use. A trivial way to see the task is, an object is represented as a point in a bi-dimensional space. The coordinates of this point are features.
SVMs perform the classification task by drawing a hyperplane that is a line in 2D or a plane in 3D in such a way that all points of one category are on one side of the hyperplane and all points of the other category are on the other side.
While there could be multiple such hyper-planes, SVM tries to find the one that best separates the two categories, in a sense that it maximizes the distance or so called 'margin' to points in either category . The supporting vectors are the set of points that fall exactly on the margin as shown in the figure 2.2.

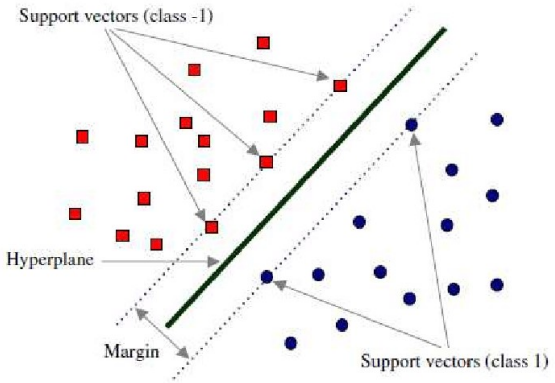

Figure 2.1: SVM parameters

To find this hyperplane in the first place, SVM require a training set i.e a set of points already labeled with the correct category, hence the supervised nature of SVM.

In the backend of the algorithm, SVM solves a convex optimization problem that maximizes the margin and where constraints say that points of each category should be on the correct side of the hyperplane.

SVMs are simple to understand methods, easy to implement (pre-developed libraries in several languages), smoothly interpretable, and very effective when dealing with small training datasets. Yet, this simplicity can limit the performance where for example in many applications, the points cannot be separated by hyperplane. A common workaround in this case is to augment data with non-linear features that are computed from the existing ones, to find the separating hyperplane in this higher dimensional space , and finally project back to the original space.
Other techniques such as Kernel Trick allows us to perform the previously described method in a very efficient way.

2.3.2.3 K-nearest neighbors (KNN)

K-nearest neighbors is a very simple machine learning algorithm that can be solved in both classification and regression problem.
The principle is explained in the two-dimensional example shown in the figure 2.2, where the objective is to classify the interrogation point in the middle into one of the two groups. In order to find the k-nearest neighbors of this given point, we need to calculate the distance between this point and the other elements of the classes.

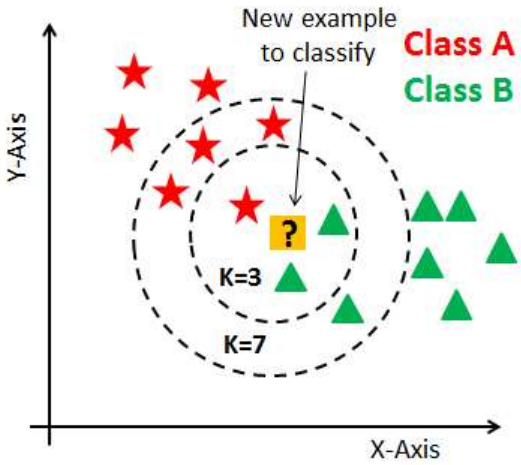

Figure 2.2: K-NN principle and parameters

There are many distance functions, but Euclidean is the most commonly used. Then, we need to sort the nearest neighbors of the given point by the distances in increasing

order.
For the classification problem, the point is classified by vote of its neighbors, then it's assigned to class most common among its k nearest neighbors. K value here controls the balance between over-fitting and under-fitting.
The best value can be found with cross validation and learning curve. A small k value usually leads to low bias but high variance, and a large k usually leads to high bias but low variance. It's important to find balance between them.

2.3.2.4 K-means clustering

It's a clustering algorithm where the input is a set of points of the dataset , and K is the number of clusters of similar data points we want to find within the dataset. We can see how data points are similar based on how far apart they are on a graph.
K-means works through the use of centroids, that are just a term for the cluster center, therefore the number of centroids that are created corresponds to the number of clusters that will be created .
To be specific a centroid isn't a data point, it's rather a representation of the cluster's center. The following figure 2.3 represents the algorithm of K-means in several iterations.

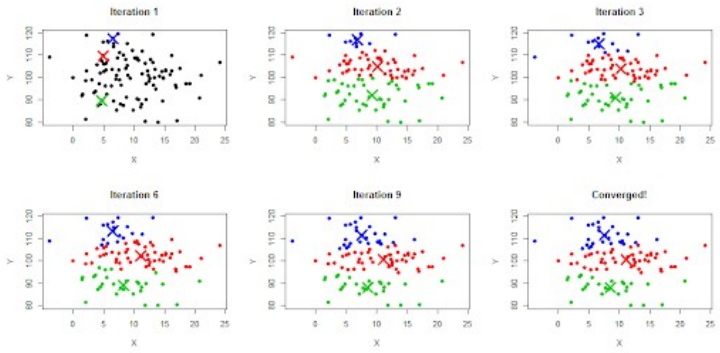

Figure 2.3: K-means Algorithm description

In the graph we dispose of set of black points and we instantiate 3 different centroids represented as a blue, green an red X marks. The choice of placement of the centroids is random. The second phase of the algorithm consists in measuring the distance between a point and the three centroids and assign it to the nearest centroid's cluster. Now that all the points belong to clusters we go on to next step where we calculate the mean of each cluster, and we repeat the measure and cluster operation using the mean value. Then the algorithm calculates the total variation within the clusters and remake the same operation

until the clusters no longer change.
Although k-means is one of the most used algorithms especially in RFF it represents several weaknesses such as its lack of efficiency when the data is limited. The fact that k is chosen manually may as well present a deficiency, even though in RFF, we normally know K in prior since we know the device's set to classify. Another important vulnerability is due to the fact that it's sensitive to outliers.

2.3.2.5 Principal Component Analysis (PCA)

In cases where the dataset presents problems of '$over-dimensionality$' , PCA is a great option to perform data compression and dimensionality reduction. In a dataset, PCA is all about considering all variables, combining them in a smart way and producing new dimensions that are correlated with each other and ranked from the most to the least important i.e a set of orthogonal variables containing the important information of the data and called Principal Component Analysis . This set is constructed in a way that if we restrict our view to only one variable we will still have a faithful representation of the data. In order to reduce dimensionality yet still preserve the maximum info, PCA solves this optimization problem using Lagrange Multipliers. This solution is highly appropriate for phone identification to reduce a large set of features.

2.3.2.6 Neural Networks (NNs)

In this section we describe the principle, the structure and the most common uses of NNs.

- **Principle**

Neural networks form the base of deep learning, a sub-class of machine learning. These algorithms are highly inspired by the structure and functions of the brains neural systems. They are based on a collection of connected units called neurons, where each of these connections can transmit a signal from one neuron to another. The receiving neuron then processes the signal and then signals downstream neurons connected to it.

- **Structure**

Typically, neurons are organized in layers. Different layers perform different types of transformations on their inputs. Signal essentially travel from the first layer called the input layer , to the last one called output layer. Layers in between are called hidden layers.

A very simple architecture of a neural network is the Multi-Layer Perception (MLP) where we dispose of only one hidden layer.

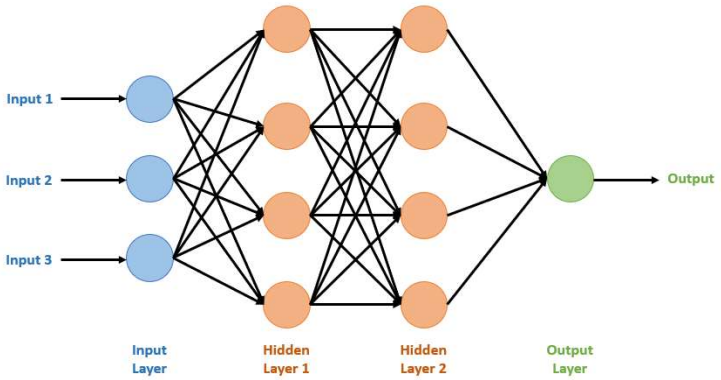

Figure 2.4: Simplified scheme of neural network

- **Learning process**

A simple way to picture a neuron is a unit that holds a number that characterizes its activation degree. The neurons of the last layer for example corresponds to our classes or labels, and therefore the activation of each neuron characterizes how much the system 'thinks' that the inputs corresponds to that certain label .
In hidden layers, though, the activation of one layer determines the activation the next via the connections between them which we call weights. The inputs are multiplied to the corresponding weights and their sum is sent as input to the next layer's neurons.
Each of these , is associated with a numerical value called the bias which is then added to the input sum. Together they will be passed through a threshold function called activation function and the result determines the degree of activation of this specific neuron. In this manner, informations are propagated forward from through the layers until the output layer where the neuron with the highest value determines the output. The values are basically a probability.
Along with the input, the NN has the output fed to it. The predicted output is compared against the actual output to realize the error which's magnitude indicates how far is the system from being right. That is done via the loss function, thanks to which the NN performs backward propagation where it suggests adjustments in weights.

Concretely, each of the hidden layers in meant to learn an aspect, a pattern, a certain type of information that would make it possible for the neural network, all assembled, to piece together and construct the whole image.

2.3.2.7 Convolutional Neural Networks (CNNs)

Convolutional neural network (CNN) is an artificial neural network that so far is been highly used in image analysis, and other data analysis or classification problems.
CNN has a type of specialization for being able to detect patterns and make sense of them.

- **Structure**

The specialty of CNN comes from the fact that it's hidden layers are called convolutional layers. There are eventually other non-convolutional layers within a CNN but Conv layers are what characterizes it. Just like any other layer, a convolutional layer (Conv layer) receives input then transforms it in some way then outputs the transformed input to the next layer using a convolutional operation making the layers capable of detecting patterns.

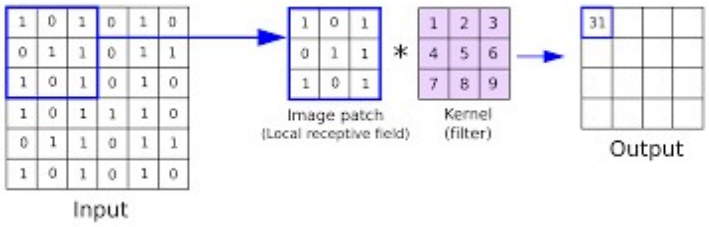

Figure 2.5: Convolution Operation explained

For each given set of data the convolutional layer shifts windows across it where it computes the Frobenius inner product of the conv kernel coefficients and the input region at every point where the window overlaps the input feature map.
In order to assure that the kernel scans each element of the the entire dataset equally we use padding which consists in adding an extension vector to the matrix of input.

With each conv layer, a number of filters is specified. Each filter exists to detect a type of pattern or information. Eventually, the deeper the network goes, the more sophisticated these filters would be.
With conv layer we usually find other types of layers such as pooling layers that serve to minimize the spacial size of the layer and to extract dominating features. A pooling operation consists in sliding a window over the data and retrieve for the next layer either the maximum value (Max Pooling) or the average value (Average pooling).

- **Performance**

CNN have shown outstanding performance with image and speech processing. A major portion of the recognition systems are built on CNNs. Later in the document, we will prove that CNNs have been classified on the top performant algorithms in radiofrequency

fingerprinting as well, but with compromises in computational efforts.

2.4 Conclusion

As much as the integration of machine and deep learning have brought privilege to the RFF problems, it also have imposed challenges. The features selection is one major challenge that have a direct influence on the performance of the classifier. Neverthless, machine learning algorithms are also another major key.These two aspects and others were studied in this chapter to open for the state of the art developed in next one.

Chapter 3

State of the art and Support works

3.1 Introduction

In this chapter , the application of the previously cited techniques is categorized under , feature and algorithm labels. In the first section, we present related works that contain approaches which serve our purpose. The rest of chapter contains a review of the supported works focusing on the developed algorithms and their performances.

3.2 Related Work

This section contains the catalogue of radiofrequency fingerprinting related work with respect to the features and the classification methods.

3.2.1 Feature-classified related work

A first classification of the state of the art is presented in this part, with respect to the used features and their according performance.

3.2.1.1 Transient features-based works

As mentioned earlier, several works have been based on features extracted from the transient part of the signal. For example, multi-resolution wavelet analysis was utilized in [8] to characterize the features contained in the transient followed by the use of a genetic algorithm to extract the wavelet coefficients that represent critical features of the transient. In [1], Choe et al developed an automated, fast signal classification and identification method using Daubechies wavelet-based feature extraction combined with an artificial neural network (ANN).

In more recent work (2001), Ellis and Serinken in [9], have shown interest to amplitude and phase obtained from complex envelope recordings to fingerprint 28 VHF radios. The same features (amplitude and phase) were deployed in other RF fingerprinting works such as Tekbas and Serinken has developed in [10] and [11], where they proved that environmental changes such as ambient temperature, additive channel noise, and battery voltage cause the feature vectors to spread over the feature space, resulting in classification performance degradation. An efficient way to solve this problem, according to the paper, is to use data collected in several environmental conditions during training. It is to note that the fact that features were extracted during variant conditions made them outperform the state of the art then.
A more diversified work proposed by Hall et al in [12] used not only amplitude and phase, but also in-phase, quadrature-phase, power and DWT coefficients to identify Bluetooth devices. The combination of all these features resulted in 8% of error rate for testing on devices from different models and different manufacturers.
In [13] Rasmussen and Capkun also performed a feature-combination technique using the transient length, the amplitude variance, the number of peaks of the carrier signal, the difference between the normalized mean and the normalized maximum value of the transient power, the first DWT coefficient, and other features, to fingerprint 10 UHF sensor devices from the same model and manufacturer.
A better performance have been noted for Ureten and Serinken's work [14] using an amplitude envelope and performing an error rate of 2%.

The performance of these features remains dependent on the precision of the transient extraction.

3.2.1.2 Steady-state features-based works

For steady-state approaches, most researchers tend to combine a variety of features in one classifier.
For instance, Brick et al [15] have implemented a multi-feature classifier considering the frequency error, the synchronized correlation, I/Q origin offset, magnitude error, and phase error, and tested it on a high-end vector signal analyzer.
The point of this work is to design, implement, and evaluate the PARADIS technique, explained in the paper, which aims to identify the source network interface card (NIC) of an IEEE 802.11 frame, and the obtained accuracy was around 99%. PARADIS has also inspired other researchers such as Shi and Jensen who used a very similar technique with radiometric features to identify Multiple Input Multiple Output devices.

As for other type of devices, such as RFID devices, modulation-based features are more frequently used. Modulation shape and spectral features were used to identify RFID transponders and have shown an error rate between 2.43 and 4.38%. In [16] frequency domain was the origin of feature extraction to identify USRP transmitters performing 97% of accuracy at 30 dB SNR.

3.2.1.3 Other types of features

A variety of these features aren't classifiable in the first two categories as theorized earlier in this document, yet have been integrated in works where some of them reached new records, especially in recent research. Indeed, in works like [17], a portion of the features used to classify devices was extracted from the transient part of the signal using variance trajectory of instantaneous amplitude and instantaneous phase responses. Yet, these features were combined with other power spectral density fingerprints and spectral correlation for classification and achieved 80% of accuracy with SNR greater than 6dB. The same performance has been reached in [18] with SNR greater than 8 dB using wavelet domain (WD) based on dual-tree complex wavelet transform (DT-CWT).
For RFID identification purposes, atypical features have been adopted recently. The use of minimum power response characteristic has performed 94.4% of accuracy in [19]. In [20] timing, modulation shape have been proved efficient in identifying devices from different manufacturers, contrary to the spectral features of device response signals that can achieve higher accuracy with devices from the same manufacturer. Clock skews also have been used in access points (APs) identification for local area networks [21] and more complex networks [22].
In GSM devices area, a developing interest appears in exploring signal parts of GSM devices. Features like the near-transient part of GSM-GMSK burst signals proved to be a promising base for classification.
Needless to say, research work won't stop in terms of finding new features for physical-layer identification. Even though the existing ones offer satisfying performances, an infinity of feature combination possibilities is always out there to prove efficiency on different algorithms.

