Synthèse de développement

Projets Industriels

PI07

VALETTE Léo – POINARD Noé – ROUSSEAU Timothée – GRENIER Mathis

JANVIER – JUIN
2023

Scanner cellulaire 4G et 5G innovant assisté par IA

Scanner Cellulaire 4G & 5G assisté par IA	Projets Industriels 2023
Objet : Synthèse de développement	**Date 27/06/2023**

Table des matières

1. Contexte

Au sein de la division cellulaire d'Avantix, le pôle de recherche et développement souhaite développer un PoC (Proof of Concept) de scanner cellulaires 4G & 5G basé sur l'utilisation de modèle d'intelligence artificielle pour améliorer les performances du scan. Pour rappel, l'objectif premier d'un scanner est de caractériser finement un environnement radiofréquence. Cette caractérisation, passive, a pour objectif d'identifier les cellules des opérateurs présents dans l'environnement du scanner, avec une granularité plus ou moins élevée, sur des bandes définies à l'avance, et de récupérer leur configuration.

La particularité de ce scanner réside sur la détermination de fréquences d'intérêt via des modèles d'Intelligence Artificielle (IA) de type Machine Learning ou Deep Learning. L'utilisation des scanners existants, fonctionnant de manière itérative, devient chronophage avec l'apparition des nouvelles technologies 5G, multipliant les bandes de fréquences à analyser ainsi que leur largeur. L'utilisation de l'intelligence artificielle devient donc essentielle afin d'augmenter la rapidité et la pertinence du scanner. Son rôle est de détecter les bandes de fréquences d'intérêt en effectuant un filtrage des pics de puissance observés, puis, d'effectuer une classification de technologie entre la 4G et la 5G.

Le scanner est composé de différentes de briques logicielles. La brique d'acquisition paramétrable s'appuie sur une plateforme SDR de type USRP permettant d'analyser les cellules 4G et 5G. Les modèles d'IA sont entrainés par des datasets, ensemble de données représentatives des données réelles. Leur génération numérique repose sur des toolboxs Matlab permettant d'émuler la partie Base Station (antenne émettrice) de la connexion à un réseau cellulaire. La partie de récupération des configurations des cellules repose également sur l'utilisation de Matlab. Cependant, cette dernière partie n'est pas être le cœur du projet étant donné que des solutions plus performantes et robustes existent déjà au sein de l'entreprise.

2. Pré-étude

2.1. Réseaux cellulaires

La 4G et la 5G étant des normes de réseaux cellulaires de téléphonie mobile, il est important de revenir sur les caractéristiques de ces réseaux.

Au cours des 30 dernières années, nous avons assisté à l'évolution de plusieurs générations de technologies qui ont transformé les services de communication. Ces avancées ont permis de passer des simples services de communication téléphonique, basés sur la voix et la transmission de messages texte, à des services axés sur les données, tels que l'internet mobile, le très haut débit et les objets connectés : GSM, 3G, 4G/LTE (Long Term Evolution) et 5G.

Ces réseaux ont une structure « cellulaire » qui permet d'utiliser de nombreuses fois les mêmes fréquences. Chaque cellule est une zone géographique couverte par une station de base composée de plusieurs antennes émettant à 120° chacune. Ces cellules sont de taille variable en fonction de la densité de population et du nombre de communications afin de s'adapter aux besoins. Elles se chevauchent pour faire en sorte qu'un utilisateur en mouvement puisse changer de cellule sans coupure de communication.

Les différents opérateurs se voient attribuer des bandes de fréquence sur lesquelles ils peuvent déployer des stations de base. Les appareils mobiles peuvent alors se synchroniser sur une de ces stations et recevoir ou émettre des données. Selon la norme utilisée, les stations utilisent différentes modulations pour partager la bande entre tous les utilisateurs, pour la 4G et la 5G, c'est l'OFDMA (dérivé de l'OFDM).