3.2.2 Algorithms-classified related work

Although several unsupervised methods have been introduced for radio frequency fingerprinting, we don't note many unusual records of accuracy or error rate. Infinite Hidden Markov Random Field-based classification has been proposed in [23]. This work inte-

grates variational incremental inference, micro-clustering techniques, and batch updates with both time-independent features and time-dependent features in order to automatically detect the dynamic variation in the devices' number. An infinite Gaussian Mixture Model was also developed in [24]. In [25] we find a combination of K-means and PCA to design an efficient model for device discrimination of spoofing/rogue devices in ZigBee networks.

Adversely, supervised methods seem to find bloom in physical-layer identification, especially with the hurry that researchers have into integrating deep learning in RFF problems after proving a phenomenal efficiency in image and speech recognition.

K-NN have been integrated in several works such us [25] as a first use of a discriminatory classifier on steady-state-spectral features and achieving 97% accuracy at 30 dB SNR and 66% accuracy at O dB SNR based on eight identical universal software radio peripherals (USRP) transmitters.
In PARADIS [15] we find a combination of K-NN and SVM for the classifier, performed on five features cited earlier in this document.
The use in RFF isn't an emerging technology , it has indeed existed since the invention of the concept. Yet it's having a major interest these years due to the development of deep learning tools , and especially pattern tracking ones.
The use of Probabilistic Neural Network (PNN) goes back to 2004 in [10]. It was also the base of the classifier in [14] where it performed 2% of error rate on the given dataset.
Artificial Neural Network (ANN) was used by Choe et al in [1] and by Radhakrishnan et al in [26] where they introduce GTID, a technique that can actively and passively fingerprint wireless devices and their types using wire-side observations in a local network.
Recently, in 2016 , O'Shea and Corgan in [27] have introduced CNN and deep learning in Physical-layer security problem by performing modulation recognition using IQ samples and convolutional neural networks, and therefore identify the modulation type of a certain device.
To our best knowledge, CNN wasn't introduced in device identification until 2018 in [28] an initiating work of ORACLE which was introduced in the first time in [29] and changed of architecture in [30]. The second section introduces both works of ORACLE and the initiating work which were the inspiration of this project.

3.3 Support works

3.3.1 Software Defined Radio (SDR)

A new level of control over the entire processing chain is now possible thanks to advancements in software-defined radio (SDR) technology, which also allows the modification of each functional block and the sampling of changes in the input waveform. Doubtless, such an advance would open more signal exploring possibilities, facilitate feature extraction, and thus brings attention to relatively new classification algorithms for RFF purposes.

3.3.1.1 Principle, interest and history

Radio waves are electromagnetic radiations that operate in the Radio Frequency (RF) band. A region of the electromagnetic spectrum known as RF is typically found between 3 kHz and 300 GHz. All kinds of radio communications use it and its manipulation has always been a complex task, which makes researchers tend to trade complexity in devices hardware for complexity in software.

The fundamental idea behind software-defined radio is to convert as many components of a traditional radio's pipeline to digital as possible. By doing this, it is possible to perform signal processing tasks that previously required specialist gear using computer's processor.
It started wish the idea of replacing certain analog stages in the real-time signal processing tasks, specifically the demodulation by simply changing the mode of the radio. The idea of Software Defined Radio was entirely shaped in 1995 and it had even more success with the continuous appearance of high-performance computing.
The evolution of software radios have resulted in low cost modern software defined radios that mark performance computing records. These devices are now present in the forms of dongles (RTL-SDR dongles) frequently used in academic purposes due to their low cost, more highly capable integrated radios that are often FPGA-based (Ettus N210..), or progressed and very-performance systems (Ettus N310..) which can handle Giga samples per second.

3.3.1.2 Performance

Certainly the difference of cost is explained by the performance difference and characteristics in how they accept clocks in dynamic range for example, in the production of their spurious signals ..etc
These systems established the open-source aspect of software radios, which make the software i.e the information structure behind them the crucial part.
This software is what makes SDRs autonomous by enabling transceiver hardware cross-layer programmability via high-level directives. The fact that radios are becoming 'cognitive' is based on the SDR concept, in which the radio can gather contextual information and adapt its own operation by changing the settings on the SDR based on what it perceives in its surroundings.

3.3.2 Deep Learning Convolutional Neural Networks for Radio Identification

To our best knowledge , the use of convolutional neural network for device identification haven't been performed before. It's in this optic that this prior work of ORACLE finds its basis where it proposes a strategy for distinguishing a specific radio from presumably similar devices combining SDR sensing and machine learning (ML) methods.
In this section , we present the main ideas of this work, the choice of features and model's architecture and its performance.

3.3.2.1 Interest for deep learning

Deep learning provides an effective framework for supervised learning. It can learn increasingly complex functions, leverages large datasets, and greatly expands the number of layers, in addition to neurons within a layer. Thus, any device fingerprinting approach used in the field of IoT security must be computationally simple for general use. For this reason, deep learning algorithms, specifically the Convolutional Neural Network (CNN), are proposed and demonstrated through an experiment in radio identification performance for a set of devices.
Owing to the hudge success of deep learning methods in advanced classification tasks in image and speech recognition, this article suggests an algorithm that conforms the performance of CNN to device identication problem without going through any intermidiate feature processing tasks.
As we noted before, a CNN method has been introduced before for physical-layer finger-

printing purpose, however, it focused only on modulation type recognition. This approach aims to identify directly a transmitter device via the impairments presented by its I/Q imbalance.

3.3.2.2 In-phase and Quadrature-phase imbalance (I/Q imbalance)

In order to understand the interest of I/Q imbalance, a brief explanation of the causes and the feature selection of different hardware impairments is essential.

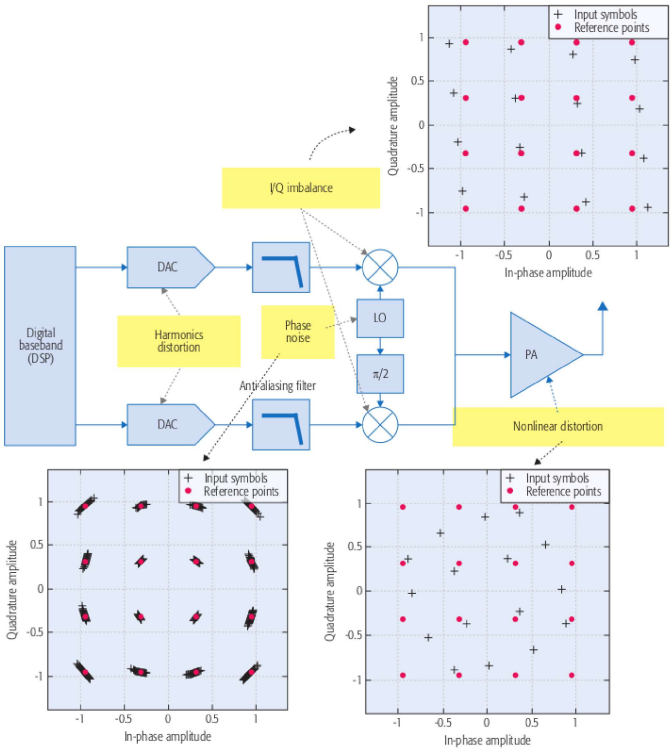

Figure 3.1: Typical transceiver chain presenting several RF impairments sources

The manufacturing process of devices usually results in imperfections that cause differences from one device to another in the physical-layer stage of digital communication. These imperfections are present within the transmitter signals and they might be caused by various parts of the transmission chain blocks as shown in the figure 3.1.
Some of these imperfections come as distortions which actually means a change in the waveform of the signal, specifically clipping them. In the figure we can observe two types of distortions ; i.e Harmonic Distortion and Nonlinear Distortion.
As we can see in the Digital to Analog Converter (DAC) level, non-linearity happen to exist the transmitter-side of DACs which results in 'Harmonic Distortions' and therefore the existence of 'Odd' and 'Even' Harmonics. Total harmonic distortion is calculated as

a ratio of the sum of the powers of all harmonic components to the power of the fundamental frequency of the signal.
As we advance on the scheme, in the Local oscillator specifically, another type of impairment could be present. In fact, the transmitter performs up-conversion of a baseband signal to a carrier frequency f_c by mixing the baseband signal with the carrier signal. Instead of producing a pure tone at frequency f_c i.e., $e^{j2\pi f_c t}$, the produced tone is $e^{j2\pi f_c t+g(t)}$, where $g(t)$ is random phase noise. Hence, a rotational jitter is caused by the phase noise. The typical phase noise level is in the [-100, -48] dBc/Hz range, with a frequency offset of [20, 200] Hz.

Another type of impairment that is easily interpretable and which happen in the entrance of the local oscillator is the In-phase and Quadrature-phase imbalance (I/Q imbalance) . The I/Q mixer is a part of the QAM modulation (or demodulation) chain so the I/Q imbalance might have an effect on the receiver as well as in the transmitter. Mathematically, the I branch represents the real part of the complex received signal, and the Q the imaginary part. In the ideal case, the phase shift is by 90 , which makes the I and Q orthogonal. However, in reality, the phase shift isn't equal to 90 which results in the deformation of the constellation diagram. Accordingly, quadrature mixers that convert baseband to RF and vice versa are frequently hampered by gain and phase mismatches between the parallel sections of the RF chain dealing with the in-phase (I) and quadrature (Q) signal paths. The analog gain is never the same for each signal path, and the difference in amplitude causes amplitude imbalance.
The complex nature of I/Q imbalance measurements makes it a potential feature for to be used in neural networks , and especially with CNN.

3.3.2.3 Data generation

This paper opts for the I/Q samples to extract the feature maps. These samples are collected from USRP SDRs that are reconfigurable as transmitters or receivers according to the purpose of experiment. Using MATLAB WLAN toolbox, random physical layer frames defined by IEEE.802.11ac standard are generated then emitted by USRP transmitters. The data is transmitted and sampled at the receiver side at 1.92 MS/s at 2.45 GHz for WiFi.
These collected samples are divided into sub-sequences then fed to the algorithm. The figure 3.2 provided in the paper, resumes the data collection setup.

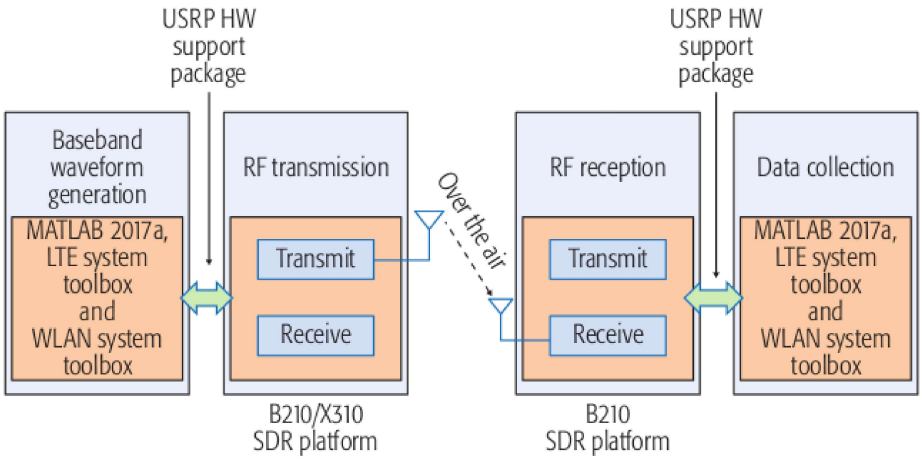

Figure 3.2: SetUp for data collection

3.3.2.4 Model's architecture

The paper treats the problem of machine learning on radiofrequency fingerprinting in its entity where it tests other algorithms such as linear SVM, and logistic regression. In this work we focus on the CNN model, and the accompanied machine learning techniques that might boost the algorithm's accuracy.

The proposed architecture is relatively simple, where it defines two convolutional blocks

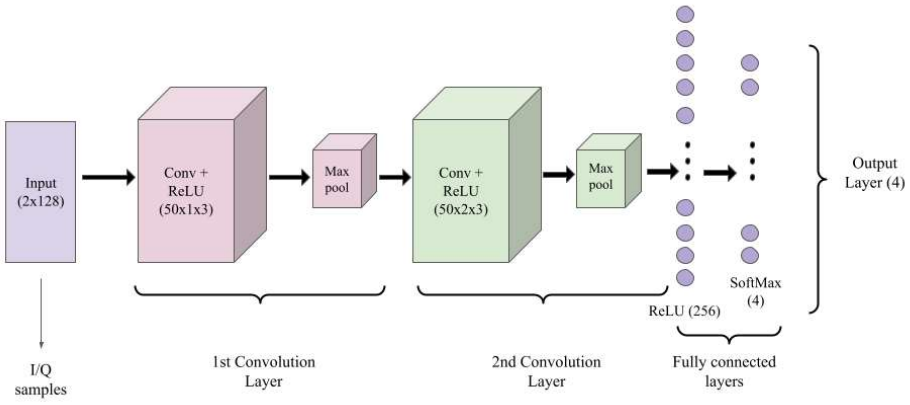

Figure 3.3: Proposed CNN architecture for RF fingerprinting

mainly for feature extraction purposes and two fully connected layers that lead to an output layer containing a number of neurons that defines the number of devices in question. Each convolutional layer incorporates a set of filters for convolutional operations that

are will perform pattern detection and thus, producing an according feature map. The first two-dimensional convolutional layer contains 50 filters of size (1x3) followed by an activation step that applies a predetermined nonlinear transformation to each element of the feature map . ReLU function is used for activation here performing an operation where it returns the maximum value between the input and 0, hence setting all negative values to 0. A max pooling layer of (2x2) is then introduced to introduce shift invariance and to save the main information present within the feature maps, all while reducing the dimensionality for computation simplicity.
The second block is quite similar, with slight differences in the filters' size (2x3 instead of 1x3) . A set of fully connected layer is then present where every neuron from a layer is connected to all the other neurons from the previous layer. The hyper-parameters are fixed as following : learning rate to 0.0001 , dropout rate to 25 percent for the first layer and 50 percent for the second and 128 for the input sequence length. The interest behind the choice of these parameters will be explained later in the 4th chapter as we try varying each of them and observe its results. The collected data is stored in a complex format then fed to the neural network as I/Q sequences to be classified. The real and the imaginary parts are treated as independent inputs stacked in a 2xl vector, where l is the length of the sequence.

3.3.2.5 Model Training and performance

The model was implemented in Keras running on TensorFlow, both introduced later in the implementation of our work, then trained using a sliding window approach in order to enhance the shift-invariance character of the algorithm, and evaluated using cross-validation.
Definitely the CNN algorithm achieved a remarkably higher performance than the SVM and logistic regression algorithms and succeeded to achieve 98% accuracy for 5 devices.Notably, the results also show that this accuracy degrades when the distance between the transmitter and the receiver is greater than 34ft, which makes the CNN algorithm distance-sensitive, but fluctuation-proof since the variance of SNR levels between 10 and 40 dB haven't had a notable effect on the accuracy.

3.3.3 Optimized Radio clAssification through Convolutional neuraL nEtworks (ORACLE)

Based on the previous work, ORACLE, an approach for detecting a unique radio from a large pool of bit-similar device, using raw I/Q samples, aims to learn and intenitionally

modify some of these features on the transmitter to boost the classifier's efficiency. The first key difference here, compared to the previous work is in the dataset. In fact, ORACLE makes an interesting observation which consists in the fact that low-end devices (such as Ettus N210 USRP) are more robust to channel variation contrary to high-end devices (such as X310 USRP) which are more channel variation-sensitive due to their manufacturing process.
ORACLE proposes a quite complete project compiled on two sub-works published on a conference and an extended version. Both of them will be presented separately .
In general, in each article, a CNN classifier architecture is introduced and evaluated. Based on the evaluation, the interpretation and on the nature of dataset, a pre-deployment method is proposed where certain modifications on the dataset are performed to enhance the performance of the classifier .
Though this report looks at the problem from a pure machine-learning angle, a brief description of these methods will be included in the following review of the papers.
The observation concluded from these methods will help us 'correct' the classifier's performance, which was treated from a hardware/signal-processing perspective using deep learning techniques included in the classifier itself in order to boost its performance.

3.3.3.1 Coference work

This paper proposes a deep CNN classifier, test it on several datasets taken under different conditions and 'sculpt' the dataset to enhance the performance of the classifier.

- **Deep CNN Architecture**

The figure 3.4 introduces ORACLE's CNN classifier which is similar to the one of the previous work but with slight differences in layers' level. µIt's also inspired from the architecture of AlexNet [31] , a deep CNN architecture specifically designed to classify 1.2 million high-resolution images available in the ImageNet dataset into 1000 different classes. It is to note that AlexNet is considered one of the most influential papers published in computer vision been cited more than 80000 times and having spurred many more papers published employing CNNs and GPUs to accelerate deep learning.

In this work, the size of the filters have been increased from 1x3 and 2x3 to 1x7 and 2x7, and an other fully connected layer was added. These modifications will be interpreted later in this section. The input data is always a matrix of 2x128 where values of I and Q are two distinct channels to be passed to the spatial filters of the CNN. The first convolutional layer is meant to extract features related to the variation over the channels

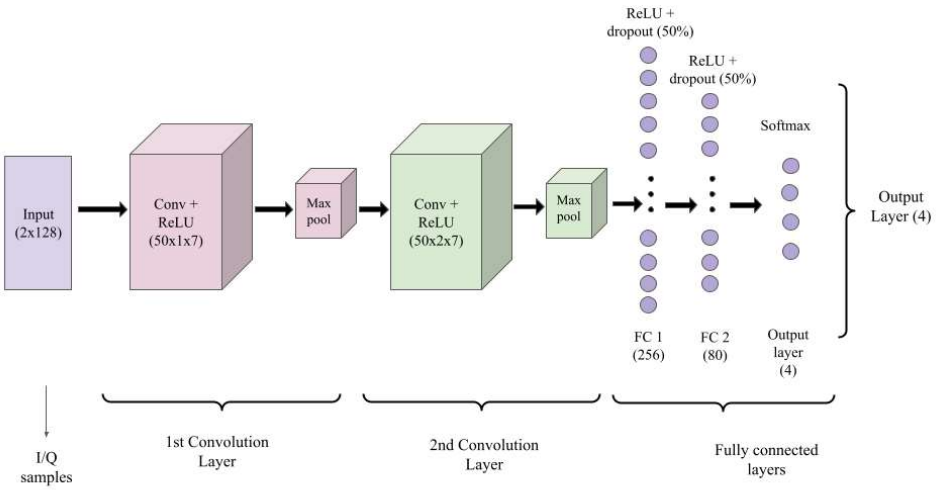

Figure 3.4: Deep CNN architecture proposed in ORACLE (conference paper)

I and Q separately. 50 distinct maps are then generated over the complete input sample and fed to the second layer which's meant to learn variations over the two channels (I and Q) combined. A ReLU activation is attached to each of the layers then applies a predefined non-linear transformation to the output.
A second fully connected layer is introduced compared to the prior architecture , where 80 neurons are placed before the final SoftMax classifier.
A regularization parameter of 0.0001 and a dropout rate of 50% are added to avoid overfitting .

- **Training and performance**

For an authentic performance evaluation, the model was trained on several datasets. A first training was performed on static channel transmission using different types of device (not high-end or bit-similar devices, for exp : phones/tablets/laptops/drones..etc). During these experiments, the transmission channel isn't changed (a.k.a cable transmission). The model is trained on data (I/Q samples) collected of 140 devices from 122 manufacturers. For testing, 16 random devices from the training set are selected and the results were as following: 99% of accuracy for 16 devices out of a training set of 100 devices and the accuracy drops to 96% when the training set is elevated to 140 devices.
A second training was performed on 16 high-end X310 USRP SDRs as transmitters and the same B210 radio as a receiver, always keeping the static channel conditions. The model achieved 98.6% of accuracy.
Experience also have shown, that even though the channel is static, different environments and locations have major effect on the accuracy and make it drop from 98% to 87% for

the same testing dataset.

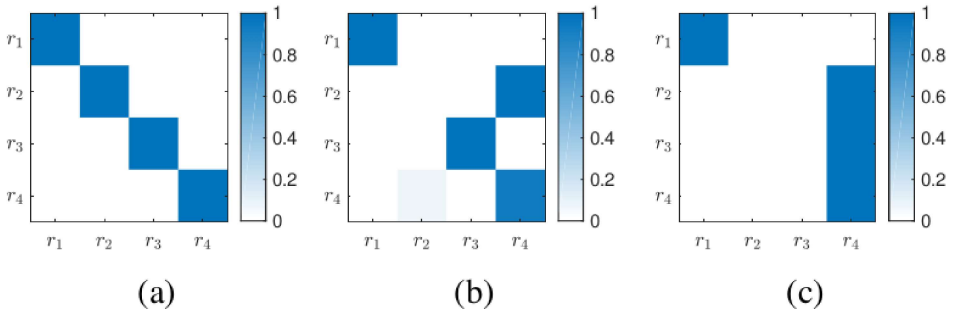

Figure 3.5: Model's classification accuracy for 4 devices testes on (a) time t1 and location l1 (b) time t2 and same location l1 (c) time t3 and different location l3)

The second part of the evaluation stands on using a dynamic channel-transmission dataset , which basically, is collected with the same setup of high-end X310 USRP SDRs and B210, but with the difference of sending data over the air. It is here that the classification accuracy drops critically when the channel is changed. The figure above summarizes the results of the testing on 4 devices where it shows that changing the channel between the train and the test results in a much lower accuracy. Also if the transmitters experience similarity in the wireless channel , it is more likely for the model to miss-classify the devices.

- **Re-statement of the problem**

The authors concluded that the nature of high resolution for the high-end USRP SDRs makes them more channel-sensitive in the sense that :

- Multi-path reflection and fading have a significant impact on received IQ samples, that the distortion makes the classifier unable to classify devices.
- In the cases of miss-classification, the similarities between two certain channels dominate the subtle hardware imperfections and therefor push the model to miss-learn the features.

ORACLE localizes the problem in major modification that the I/Q samples undergo due to wireless channel effect and proposes a method where in a pre-deployment phase the receiver provide feedback to the transmitter to incorporate controlled impairments and

then used demodulated symbols in the classification process rather than raw I/Q samples. This technique aims to make the classifier robust to channel variation in the sense that it makes the transmitter hardware dominate the channel.

- **Feedback approach for performance enhancement in wireless channels**

A detailed study is presented in the paper, where the impact of these intentionally introduced impairments on the demodulated symbols is fully analysed and how a given combination of impairments would result in a repeatability in the classification's outcome. Once the hypothesis has been confirmed, the solution consists in selecting feasible impairments that produce remarkably dissimilar I/Q samples, that satisfy the prior conditions while reducing the BER influence for the transmitter. The classifier is, first, the pre-trained on a virtual setup (implemented on GNU radio), in order to make it learn the artificial impairments pattern which is proven both device and channel agnostic. Then, demodulated data is collected over-the-cable after introducing impairments. For data augmentation purposes, a Gaussian noise is introduced all while maintain the power of noise between the original and the generated data below -13dB. Next, the model is trained on the generated data.

- **Performance using demodulated symbols**

Demodulated data is collected over-the-cable after introducing impairments. For data augmentation purposes, a Gaussian noise is introduced all while maintain the power of noise between the original and the generated data below -13dB. Next, the model is trained on the generated data. The classifier performs 99.76% of accuracy .
Once the model have learned from cabled data, it is again evaluated on demodulated data collected wirelessly (with introduction of the same impairments). It achieved 99.5% of accuracy which demonstrates that the unique patterns created by the impairments can still be detected even in the presence of random noise.
Yet , when tested on data transmitted from the same devices but without introducing any artificial impairments, the classifiers accuracy degrades to 35.96%.

- **Interpretations**

Although, the accuracy obtained by ORACLE's classifier is significantly important compared to the state of the art , especially with high-end devices , which were very little used in RF fingerprinting problems, it presents slight deficiencies given its dependence for prior hardware-related stages, which reduces the applicability of this method in real-

life/applicable environments.
The change that were done on the architecture had considerable effect on the accuracy and the problem re-statement. For instance, the increase in the size of kernels from the very common size '3' which is highly used for image processing problems, to 5 and 7, have a major effect on augmenting the number of parameters and therefore allowing a more expressive power and more informations to be learned, but increases the risk of over-fitting on the other hand. Also , adding another fully connected layer have definitely served to extract higher levels of non-linear combinations of the feature maps.

3.3.3.2 Extended work

• Deep CNN architecture

The architecture, shown in the figure 3.6 , is also inspired from AlexNet. The authors definitely opted for a deeper architecture based on the theory of deepening the network for a more accurate result.

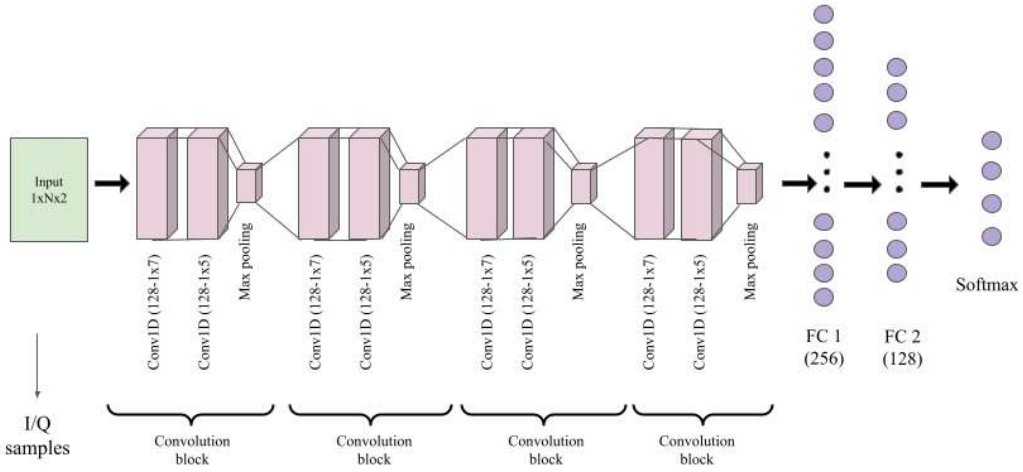

Figure 3.6: Proposed deep CNN architecture in ORACLE 2

Unlike the previous architecture, this one includes totally 15 layers including 'convolution blocks' instead of convolution layers. As shown in the figure, each block counts 2 convolution layers and a max pooling layer at the end. The output of the 4 stacked block is then fed to the fully connected layers, in which we can observe that the second fully

connected layer has incremented to 128 neurons instead of 80 in the previous paper.
It is also to mention that in this architecture only 1D convolution layers are used while introducing two sizes of filters. The number of filters have also been increased to 128 for each layer.

- **Training and performance**

This model have been trained with the same hyper-parameters of the first, while specifying a batch-size of 1024.
Similarly to the previous work, this one have also been tested first under static channel conditions and have indeed achieved very identical results. The accuracy for testbed I/Q samples in a static channel is around 98.6% . Yet have shown major vulnerabilities in over the air transmission, where it showed the same poor classification accuracy for 4 devices of high-end USRP devices and a maximum of 46% for external dataset of wireless transmission .

- **Under-complete demodulation and its performance**

A first attempt to boost the classifier's accuracy was by using under-complete demodulation. It is all about removing the effect of the channel without channel estimation or equalization which will help with deleting the effect of the channel without compensating the device's inherited imperfections.
That could be done following the steps : first estimating and compensating the carrier and sampling the frequency offsets, then estimating the channel thanks to pilot training sequence and finally reapplying the previously calculated frequency offsets.The output of this step is demodulated symbols to be injected in the CNN classifier. The performance of this under-complete demodulation approach was evaluated on external dataset and shows a maximum of accuracy of 85% for 50 devices.

- **IHOP : approach and performance**

Still aware of the fact that channel effects should be reduced in order to enhance the classifer's accuracy, in this paper ORACLE ows to the previous feedback-driven demodulation approach's (approach presented in [29])) vulnerabilities. In fact, although the previous method had considerable success in the boosting ORACLE's identification accuracy as it satisfies scalability, accuracy and reduction of the communication impact, yet it still presents other scalability and security issues.
The previously proposed algorithm suggests that each transmitter device is assigned a

unique impairment in order to minimize the total BER experienced by all radios in the network. However, if we do so, the number of radios to be identified is limited by the number of feasible impairments which won't be efficient since, due to the BER constraint of the radios, the number of impairments does not scale up , thus preventing the use of ORACLE for a large number of radios.

The second limitation is security-related as eavesdropping or brute-forcing techniques can be used by an adversary radio to learn the unique impairment used by a legitimate radio. It can then simulate the characteristics of the real device simply by introducing the learned impairment during its own transmission, which makes this approach vulnerable to spoofing attacks.

To solve these problems, ORACLE replaces the old technique with the Impairment HOPping spread spectrum 'IHOP' approach which is inspired from the Frequency Hopping Spread Spectrum (FHSS). A detailed theory behind this method is presented in the paper , it hovers around identifying devices through an artificially introduced random peudo-noise sequence, which will consists in the identification key of the device. The receiver shares the coefficients of this key over a secure channel with each transmitter. Thanks to that binary sequence, a set of pairwise impairments are selected and applied on the transmitter. Once a signal is emitted by the transmitter, the reciever demodulates it and a sequence of demodulated symbols is produced. The figure 3.7 summarizes the whole process.

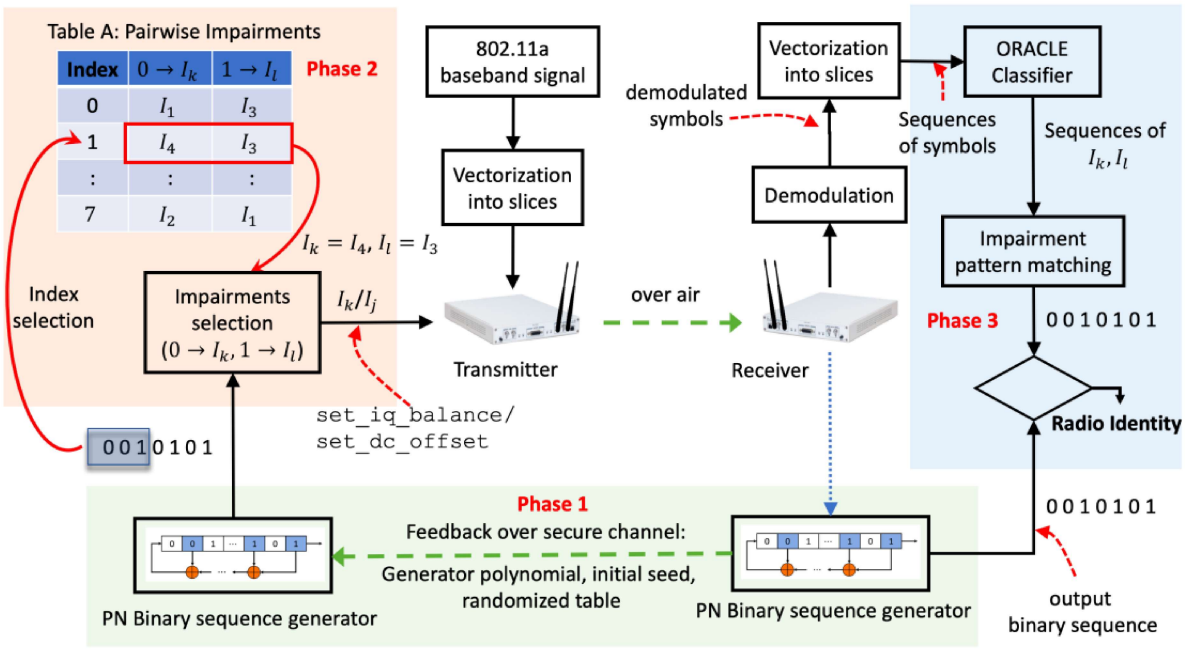

Figure 3.7: Illustration of different phases of radio devices using IHOP method

The use of IHOP has definitely led to considerably better results.
The first evaluation consisted in training the CNN with data transmitted over the cable using IHOP method (replacing the over air transmission by wired transmission), and it gives an accuracy of 99.76% on 16 X310 devices. For wireless channel , data was collected in two different environments, one typical indoor and another with less reflections , and the accuracy was around 99.5% for both. A final experience consisted in testing the classifier on raw I/Q samples , and it showed severely minor accuracy, around 35%.

- **Interpretations**

Proposing a new architecture didn't perform a major improvement on the identification accuracy for raw I/Q samples collected over-the-air, yet it showed near perfect results with IHOP method, which highlights the model's architecture.
Despite the actual trend of opting for simpler and less deep neural network for deep learning problems, which is mainly motivated by the continuous seeking for less computation powers, it is evident that for CNN-based models perform much better when they're deeper. And that is definitely explicable given the nature of these neural networks that is based on learning from each added layer.
The main issue with shallow networks is that they are very good at memorization, but not so good at generalization. The advantage of multiple layers is that they can learn features at various levels of abstraction which makes deep CNNs much better at generalizing because they learn all the intermediate features. As a proof, outstandingly performant models can reach 1000 layers (ResNet).
Forasmuch, adopting a deeper architecture would definitely serve for the best , especially that we are dealing with a huge amount of data. Furthermore, we can observe that convolution layers are stacked one before the other with no Max-pooling in between, unlike the old architecture where each convolution layer was followed by a max pooling operation. This architecture opts for stacking two convolution layers 'vertically' one behind the other to minimize the number of parameters while retaining the main information, which is known to improve the model's performance in terms of training time (number of parameter) and learning convergence. According to [32], we can see as shown in the figure 3.8 that one layer of (5x5) convolutions is equivalent to 2 stacked layers of (3x3) convolutions kernels in terms of number of layers .

The picture below shows also that accuracy of two convolution layers (3x3) stacked is higher than the one obtained for the same dataset with only one conv layer of (5x5).
This is mathematically explainable since in the first case (one conv layer (5x5)), the

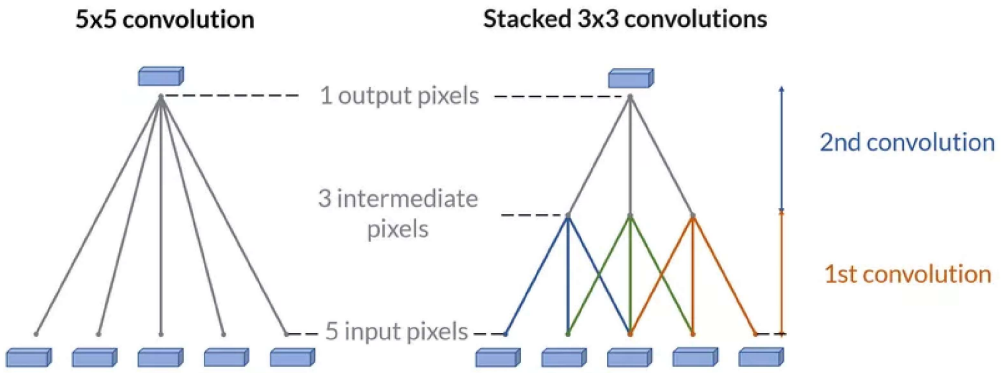

Figure 3.8: Convolution of 5x5 layers vs two 3x3 layers

activation of the final neuron is a function of 5 variables , $f(x1, x2, x3, x4, x5)$, while for the second case (2 conv layers of (3x3)) the output neuron is a function of only 3 variables $f(x1, x2, x3)$ where each of the variables is a function of only 3 variables $f'(x'1, x'2, x'3)$, therefore the result in the final layer is a composition of the two function (fof' or $f(f'(x'1, x'2, x'3))$).
This composition operation is obviously at the origin of the conservation of relevant information, unlike the case of one layer of large kernels where a lot of side informations are taken in consideration .

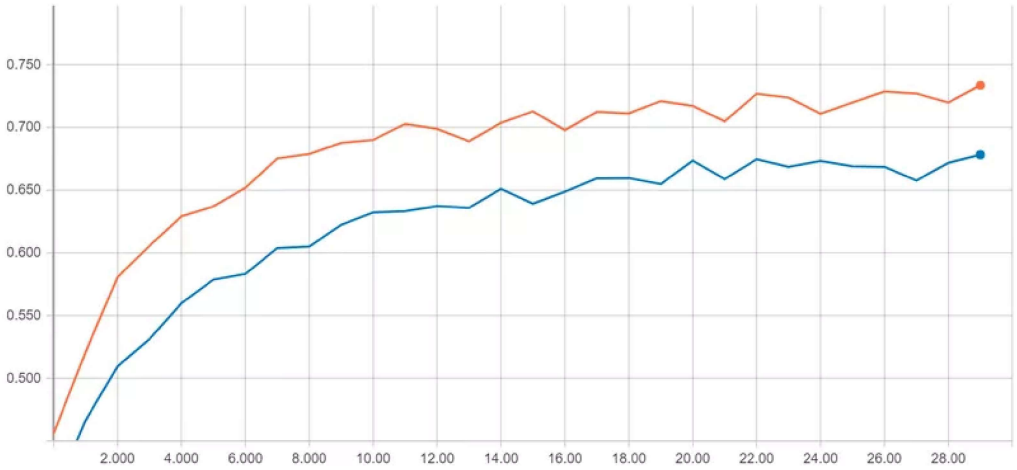

Figure 3.9: Performance of convolution of 5x5 layers (blue) vs two 3x3 layers (orange)

The increase in the number of neurons in the last-but-one layer is also potentially coming from the same perspective. In fact, the number of neurons is half the number of neurons of the first FC , which makes and average of two input neurons from the first FC result in one neuron in the second FC. This choice might then be motivated by the need to conserve more information before the output layer.

3.4 Conclusion

This chapter presented related work and ORACLE, the first CNN to be applied for radiofrequency fingerprinting classification problem. Being a promising work especially with the fact that it's improvable with the continuous development in areas of ML/DL , we adopt ORACLE's approach to build our own classifier trying to achieve better results on over-the-air transmitted data.

Chapter 4

Implementation and development

4.1 Introduction

This chapter presents the implementation process of the classifier. We study details regarding the nature of the signal, their characteristics and their effect on the performance of the model. We get then to the development details citing tools, approaches to improve the performance of the model on wireless transmission. We finally present our results and suggestions to enhance the classifier's performance under different circumstances.

4.2 Problem statement

As mentioned before, the main limitations of ORACLE classifier's performance , go back to the transmission channel's effects, where noise and other phenomenons interfere and change the constellation of I/Q imbalance which results in miss-classification or non-recognition of a certain device.
In order to have a better understanding of the details of the problem , a prior knowledge of the nature of signals and standards is needed.

4.2.1 Signals of interest

In this work, the datasets are generated using transmitters that are bit-similar USRP X310 radios that emit IEEE 802.11a standards-compliant frames generated via a MATLAB WLAN System toolbox. That being said we are interested in OFDM signals.

4.2.1.1 FDM

To know how OFDM works we need to first briefly present FDM or Frequency Devising Multiplexing. It's a multiplexing method used to divide the transmission channel into many non-overlapping sub-channels which will , therefore allow multiple users to share one single link.

4.2.1.2 OFDM

Orthogonal Frequency Devising Multiplexing (OFDM) is variety of FDM that is highly popular in wireless and telecommunication standards such as WiFi 802.11ac , 4G, 5G, cellular phone technologies..etc
As FDM divides available bandwidth into different non-overlapping sub-channels, it keeps a guard-band which is a narrow frequency range, inserted between adjacent sub-channels so that different signals travel separately and simultaneously without interfering with each other.
In OFDM, (as shown in the figure 4.1 provided in [33]) these sub-channels are closely spaced so that not only, there's no guard band between them, but they are also overlapped.
This being said, is is obvious that OFDM allows more data transmission within a certain available bandwidth than FDM.
To solve the interference issue, OFDM combines the signals closely together all being orthogonal to each other *i.e* they act independently. Any neighbor signals in OFDM operate without dependence or interference with one another as when a signal reaches its peak, its neighbors land at 0 thanks to their orthogonal nature.
The signals are multiplexed in a way that a peak of one occurs only at null of all the other neighbor signals.

At the receiving end, the demultiplexer would separate them based on the orthogonal feature.
OFDM uses, therefore, wisely the available bandwidth and offers higher data transmission rate than FDM.

4.2.1.3 IEEE 802.11

It defines the collection of media access control (MAC) and physical layer (PHY) protocols for implementing wireless local area network (WLAN) computer communication.

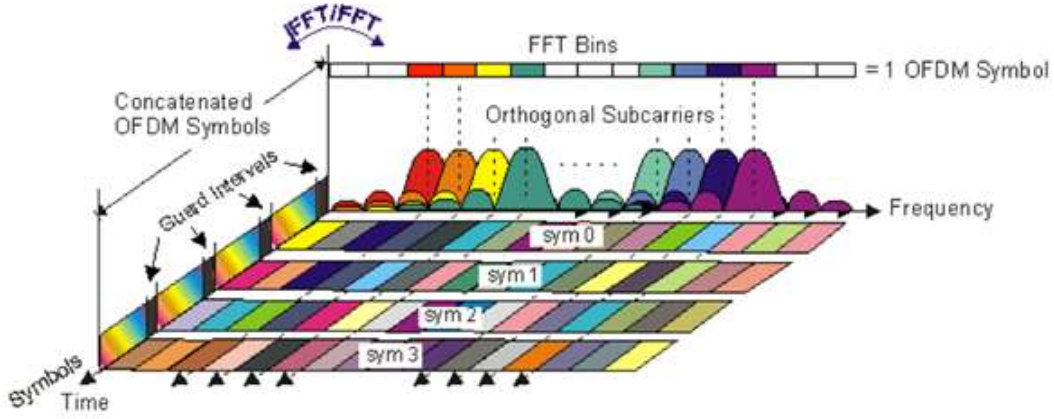

Figure 4.1: OFDM transmission

IEEE 802.11 is a subset of the IEEE 802 set of local area network (LAN) technical standards. The standard and amendments are the most commonly used wireless computer networking standards in the world and serve as the foundation for wireless network products bearing the Wi-Fi brand. The majority of home and workplace networks employ IEEE 802.11 to enable wireless communication and Internet access between computers, printers, smartphones, and other devices.
The figure 4.2 illustrates the structure of an OFDM data frame citing different features contained in different parts of the signal.

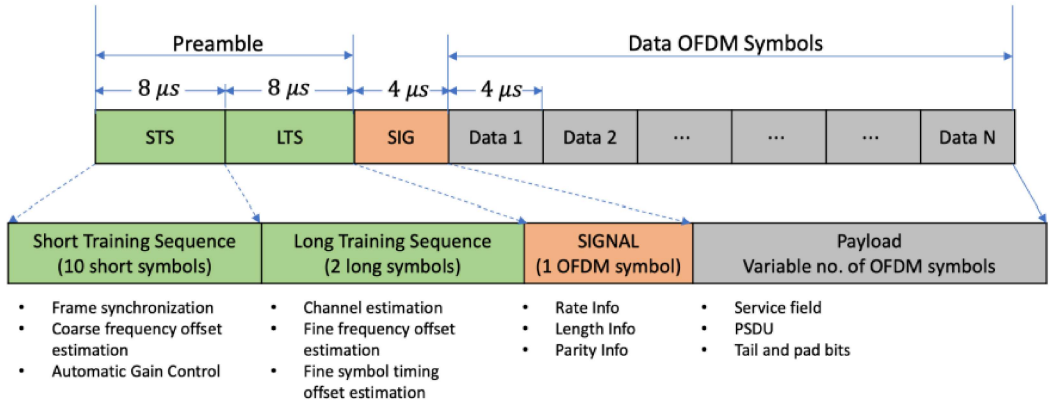

Figure 4.2: Structure of an IEEE 802.11a OFDM frame

4.2.2 Multi-path effect on wireless transmission

It has been shown in several previous works (ORACLE's conference paper included) that multi-path reflection and interference is a major reason behind the weaknesses of the

classifiers when dealing with transmitter signal features. This phenomenon, results in eventual deformations in the signal's features and generates a certain divergence in the classification process.

4.2.2.1 Basis

Complex baseband is composed from and in-phase and a quadrature-phase components. Let's note the transmitted passband signal as p(t). If we dispose of an idealized single path channel model where no objects of reflections are present, the received signal is :

$$r(t) = h_0 p(t - \tau_0) \qquad (4.1)$$

Received signal is attenuated by a factor h_0 and delayed by some time τ_0. h_0 depends on propagation medium, transmitted frequency, separation between BS and MS, Tx/Rx gains... As for the delay, it is mainly dependant on the speed of the EM wave in propagation, and the separation between BS and MS.
Nevertheless, actual wireless transmission environments are nothing like the idealized model. Objects of reflection exist in several positions which results in reflected/scattered paths. As shown in the figure, Each of these paths, have its specific attenuation and delay and the received signal is then the superposition of multiple paths.

$$r(t) = \sum_{n=0}^{m} h_n p(t - \tau_n) \qquad (4.2)$$

where m is the number of paths generated due to the presence of objects.
Furthermore, the previous channel characteristics h_n and τ_n, can be time variant due to mobility, hence introducing time to these characteristics: $h_n^{(t)}$ and $\tau_n^{(t)}$.
These new characteristics have eventual effect on the transmission quality.

4.2.2.2 Effect of multi-path fading in wireless transmission

Multi-path fading is one of the major significant problem in wireless communications due to various obstacles like building, scattering, reflection. The previous time variant and invariant characteristics often results in multi-path channel fading.
A channel can be characterized, as mentioned before, by a number of paths, and the dependence of the channel coefficients/delay on time. Depending on the number of paths,

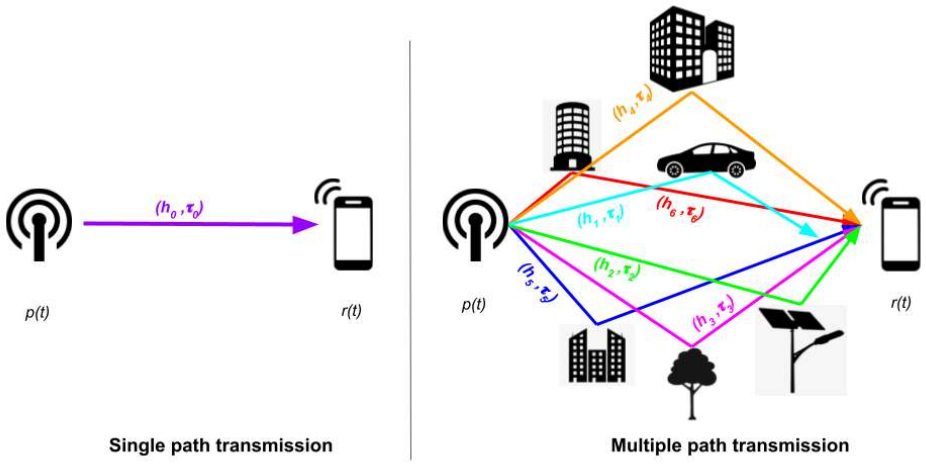

Figure 4.3: Modifications introduced on the emitted signal in case of one path channel (left) vs multi-path channel (right)

two main types of fading can exist: Frequency Flat Fading and Frequency Selective Fading.

Moreover, time dependence, can result either in fast or slow fading channels (*i.e* depends on the coherence time). These phenomenons tend to affect transmitted signals. This effect might be manifested as a :

- Variations in signal strengths over small distances or short time intervals.
- Shifting in the carrier frequency due to device motion.
- Echoes caused by variable delays of multi-path components.

One way to modelize a multi-path channel is by '*Rayleigh fading channel*'. It is to mention, that the Rayleigh distribution has been selected by the IEEE 802.11 working group to assess the performance of the modulation used in IEEE 802.11a and IEEE 802.11b wireless multi-path environments.

In this modelization, each multi-path channel component is represented by a couple of values; a single tap and its associated delay. In Rayleigh distribution each single tap is modelled as following :

$$\alpha_k = A + jB \qquad (4.3)$$

where A and B are zero mean independent, identically distributed Gaussian random variables with variance σ^2. A detailed mathematical representation of the Rayleigh fading channel is presented in [34] which concludes that the received signal r(t) and the transmitted signal p(t) are related as following :

$$r(t) = p(t) * h(t, \tau) + n(t) \tag{4.4}$$

With :

$$h(t, \tau) = \sum_{k=1}^{m} \alpha_k(t)\sigma(t - \tau_k T_s) \tag{4.5}$$

Where τ_k is the delay of the k^{th} path normalized by T_s, and n(t) complex Additive White Gaussian Noise.

There are several other representation such as *Rician fading* which consists in the same physical and mathematical basis as in Rayleigh fading but with the addition of one dominant path that is directly in the line of sight (transmitter,receiver).

Needless to say, all these models aim to construct a compromise basis between a realistic representation and a mathematical one, in order to overcome the challenges imposed by wireless transmission in multipath environments. In [35] we find a wide study on the effect of multipath on RF fingerprinting and some proposed solutions such as channel estimation ..etc.

4.2.3 Impulsive noise : usrp-host interface throughput

It is expected that raw collected data might contain a lot of non-significant information that might potentially cause the divergence of the model. The causes of such data noise are several. For instance , during data collection, USRP devices are highly likely to capture some impulsive noise that will result in high peaks within data samples, quite regularly. In fact, sustainable data throughput is required between the receiver and host PC. Although, the nowadays platforms (such as Ettus X310) offer high accuracy ranging and positioning, yet the risk of eventual data offloading is high as well during signal acquisition .
In [36] several potential peaks generation are presented during signal acquisition.
We also note that the presence of non-meaningful peaks might be in forms of relatively long samples, *i.e* some data sequences do not contain any relevant information, rather only carrier signal. This being said, data filtering is a necessity to avoid any risk of unimportant data.

4.3 Proposed techniques

In this section, we describe techniques that we believe might potentially enhance the performance of ORACLE's classifier based on the previous analysis of the potential problems. Some of the techniques cited below are inspired from signal processing problems which we raised in the previous section. Yet the proposed solutions are purely algorithmic and do not require any hardware or software intervention during data acquisition.
The other techniques are purely machine learning and CNN related methods that are known to enhance the performance of deep CNNs in general , and 1D convolutions specifically.

4.3.1 Data normalization

As we mentioned earlier in the section where we raised the issue of the multipath effect, this latter have major consequences on the amplitude of the signal and therefore on the values of captured I/Q samples.
One way to solve this without prior set inteference is normalization with respect to the maximum value as this method practically deletes the effect of multipath on the amplitude.
Normalization is a traditional machine learning technique for data preparation. Normalizing inputs not only reduces the risk that the neural network learns non-significant features, but also helps remarkably with the training speed.
Concretely, this is performed as the cost function is directly affected by the learned features. So , if one feature ranges from two values x1 to x1' , while the other feature ranges between x2 and x2' with x1<<x2 and $x1' >> x2'$, the cost function is more likely to be elongated on a 2D plan which will lead to longer step for the gradient decent to reach the minimum. Whereas, if features were normalized, which means that the values are close enough, the cost function would rather be circular/symmetric which makes the gradient descent finds its way to the minimum faster wherever it starts.
Data on similar/close scales is easier to process and to learn from, yet this step needs to be performed after filtering peaks (next subsection) to achieve higher results.
In our case we normalized I and Q values with respect to the maximum value of the recording file. Below are the constellations of I/Q samples before and after normalization.

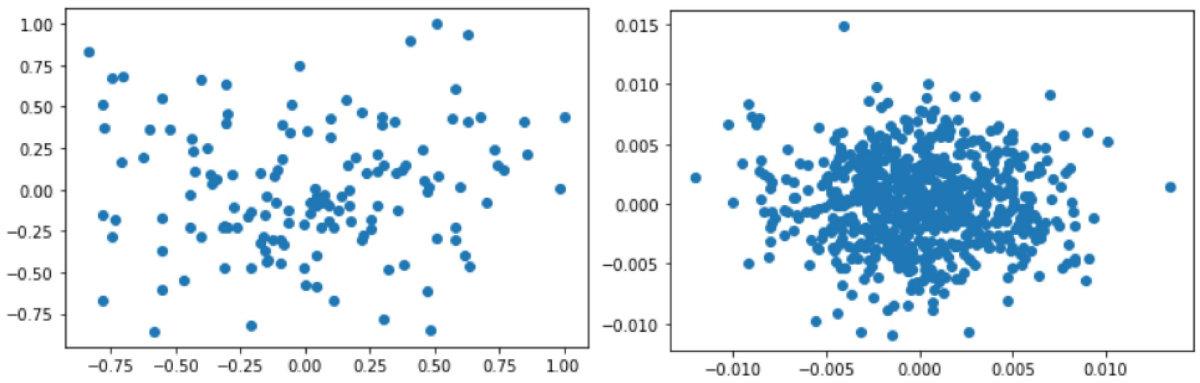

Figure 4.4: Constellation of I/Q samples acquired from a x310 USRP device over-the-air before normalization (left) and after normalization (right)

Since the values of our data are originally ranging between -1 and 1, scaling them didn't have a major influence on the range of values as much as it did on the shape of constellation. In fact, normalization with respect to the maximum values of the record made the constellation narrower and more concentrated around 0, unlike the first constellation where mini clouds of dots where scattered (mainly because of the gain and phase shift due to multipath effect) .
In later section we will demonstrate how normalization helped increase accuracy since it played a data cleaning role before classifying.

4.3.2 Filtering peaks

In order to solve the issues of non-significant peaks of data, we suggest to perform a peaks-filtering operation before normalizing data.
Generally, in signal processing signal filtering is a crucial pre-processing step aiming to smooth out high-frequency noise associated with measurements as we mentioned before. In our case , plotting the I/Q samples, have shown eventual appearence of high values that, depending on the record, can be more or less frequent.
That being said, we suggest to detect a peak with respect to its neighbors. To do that, we resort to scipy-signal library of python (introduced later in the implementation and developement section).
This library offers a pre-defined function allowing to detect peaks specifying a minimum height to be considered a peak, and threshold defining a vertical distance between the peak in question and its neighbors (an alternative of calculating the prominence which is originally a topographic term also used in signal processing, and it means the height of a certain peak relatively to its contour lines. Calculating prominence is shown to be

computationally heavy , hence the use of threshold parameter).

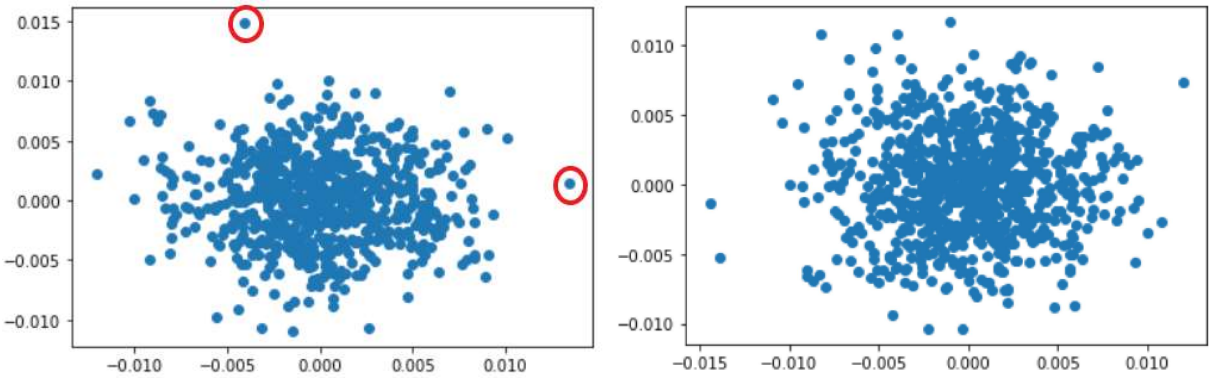

Figure 4.5: Constellation of normalized I/Q samples acquired from a x310 USRP device over-the-air before filtering peaks (left) and after filtering peaks (right)

As a primary visualization attempt of the impact of peaks filtering , we tried performing the operation on the same data file that has been already normalized.
The constellation of the figure above show the disappearance of constellation points that are relatively distant/discarded from the main cloud of dots.
Yet, for a real enhancement, filtering should be performed on raw data before normalization. Doing so, have boosted the performance of the classifier since it helped us provide only significant data to be learned from.

4.3.3 Batch-normalization

As we explained earlier in the data normalization section, large and non scaled data points can cause instability within the neural network, mainly because relatively large inputs can cascade down through the layers in the network which may cause imbalance gradients hence the problem of exploding gradients (a problem we will be tackling later).

However, even after normalizing data, the weights being updated with each epoch during training via Stochastic Gradient Descent, some of the weight might possibly be drastically larger than the other weights which will result for the corresponding neuron's output to be extremely large. This imbalance continues to cascade in the neural network causing previously invoked instability.

Batch Normalization comes in handy as it is applied, by choice, to certain layers to first normalize the output from the activation function using the mean and the standard deviation, then multiply that normalized output by an arbitrary parameter and finally adds the it to an other arbitrary parameter. This operation sets a new standard deviation and mean for the data. The mean, the standard deviation and the two arbitrary parameters are all trainable, *i.e* they too will be optimized during the training process.

Doing so, the weights within the network don't become imbalance with extreme (high or low) values since the normalization is included in the gradient process.
Adding batch norm to our model have indeed increased the training speed and likely reduced the ability of outlying large weights that might over influence the learning process.

Unlike pre-processing data normalization which occurs before feeding data to the input layer , batch normalization serves rather to normalize output data from the activation functions from the individual layers , on a per batch basis.

In our model we used Keras to implement batch normalization by specifying the axis , the initializers and other optional parameters as well.

4.3.4 Regularization

Regularization refers to a set of techniques for preventing over-fitting in NNs and thus helping the model generalize more, hence , improving its accuracy when confronted with completely new data from the problem domain.
Over-fitting is the phenomenon in which a neural network learns the training data extremely well but fails to detect a pattern that is generalized on unseen data. It is caused by noise in the training data, which the neural network detects and learns as an essential feature of the data during training.
Yet, noise is unique to each training samples, thus is an unreliable element to consider and makes the model incapable of recognizing crucial information in new data.
If the model is of a higher complexity, learning noise will be highly likely since , it picks up even random details due to slight fluctuation or error.
Regularization comes in handy in such problems. Two techniques are highly used in this area; L2 and L1 regularization.

4.3.4.1 L2 Regularization

Also known as 'Ride Regression' or 'Weight Decay', this type of regularization is the most used one. It consists in adding a term to our loss function that penalizes for large weights. This term is actually the sum of the squared norms of the weight matrices multiplied by a very small constant. Let:

$$\sum_{j=1}^{n} ||w^{[j]}||^2 * \frac{\lambda}{2m} \tag{4.6}$$

Where n is the number of layers, $w^{[j]}$ is the weight matrix for the j^th layer, m the number of inputs and λ is the regularization parameter. Norms are obviously used to convert the weights' values into positive values while conserving their characteristics. The previously elaborated term is added to the loss function in order to penalize for large weights.
If λ is significant, then the term $\frac{\lambda}{2m}$ is also as significant. As long as our weights are large,if we multiply the previous term by the sum of the squared norms, the product might also be huge. This indicates that in order to reduce loss, our model is motivated to choose small weights that maintain a low value for the whole function. This type of regularization is called regularization for simplicity since it forces weights toward zero but it does not make them exactly zero.

4.3.4.2 L1 Regularization

Alternatively called regularization for sparsity, it is used to manage sparse vectors. Those are primarily composed of zeroes, as the name implies. Sparse vectors typically result in very high-dimensional feature vector space. As a result, handling the model becomes exceedingly challenging. By deducting a tiny amount from the weight at each iteration and eventually reaching zero, L1 regularization causes the weights of uninformative features to be zero. The used term in this type of regularization is the sum of the absolute values of the weight parameters in a weight matrix as shown in the following equation.

$$\Omega(W) = ||W||_1 = \sum_i \sum_j |w_{ij}| \tag{4.7}$$

This term is then multiplied by $\frac{\lambda}{m}$ and added to the lost function.

As explained earlier, either L1 or L2 regularization perform operations that reduce the complexity of the model, which reduces the risk of modeling the data noise and therefor

avoids over-fitting. Yet, the performance of these operations remains dependant of the aware choice of the regularization parameter allowing to balance model's complexity and accuracy.

4.3.4.3 Dropout

Dropout is a very popular regularization technique. It consists simply in eliminating the effect of some neurons of a certain layer (turning them off). The regularization parameter here is concretely a probability. For example, if the regularization parameter is equal to 0.5, and we dispose of layer of 100 neurons, the random 50 neurons will be turned off during the training. The dropout rate is defined to each layer and is mostly used in fully connected layers.

4.4 Implementation and development

In this section we describe the implementation process of the CNN-based identification algorithm. We start by introducing ORACLE's dataset which we used to test the performance of the model and which allowed us to visualize the effect of the introduced techniques.

4.4.1 Datasets

To build our classifier we used ORACLE's published dataset which is available in Genesys Lab's website. In this section, a detailed description of the dataset, the experimental setup and the format are introduced.

4.4.1.1 Dataset description

- ***ORACLE's dataset***

The main purpose of our work is to use deep learning for detecting unique transmitter signatures contained in the emitted signal. In Genesys Lab's website two datasets are available:

1. **Raw In-phase and Quadrature-phase samples (I/Q samples) :** consists in IQ samples collected wirelessly (over-the-air) without any prior intervention.
2. **Demodulated symbols :** consists in I/Q samples that are recovered without change in the channel of transmission (cable transmission).

16 high-end bit-similar USRP X310 are used as transmitters, and a the same USRP B210 is used as a receiver.

To have a versatile dataset and in order to make the classifier generalize the learned feature, several conditions are taken into consideration.

In fact, for each transmitter around 20160000 I/Q samples are collected and stored in binary files. From file to file , the distance between the transmitter and the receiver varies (from 2ft to 62ft) , as well as the environment of the experiment as shown in the figure 4.6.

For each transmitter, 2 runs (records) are performed for a fixed transmitter-receiver distance , in the locations shown in the figure where more or less objects of reflections are introduced.

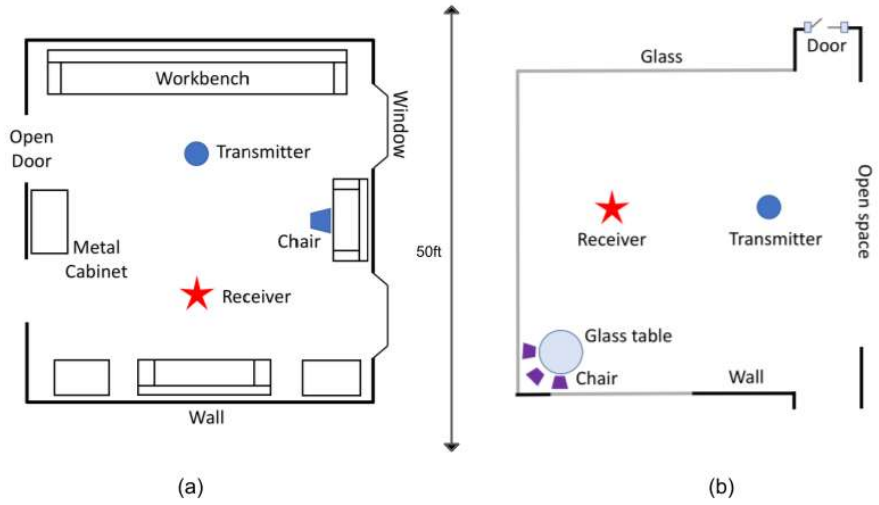

Figure 4.6: Two different experiment locations (a) closed lab area, (b) more open area with much less reflections

The receiver SDR samples the incoming signals at 5 MS/s sampling rate at the center frequency of 2.45 GHz for Wi-Fi.

- ***SCEE's testbed dataset***

For preliminary evaluation, we tested the performance of the classifier using SCEE's testbed dataset . This dataset consists in I/Q samples collected over the cable using

USRP X310 as transmitters and USRP B210 as receiver.
At the receiver end the sampling rate is fixed to MP/S.

Both datasets (SCEE testbed's and ORACLE's) are stored in SigMF files, a format that we will introduce in the next section.

4.4.1.2 SigMF format

SigMF is a specification for storing digital recordings of sampled data and the metadata that describes it.

- ***Interest***

The idea behind SigMF format is originally motivated by the need of sharing sets of recorded signals as it is a key element in the advance of research work. Before the development of SigMF, datasets of signals weren't easily portable and were mostly lost which delimits the development of research signal-related works to those who have access to equipments allowing them to reproduce their own datasets.

SigMF defines a standard way to describe signal recording by saving both the data and the recording details and therefore enables it to be safely portable and exploitable by multiple users.
The SigMF python library has been launched in 2017 by the start-up DeepSig, have been update two times after the launching and is still under development.

- ***Specifications***

Datasets and their related metadata (ralted capture information) are the two main categories of information that the SigMF definition covers.
The datasets are mainly collections of digital measurements representing any source of information that changes over time, temperature readings from a thermal sensor, geolocation coordinates from a GNSS receiver, or any other digital measurement data that has been saved. In our case for example, SigMF datasets contain In-phase and Quadrature-phase samples.
As for metadata, it describes informations about the recording experiments intended for both human users and computer application, to help get a better understanding of it and a potential reproduction of the dataset. This description contains details such us the title, capture specifities, signal frames' details and other types of informations that will be detailed later in this document.

- ***SigMF files***

A SigMF file is actually composed from two separate files , a data file (binary-compliant) with the extension 'sigmf-data' and a metadata file with the extension 'sigmf-meta'.
For data files , there are four orthogonal characteristics of sample data: complex or real, floating-point or integer, bit-width, and endianness. For example the string 'cf32_le' stands for 'complex 32-bit floating-point samples stored in little-endian'. It is also to note that data samples are stored without separation. In cases like ours where the data is complex, samples are interleaved, with the in-phase component first (i.e., I[0] Q[0] I[1] Q[1] ... I[n] Q[n]).
As for the metadata files, it's a text format containing elements as key/value pairs. The three component fields/objects of a sigmf metadata file are :

- Global : a group of key/value pairs providing general essential information to spot and understand the dataset. These informations vary from a recording to another, but mostly they concern data type, the sampling rate, the SigMF version used to store data, the author(s), the location of the recording system and other crucial informations.

- Capture: Mostly organized into segments. Each of these latters describes a chunk of samples that can be mapped into memory for processing. This contains informations regarding the the index of the beginnings of the sample in the dataset, the timestamp of the sample index , the center frequency of the signal, the header bytes ... etc.

- Annotation : This section contains informations that aren't included in the previous 2. For example for ORACLE's case, this section contains the 'environment' field, where a brief description of the experiment's environment is provided (indoor/outdoor), transmitter and receiver identification parameters, distance between the transmitter and the receiver ...etc.

Depending on the need, eventual objects can be added to the meta-file. For example for RF fingerprinting purposes objects of transmitter and receiver containing informations as the model, he serial number, the low and high frequency, the antenna..etc might be highly useful.

Indeed, for ORACLE's published dataset, three additional objects are introduced : transmitter object, receiver object, and antenna object.

- ***Data format***

This dataset was stored as 64 bit floating point instead of 32 bit, which means that, an entire I/Q samples is stored on 128 bits (64 bits for In-phase and 64 bits for Quadrature-phase). It is also to mention that the files were stored on an outdated version that required updating using JSON in order to make it python 2.9-compliant and be able to exploit it. The data files were read as binary files specifying the type of data as Complex 128.

4.4.2 Model

To implement our deep learning model we adopted ORACLE extended paper's architecture. It's to mention that no code or development/training details were provided besides the description of the architecture in the paper.
That being so, the choice of the model's hyper-parameters has been utterly a work of testing, observation and improvement.

4.4.2.1 Adopted architecture

We have inspired from the architectures described in ORACLE papers to develop our own classifier, with some changes. In this section we describe the interest for the architecture and the choice of parameters.

- ***Deeper architecture***

Our preference for the second architecture has been motivated by, first, the great classification results in transmission over-the cable with high-end USRP devices, and second, by the fact that such architecture is more suitable to be solve our RF fingerprinting problems and susceptible to be improved.

In fact, a deeper architecture with tighter kernels is more susceptible to detect more precise features as we explained earlier.
In our conception, the key element behind the potential of this architecture is the convolution blocks. Through machine learning history, stacking convolution layers in blocks followed by max pooling is decisive to have compromise between precision in feature extraction and computation decencies. This idea of blocks started with the concept of $'Bottleneck\ Block'$ and is generating outstanding results with more recent types such as $'Residualblocks'$, $'Dense\ Blocks'$ and $'Squeeze-and-Excitation\ Blocks'$ implemented in deeper architecture that have proved efficiency in image processing problems and that might potentially be a promising option for radio frequency fingerprinting.

- ***1D convolutions***

For the purpose of this work, we target features embedded within the hardware impairments that are time-invariant and identifiable equally in different moments of the signal (figure 4.2) . For this reason, 1D CNNs are a smart choice in the architecture, as they are useful for feature extraction from fixed-length segments of the datasets in cases where subtle informations are location-invariant within the signal.
1D CNNs are highly effective in 1D signal processing. They were proved adequate for problems regarding ECG signals, real-time monitoring high-power circuitry, damage detection.. etc.
A study including 90 papers [35] have shown that 1D convolutions considerably outperform 2D convolutions in 1D signal problems as they for various reasons :

- under equivalent conditions (same model configuration, network and hyper parameters) 1D convs reduce computational complexities comparing to 2D convs.
- 1D CNNs usually use more compact configurations (2 hidden layers at most) which results in lower number of parameters and lower training time all while having achieving very similar performance results as 2D CNNs.
- Due to the reduced complexity, 1D cNN are more adequate for real-time applications
- 1D CNNs have shown superior performance for applications with limited labeled data and high signal variations acquired from different sources

Be as it may, implementing a 1D CNN is a wiser choice for our work where we have signals emitted from different transmitters, the number of transmitters is fixed and limited.

4.4.2.2 Architecture-related changes

No major changes were introduced on the model initially. However, after running the first tests (whose results will be presented later) we tried varying several hyper-parameters and we have settled for only two changes in terms of neural network's architecture. The first in the input dimension, which was fixed to 128x2 and changed to 256x2.
In fact, as explained in both papers, according to the authors' tests for the model the input dimension of the neural network haven't had major effect on its performance as long as it is compatible with dimension of the first layer. However, after our tests, we found that this assumption is true when only one recording (file of I/Q samples) per device

is fed to the neural network. However when the input dimension is fixed to (256x2) better accuracy is achieved when dealing with a lot of data (from various recording per file).

The second change is in the number of neuron within the second fully connected layer that was initially set to 128. During tests and with the dropout rate that we set, we found that 64 is an optimal size for this last-but-one layer.
As for the output layer, we opted for making it 'customizable' as the number of units is user-specified and dependant of how many device's are introduced to the neural network. Therefore, the model building function takes as a parameter the number of devices.

4.4.2.3 Training

To train the model, we have set the dropout rate for the fully connected layers to 50% and use Adam Optimizer with a learning rate of 0.0001. Adam Optimizer is a an optimization algorithm as it's an extension to stochastic gradient descent. It performs optimization by calculating an exponential moving average of the gradient and the squared gradient, and controlling the movement of the parameters that are related to these averages. The number of epochs and the batch-size were fixed after several tests and we finally set to 80 epochs and 1024 for the batch-size.

4.4.2.4 Data loading and formatting

As mentioned before, our dataset is stored in SigMF files in complex format. As ORACLE's dataset was stored in an outdated version, we have opted to load the recording as binary files then process it using python libraries. For the labels we use device IDs (embedded in the files' names) as identifiers for transmitters.
The data path is schematized in the figure 4.7 describing the following steps:

- First, the data is read from sigmf-data binary files as an array of complex values in the form :

$$I1 + jQ1$$
$$I2 + jQ2$$

$$\ldots \tag{4.8}$$

$$In + jQn$$

- The I/Q samples are then partitioned to be fed to the neural network into sequences of 128 samples. We tried both non overlapping and overlapping sequences, and we had better results with overlapping sequences. The choice of the length is motivated by the study presented in ORACLE's paper where it's shown that the length of the sequence interferes more in larger sequences since in shorter periods of time, the channel maintains its state. The study attested that the length of sequences wouldn't have a considerable impact if it exceeds 128 sample per sequence.
- After partitioning the signals we generate 2 sub-sequences of each sequence , one for In-phase ([I1, I2,...,In]) and the second for quadrature-phase ([Q1,Q2,...,Qn]).
- The I and Q sequences are then filtered to eliminate non-significant peaks and normalized before being stacked together to satisfy the required input shape (2x128 or 2x256).

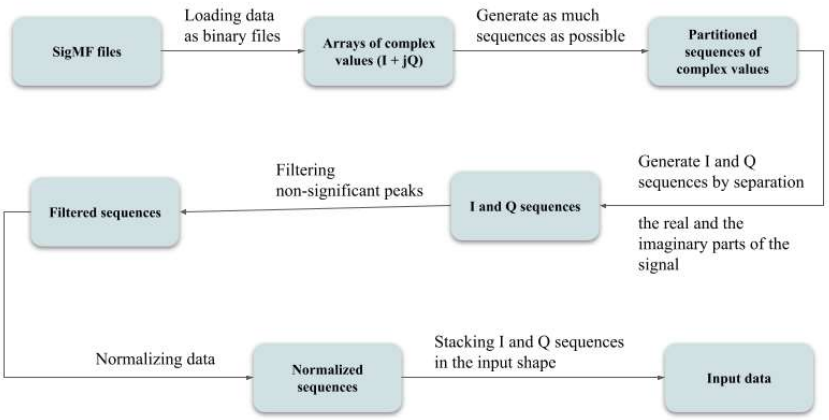

Figure 4.7: Data path from SigMF files to input shaped data

4.4.2.5 Training

To train the model, we have set the dropout rate for the fully connected layers to 50% and use Adam Optimizer with a learning rate of 0.0001. Adam Optimizer is a an optimization

algorithm as it's an extension to stochastic gradient descent. It performs optimization by calculating an exponential moving average of the gradient and the squared gradient, and controlling the movement of the parameters that are related to these averages. The number of epochs and the batch-size were fixed after several tests and we finally set to 80 epochs and 1024 for the batch-size.
We tried to find an optimal method for feeding the input I/Q samples to the neural network by testing multiple length of sequences, multiples parts of the recording and a couple of ways to partition the signals whose result are shown later.
For loss function, we tried binary cross-entropy and categorical cross-entropy which w adopted for the rest of the tests. The main difference between binary and categorical cross-entropy is in the mathematical function as binary cross-entropy is defined by :

$$p(x).log(q(x)) + (1 - p(x)).log(1 - q(x)) \qquad (4.9)$$

X being the predicted class, and p(x) being the predicted probability of the point being equal to 0 (neuron deactivated) and q(x) the predicted probability of the target being equal to 1 (neuron activated), the first term cancels when the target is 0 and the second when the target is 1. This loss function works better on multi-label classification. Categorical cross entropy is defined by :

$$-log(q(x)) \qquad (4.10)$$

Conserving the previous definition of q(x), this loss function works more for multi-class classification, i.e each example belongs to a single class.
We note that both loss functions performed very similar results , only with a slight difference in favor of categorical cross-entropy in terms of speed of training, which is why it was adopted for the training.

For the output, we assigned every device to a number defining the associated label. These labels, were encoded using ' One Hot Encoding' for better label separations. This technique encodes categorical features into one-hot numeric arrays.

4.4.3 Development tools

To develop the code, the model was build using Tensorflow and Keras. Other python libraries were essential especially for the data pre-processing stage.

4.4.3.1 TensorFlow

TensorFlow is a flexible, open source machine learning library mostly used for deep learning purposes developed and launched the google brain team in 2015. Tensorflow and its

associated higher layers (such as Keras) makes building deep learning models simple and understandable as they offer a set of updated and comprehensive tools allowing to build, train, predict, save, reload and even fine-tune DL models. TensorFlows libraries convey operations such as pre-processing, data ingestion, evaluation, visualization and serving. Its nature of being able to run on several devices and platforms makes it portable and deploy-able anywhere which have encouraged the launching of deep learning experiments in topics where it wasn't motivated before due to complexity , such as RF fingerprinting.

4.4.3.2 Keras

Keras is a higher-level API supporting multiple back-end neural network computations. It is open source high level framework of TensorFlow as it provides essential abstractions and building blocks for developing and shaping ML problems with high iteration velocity. Keras consists a reliable simplified interface that helps the implementation of commonly used NN building block layers using Theano and TensorFlow and running on CPUs and GPUs. Our model for instance is built using Sequential module that defines a neural network for which we can add customized layers (other modules) and perform training, optimization and evaluations operations.

4.4.3.3 Scipy

It's a free open-source python library used for scientific computing as it stands for Scientific Python. Along with numpy, which is more common library, they both form the fundamental python packages for numerical and scientific programming. While Numpy contains more array data and basic operations, Scipy contains rather fully-featured versions of mathematical functions. Scipy englobes a set of powerful sub-packages for various scientific computation problems.
As mentioned before, we have utilized Signal sub-package of Scipy to filter peaks via the function $'find_peaks()'$. Signal introduces useful tools for signal processing such as creating filters, creating and analysing spectrograms..etc.

4.5 Preliminary results

To validate the model primarily, it was trained and tested on demodulated symbols provided by Genesys team, i.e data transmitted over the the cable. We needed to find similar results to approve our implementation of the model. First we have plotted the constellation of the I/Q samples corresponding to data, where no channel effect is introduced.

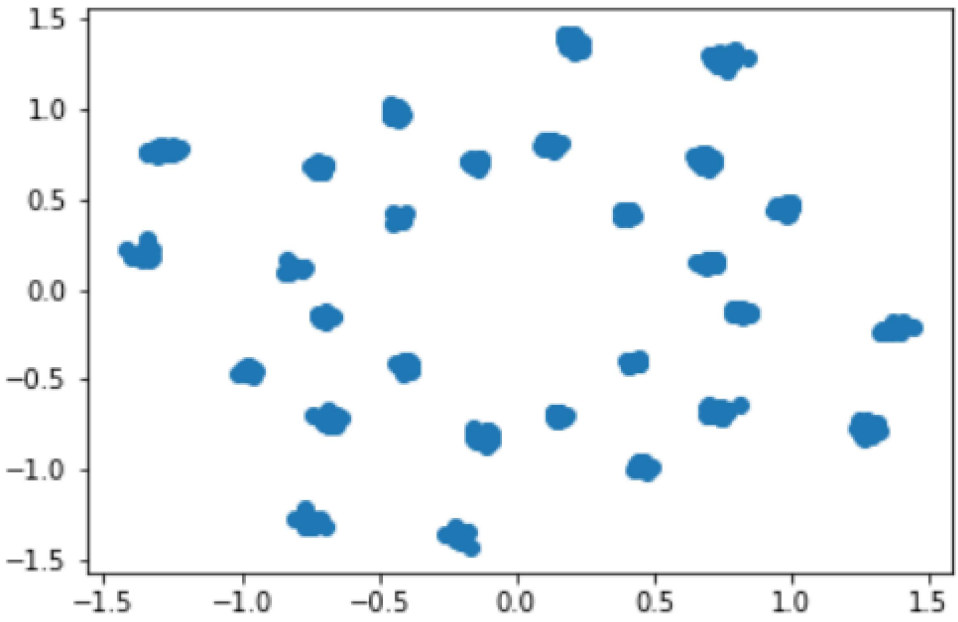

Figure 4.8: Constellation of a chunk of demodulated symbols

The difference between the constellation associated to the transmission over-the-cable and over-the-air for the same device (312D76) is obvious. The first one is more close to a perfect modulation constellation, whereas for the second, deformation has occurred on the received signals that has resulted in a more centred cloud of points.
We have trained the model using 3 files of the demodulated symbols, each corresponding to a different device. 480000 samples were used for training, 120000 for validation and 200000 for testing. The accuracy was of **97 to 99%**.
The accuracy for this dataset is plotted in the following figure 4.9. We note that the sequences of data were non-overlapping and the batch-size is set to 1024. No operations of normalization or filtering were performed either, we just fed raw data to the neural network.

Once the model was validated on ORACLE's demodulated signal, we tested it on our own dataset, where I/Q samples were transmitted over the cable as well. We had similar results , i.e **99.3%** of accuracy for the same terms. The evolution of the accuracy is presented in the following figure.

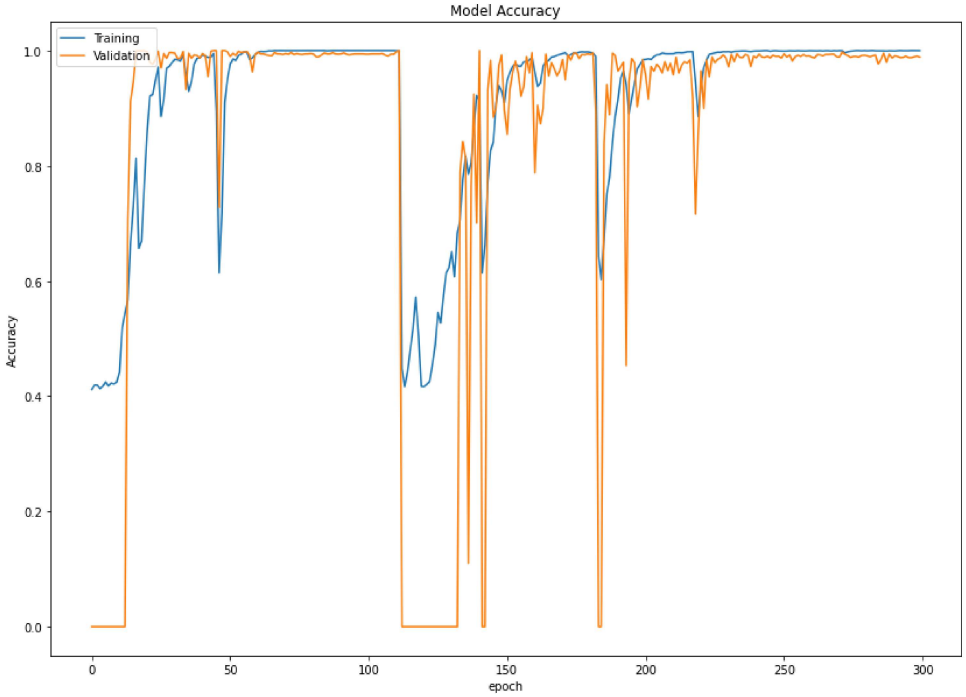

Figure 4.9: Evolution of the accuracy of the model using over-the-cable collected data of ORACLE's testbed

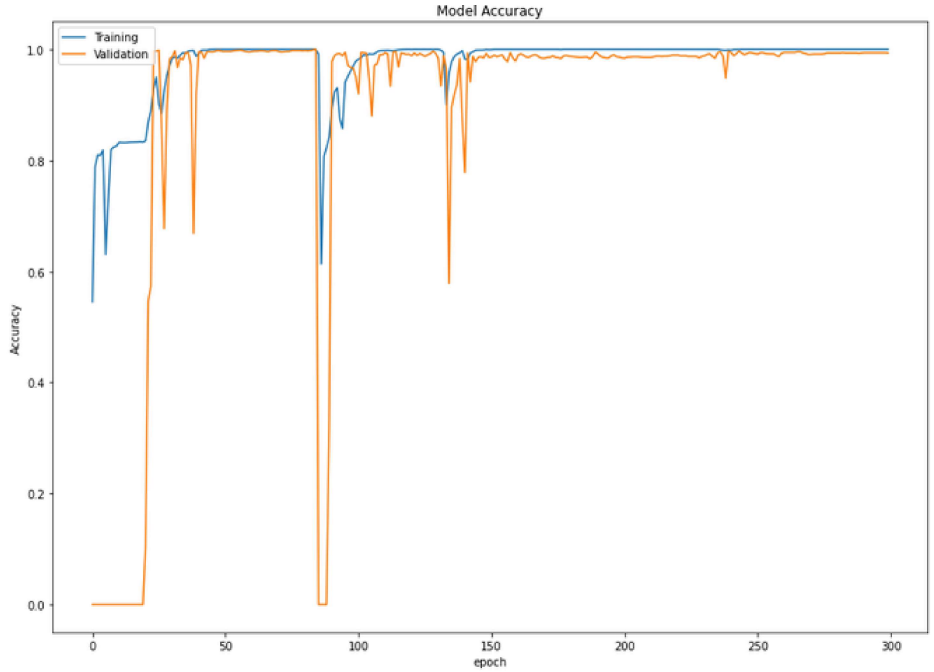

Figure 4.10: Evolution of the accuracy of the model using over-the-cable collected data of SCEE testbed

Once the performance of the classifier was approved on over-the-cable transmitted data, we conducted multiple tests using raw I/Q samples.
Being aware of the limited accuracy of the classifier using raw data, we tried varying parameters such as the batch size to find an optimal value. We also tried feeding data from different experience setups, i.e fixing a transmitter and a receiver and feed data captured in different environments and in different distances between the two.
In a second place, we tried feeding normalized data (without previous filtering) instead of raw data and we summarized the preliminary results in the table 4.1.

Table 4.1: Preliminary results on raw I/Q samples

Conditions	Accuracy
Raw data, batch size = 150, 1 file per transmitter	Training Accuracy = 0.62 Validation Accuracy = 0
Raw data, batch size = **1024** , 1 file per transmitter	Training Accuracy = **0.8** Validation Accuracy = 0
Raw Data, batch size =1024 , 3 files per transmitter	Training Accuray = 0.42 Validation Accuracy = 0
Raw Data, batch size = 1024, **8 files per transmitter**	Training Accuray = **0.83** Validation Accuracy = 0
Normalized data, batch size = 1024, 8 files per transmitter	Training Accuracy = 0.88 Validation Accuracy = **0.2**

In the table above, 'Raw data' is associated to raw I/Q samples that haven't been normalized neither filtered. Normalized data is associated to I/Q samples that have been normalized but not filtered.
The previous results show that the optimal batch size is equal to 1024. It shows also that varying the input data , i.e, feeding the network with data taken in different conditions is highly likely to help the mode learn more features and therefore improve the training accuracy. However, that doesn't help with generalizing, since the validation accuracy is set to 0. Nevertheless, normalizing data even without filtering has helped improving

slightly the validation accuracy.
With this in mind, we tried to visualize the effect of feeding overlapping sequences of data instead of non-overlapping ones to help with the shift invariance. We plotted the evolution of the accuracy in the figure 4.11.

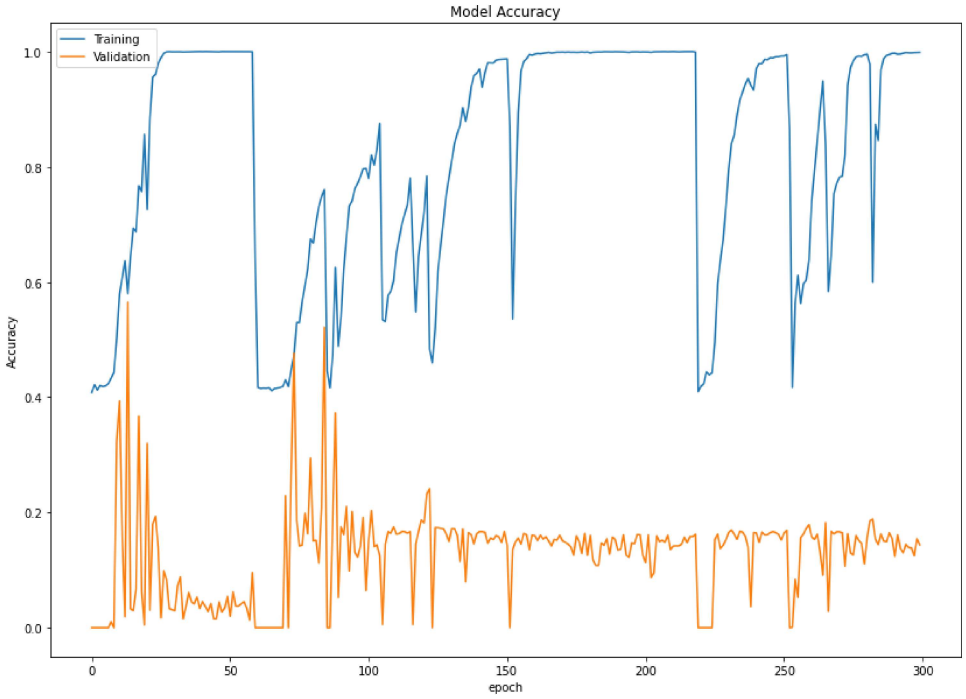

Figure 4.11: Evolution of the accuracy of the model using overlapping sequences of I/Q samples

It's obvious that trying overlapping sequences without prior filtering, haven't had a stabilizing impact on the validation accuracy, i.e the model still fails to generalize learned features, despite the relatively high training accuracy.
We suspected that non significant peaks are responsible of the 'false-learning' of the model, which led us to try filtering before normalization.

The filtering operation have definitely helped the model stabilize , the training accuracy is set to 1 and the validation have improved to 0.46 . Although it's not a satisfying result, we could approve that filtering have a major influence on the NN's performance.

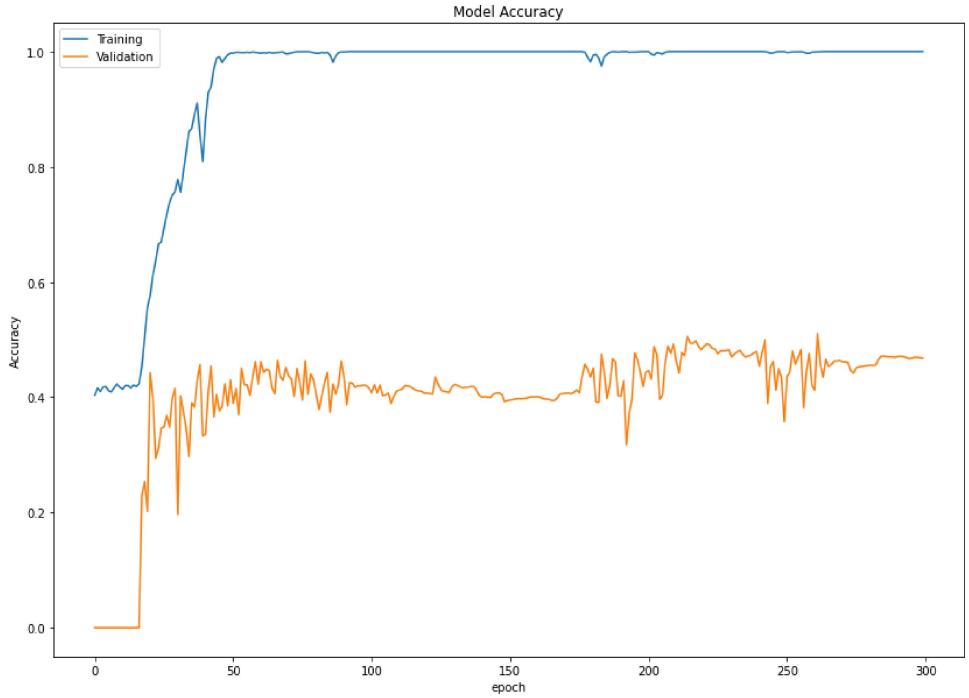

Figure 4.12: Evolution of the accuracy of the model using filtered I/Q samples

4.6 Limitations and solution

As shown in previous results, the main issue with the performance of the model is generalization. Therefore we tried to introduce machine learning techniques that might boost the generalizing character of the model. A well known one is cross-validation.

4.6.1 Cross-validation

The need for cross validation comes from the fact that training the model on the same data, excites it to eventually memorized that data which results in over-fitting problem. Cross - validation is a technique that helps a model be trained and evaluated on a portion of a database, before reapportioning the dataset and evaluating it on a new potion.
The figure simplifies the process of cross validation . The dataset is divided into smaller sets. For each iteration one set (the red one) is used for test and the others (the blue ones) are used for training. In the next iteration another set is used for test and the rest of the dataset is used for training. At the end of the training , the model is trained and evaluated on every portion of the data which makes it capable of learning features rather than learning data, and therefore generalizing on unseen data.

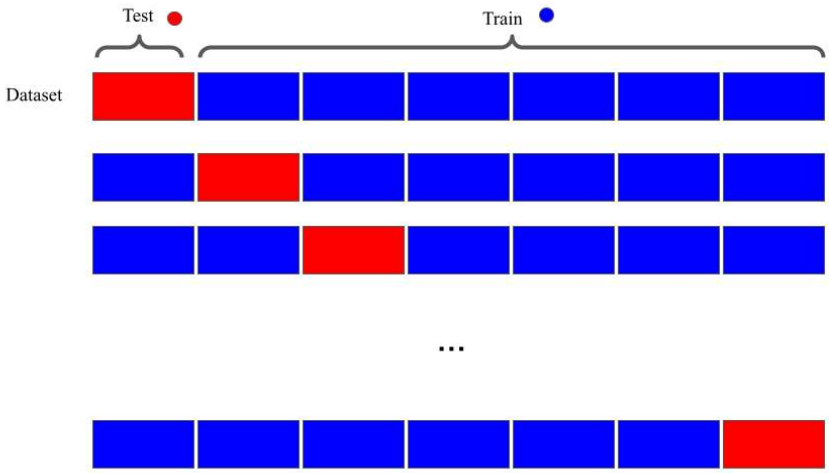

Figure 4.13: Data splitting with cross validation

To test the efficiency of this technique we chose to launch training on only two records of two different transmitter, i.e the factors of distance and setup environment are not interfering. The purpose here, is to visualize what impact has cross-validation on data collected over-the-air, filtered then normalized.
To do so we performed cross-validation using three distinct techniques varying between manual data splitting and using Keras tools.

1. Manual Cross-validation : It's about splitting data outside of the training functions into a number of folds , then retraining the model using one fold for evaluation and the others for training. Although our awareness that the model wouldn't memorize the learned patterns from previous folds (since the training function is reinitialized every time), yet the launching of this test helped us shape the idea of fine-tuning as a potential technique to enhance the performance of the model.

2. Cross-validation using Keras K-folds : Keras provides two main tools for cross-validation . Both of them splitting the dataset into folds and applying the cross-validation concept inside the training function. In K-folds approach , the number of the folds is specified using the parameter K.

3. Stratified Cross-validation using Keras StratifiedK : Very similar to the K-folds approach, StratifiedK ensures that each fold of dataset has the same proportion of observations with a given label. This type works rather when we want to compensate imbalance in the classes distributions.

The results of the three variants are summarized in the table 4.2.

Table 4.2: Results on over-the-air collected data using cross-validation on 2 devices

Method	Fold number	Training Accuracy	Validation Accuracy	Testing (prediction) Accuracy
Manual Cross-Validation	Fold 1	0.46	0.16	0.1
	Fold 2	0.41	0.2	0.5
	Fold 3	0.63	0.21	0.35
	Fold 4	0.4	0.12	0
	Fold 5	0.78	0.35	0.5
StratifiedK Validation	-	0.63	0.46	0.5
K-Folds Validation	-	**0.97**	**0.99**	**0.9996**

The results have shown that the most accurate method is K-folds cross-validation with near-perfect results on two transmitter devices with no distance or experiment set-up variations.
Once, that was confirmed, we have tested cross-validation on a higher number of devices, with variation , of the experiment setup first, i.e train the neural network on data captured in different environments (presence of a different amount of reflection), then on different distances and different experimental setups and finally , we tried to train on data captured from a certain distance and evaluate the prediction using data captured from a different distance. The results are grouped in the table 4.3 per number of device.

Table 4.3: Results on over-the-air collected data using cross-validation on more than 3 devices

Method	Number of devices	Training Accuracy	Validation Accuracy	Testing (prediction) Accuracy
Same distances different runs	3 devices	0.98	0.99	1
	4 devices	0.973	0.988	0.999
	5 devices	0.7	0.36	0.63
Different distances, different runs	3 devices	0.71	0.52	0.5
	4 devices	0.366	0.36	0.23
Train on a distance and test on another	3 devices	0.98	0.99	0.35

4.6.2 Interpretations

Despite the fact that introducing techniques such us filtering and normalization along with other machine learning methods that we introduced on the NN have considerably boosted the performance of the classifier compared to the results of ORACLE's extended paper (0.35 accuracy for data collected over-the-air), yet, there are several parameters that are still influencing Thu performance.

• The distance between the transmitter and the receiver is shown to be a highly impactful element on the CNN's performance, in opposition to the experience environments (time and location) element, whose effect seems to be highly attenuated with filtering and normalization.

• The number of devices has a major effect on the learning process as well, we can observe a relative decrease in the training accuracy when the number is greater that 4 devices .

• We have also noticed that in the confusion matrices the classifier tends to be confused between the device 2 and 4 when training in the same distance, which is probably due to very similar properties of the two devices.

• During the training we have also noted unusual behavior of the neural network in few cases. One of these common issues are the grokking phenomenon and the vanishing / exploding gradient phenomenon.

4.6.2.1 Grokking phenomenon

In some cases, when the number of epochs is relatively high (300 for example), we have noticed that the validation accuracy would drop to 0 after being settled on 0.99. In (50) a study has been conducted on this purpose, and it was shown that if this phenomenon might be caused by many elements.

Mostly, a poor choice of the number of epochs is one of the most frequent causes. In fact, when the number of epochs is too high , we have a low value for batch-per-epoch. Therefore, the model learns fast over small amounts of data , reaches over-fitting and fails to generalize (after a certain time starts to memorize data rather than learning patterns). Thus, the optimal solution is to reduce the number of epochs accordingly with the convergence speed of the model until reaching the optimal value.

4.6.2.2 Unstable gradient

Another issue that has appeared several times when training the neural network, consisting in the fact that the neural network doesn't learn, or at least learns extremely slowly which makes it stuck in the accuracy of the first epochs. This is the problem of unstable

gradients.
Specifically, this problem happens within the wights of the early layers of the NN . During the training process, stochastic gradient descent is being updated with respect to the weights as we explained in earlier parts of this report. In some cases , this calculated gradient with respect to the earlier layers becomes extremely small. This small value is used to update the weight , hence if the gradient is vanishingly small, the update will be vanishingly small as well, thus very limited changing will be transferred within the network to reduce the loss. This issue can be mainly caused by a low value of weights of the first layers.

Similarly, if the weights within the earlier layers are too large, the gradient will be larger with each layer of the network, hence exploding gradient problem. This exploding gradient is the key behind updating the weights. The movement of the weights are then large that the optimal value will probably never be achieved because the proportion to which the weight is being updates is too large and continues to move away from its optimal value.

4.6.2.3 Potential future solutions

We assume that diagnosing these issues is a big step towards solving them. In this section we propose some potential techniques that might considerably increase the performance of the CNN.
• Starting with the last problem, i.e Unstable gradients, we found that multiple solutions exist and might help with reducing this phenomenon. For example, weight initialization technique is a simple way to avoid extreme values of weights that are randomly initialized. Typically these random values will be normally distributed with a mean of 0 and a standard deviation of 1. There's a detailed study in [37] showing the whole theory behind weight initialization where the variance of these weights is calculated with respect to the input and output value of a certain neuron.
Some of the well-known initialization techniques is $'Xavier\ initialization'$ developed in the previous paper. In practice keras offers the option of choosing the initializer of the weights such as $'glorot\_uniform'$, $'glorot\_normal'$ ..etc.

• To overcome the effect of the distance, we suggest normalizing data with respect to the SNR introduced by the distance. We conceptualize this approach from the basis that the impact of the distance between the transmitter and the receiver is due to more or less SNR values. We believe that if we find an optimal SNR value that gives a high accuracy,

normalizing data with respect to this SNR would help eliminate the impact.

• We also propose to perform fine-tuning to have an optimal performance under different circumstances. This can be done by saving the optimal weights and parameters of the model on a certain distance, the import it fine-tune it and retrain it one data from a different distance. Fine-tuning might include adding a removing layer, increasing or decreasing the hyper-parameters until finding the best option.

4.7 Conclusion

The implementation of the model using ORACLE's architecture have indeed shown great results on wired communication either on ORACLE's dataset or on SCEE's testbed dataset, as we anticipated. Through this chapter we have cited our analysis and solutions for the decrease in the performance in wireless communication. After improvements, we had perfect results on relatively small numbers of devices , and where the distance between the transmitter and the receiver is quite stable. These two axis are the center of attention for the potential improvements that we might implement to generalize the near perfect results we had on any condition of data capturing.

General Conclusion

Radio frequency fingerprinting have been increasingly put under the spot during the previous years, mainly because of the necessity of such concept in assuring the desired security in IOT network.

The motivation for developing research work in this regard comes from the fact that the nature of RFF as a concept relies on inherent hardware imperfections that are difficult to reproduce and that assures the efficiency of these techniques in comparison of other traditional ones.

This project tackles the problem of integrating machine learning in RFF as an efficient, performant and up-to-date alternative.

In this document we have covered big lines of the research and implementation steps of this work. That was done by first introducing the key concepts related to RFF. Then a brief exposition of the related work and the major inspirations was presented according to the used features and classification algorithms.

After studying the papers related to ORACLE, a CNN-based classification algorithm, we adopted its architecture with some technical changes regarding the model's architecture, training and evaluation process in order to build our RFF classifier.

Indeed, after integrating multiple data processing and model enhancement techniques we have came to achieve similar result (99.6% of accuracy) on wirely transmitted signals and better performance on wirelessly transmitted data (99% of accuracy up to 5 devices).

Although the performance of the classifier was limited when a greater number of devices is introduced and when the element of distance interferes, we have proposed methods that we believe can overcome these challenges.

We propose to operate normalization of the data with respect to the SNRs to avoid the unwanted effect of the distance. We also suggest pure machine learning techniques that have proved efficiency with image and video processing , such as fine-tuning and weight initialization to reshape the model every time that data under new circumstances is introduces.

Finally to enhance the applicability of the system in real life, we suggest to add a python module that collects data in real-time process it and pass it to the classifier, in order to classify devices and eventually recognize malicious attacks.

Bibliography

[1] MY. Andrea H. C. Choe, C. E. Poole et H. H.Szu : Novel identification of intercepted signals from unknown radio transmitters. *Proc. SPIE, Wavelet Applications II*, 2491:504–517, April 1995.

[2] Jeyanthi Hall, Michel Barbeau et Evangelos Kranakis : Enhancing intrusion detection in wireless networks using radio frequency fingerprinting. *Communications, Internet, and Information Technology*, 01 2004.

[3] L. Di Gregorio M. Leonardi et D. Di Fausto : Air traffic security: aircraft classification using ads-b message's phase-pattern. *Aerospace*, 4:51, 2017.

[4] K. Sowerby S. U. Rehman et C. Coghill : Rf fingerprint extraction from the energy envelope of an instantaneous transient signal. *Australian Communications Theory Workshop (AusCTW)*, pages 90–95, 2012.

[5] Y. Zhang X. Li et M. G. Amin : "multifrequency-based range estimation of rfid tags,. *2009 IEEE International Conference on RFID*, pages 147–154, 2009.

[6] J. Toonstra et W. Kinsner : A radio transmitter fingerprinting system odo-1. *Proceedings of 1996 Canadian Conference on Electrical and Computer Engineering*, pages 60–63, 1996.

[7] Y. Yang N. Soltanieh, Y. Norouzi et N. C. Karmakar : A review of radio frequency ngerprinting techniques. *IEEE Journal of Radio Frequency Identication,*, 4(3):222–233, 2020.

[8] J. Toonstra et W. Kinsner : Transient analysis and genetic algorithms for classification. *Proc. Conf. Commun. Power Comput. (IEEE WESCANEX)*, pages 432–437, 1995.

[9] K. Ellis et N. Serinken : Characteristics of radio transmitter fingerprints. *Radio Sci*, 36(4):585–597, 2001.

[10] N. Serinken O. H. Tekbas et O. Ureten : "an experimental performance evaluation of a novel radio-transmitter identification system under diverse environmental conditions,". *Can. J. Elect. Comput. Eng*, 29(3):203–209, 2004.

[11] O. Üreten Ö. H. Tekbas et N. Serinken : "improvement of transmitter identification system for low snr transients,". *Electron. Lett.*, 40(3):182–183, 2004.

[12] J. Hall M. Barbeau et E. Kranakis : "detection of rogue devices in bluetooth networks using radio frequency fingerprinting,". *Proc. 3rd IASTED Int. Conf. Commun. Comput. Netw. (CCN),*, pages 4–6, 2006.

[13] K. B. Rasmussen et S. Capkun : "implications of radio fingerprinting on the security of sensor networks,". *Proc. 3rd Int. Conf. Security Privacy Commun. Netw. Workshops (SecureComm),*, pages 331–340, 2007.

[14] O. Ureten et N. Serinken : "wireless security through rf finger-printing,". *Can. J. Elect. Comput. Eng.*, 32(1):27–33, 2007.

[15] M. Gruteser V. Brik, S. Banerjee et S. Oh : "wireless device identifi-cation with radiometric signatures,. *Proc. 14th ACM Int. Conf. Mobile Comput. Netw*, pages 116–127, 2008.

[16] F. J. Mullany M. M. Buddhikot K. E. Nolan and T. W. Rondeau I. O. Kennedy, P. Scanlon : "radio transmitter fingerprinting: A steady statefrequency domain approach,". *Proc. IEEE 68th Veh. Technol. Conf.*, pages 1–5, 2008.

[17] M. A. Temple M. J. Mendenhall W. C. Suski, II et R. F. Mills : "using spectral fingerprints to improve wireless network security,". *Proc. IEEE Glob. Telecommun. Conf. (IEEE GLOBECOM),*, pages 1–5, 2008.

[18] M. A. Temple R. W. Klein et M. J. Mendenhall : "application of wavelet-based rf fingerprinting to enhance wireless network security,". *J. Commun. Netw.*, 11(6):544–555, 2009.

[19] D. R. Thompson S. C. G. Periaswamy et J. Di : "fingerprinting rfid tags,". *IEEE Trans. Depend. Secure Comput.*, 8(6):938–943, 2011.

[20] T. S. Heydt-Benjamin B. Danev et S. Capkun : "physical-layer iden-tification of rfid devices,". *Proc. USENIX Security Symp.*, pages 199–214, 2009.

[21] S. Jana et S. K. Kasera : "on fast and accurate detection of unauthorized wireless access points using clock skews,". *IEEE Trans. Mobile Comput.*, 9(3):449–462, 2010.

[22] A. Broido T. Kohno et K. C. Claffy : "remote physical device fingerprinting,". *IEEE Trans. Depend. Secure Comput.*, 2(2):93–108, 2005.

[23] C. Shahriar C. Lu W. Lou F. Chen, Q. Yan et T. C. Clancy : "on passive wireless device fingerprinting using infinite hidden markovrandom field,". *Submitted for publication.*

[24] Z. Han N. T. Nguyen, G. Zheng et R. Zheng : "device fingerprinting to enhance wireless security using nonparametric bayesian method,". *in IEEE INFOCOM,*, pages 1404–1412, 2011.

[25] F. J. Mullany M. M. Buddhikot K. E. Nolan and T. W. Rondeau I. O. Kennedy, P. Scanlon : "radio transmitter fingerprinting: A steady state frequency domain approach,". *in IEEE VTC*, pages 1–5, 2008.

[26] A. S. Uluagac S. V. Radhakrishnan et R. Beyah : "gtid: A technique for physical device and device type fingerprinting,". *IEEE Transactions on Dependable and Secure Computing,*, 2015.

[27] T. J. O'Shea et J. Corgan : "convolutional radio modulation recognition networks,". 2016.

[28] S. Ioannidis S. Riyaz, K. Sankhe et K. Chowdhury : "deep learning convolutional neural networks for radio identification,". *IEEE Communi- cations Magazine,*, 56(9): 146–152, 2018.

[29] Kunal Sankhe, Mauro Belgiovine, Fan Zhou, Shamnaz Riyaz, Stratis Ioannidis et Kaushik Chowdhury : Oracle: Optimized radio classification through convolutional neural networks. pages 370–378, 04 2019.

[30] Kunal Sankhe, Mauro Belgiovine, Fan Zhou, Luca Angioloni, Francesco Restuccia, Salvatore d'oro, Tommaso Melodia, Stratis Ioannidis et Kaushik Chowdhury : No radio left behind: Radio fingerprinting through deep learning of physical-layer hardware impairments. *IEEE Transactions on Cognitive Communications and Networking*, PP:1–1, 10 2019.

[31] Alex Krizhevsky, Ilya Sutskever et Geoffrey Hinton : Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012.

[32] Arnault Chazareix : About convolutional layer and convolution kernel. Link on internet < https://www.sicara.fr/blog-technique/2019-10-31-convolutional-layer-convolution-kernel >.

[33] Frédéric Launay : 5g radio interface – frames, numerology and resource allocation. Link on internet <https://blogs.univ-poitiers.fr/f-launay/tag/ofdm/>.

[34] L. Mohamad Fadul : *THE IMPACT OF RAYLEIGH FADING CHANNEL EFFECTS ON THE RF-DNA FINGERPRINTING PROCESS.* Thèse de doctorat, The University of Tennessee at Chattanooga Chattanooga, Tennessee, August 2018. Link on internet < https://scholar.utc.edu/cgi/viewcontent.cgi?article=1726context=theses >.

[35] Irwin Kennedy et Alexandr Kuzminskiy : Rf fingerprint detection in a wireless multipath channel. pages 820–823, 09 2010.

[36] Cherif Diouf, Gerard Janssen, Han Dun, Tarik Kazaz et C.C.J.M. Tiberius : A usrp-based testbed for wideband ranging and positioning signal acquisition. *IEEE Transactions on Instrumentation and Measurement*, PP:1–1, 03 2021.

[37] Alethea Power, Yuri Burda, Harrison Edwards, Igor Babuschkin et Vedant Misra : Grokking: Generalization beyond overfitting on small algorithmic datasets, 01 2022.