Figure 1 : Distribution des bandes de fréquences en France par opérateur

Pour permettre aux utilisateurs de se synchroniser à elle, les stations de base émettent régulièrement des blocs d'informations : le « Master Information Bloc » (MIB) et les « System Information Bloc » (SIB). Le MIB contient les paramètres principaux de la cellule, son identifiant et permet d'accéder aux SIBs qui contiennent d'autres informations de paramétrage et de synchronisation.

2.2. Modulation OFDM

Les technologies 4G et 5G sont caractérisées par l'utilisation de la modulation OFDM (Orthogonal Frequency Division Multiplexing). Dans cette modulation, le signal numérique à transmettre est répartie sur un certain nombre de sous-porteuses, orthogonales les unes par rapport aux autres, afin de maximiser l'efficacité spectrale. L'utilisation de ce type de modulation permet de réduire les interférences, le bruit et l'atténuation, et permet un débit supérieur à d'autres types de modulations.

Dans le domaine spectral, la modulation OFDM est très reconnaissable, l'orthogonalité des sous-porteuses rendant le signal rectangulaire :

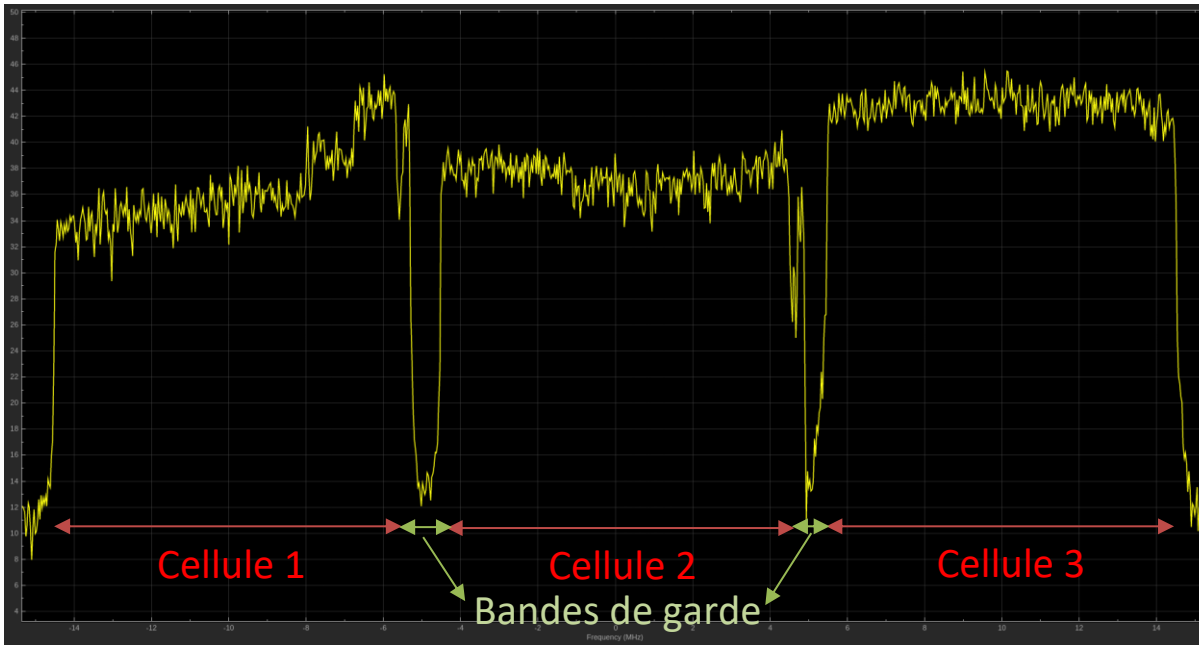


Figure 2 : Bande B20 (790-820 Mhz) : 3 cellules de 10 Mhz

Cette forme est significativement différente des autres signaux que l'on peut retrouver sur le spectre de fréquence. Par conséquent, il est possible d'utiliser cette particularité pour distinguer la 4G et 5G des autres technologies. Par exemple, dans la modulation OFDM, le front montant possède une transition rapide, ce qui n'est pas le cas dans d'autres technologies :

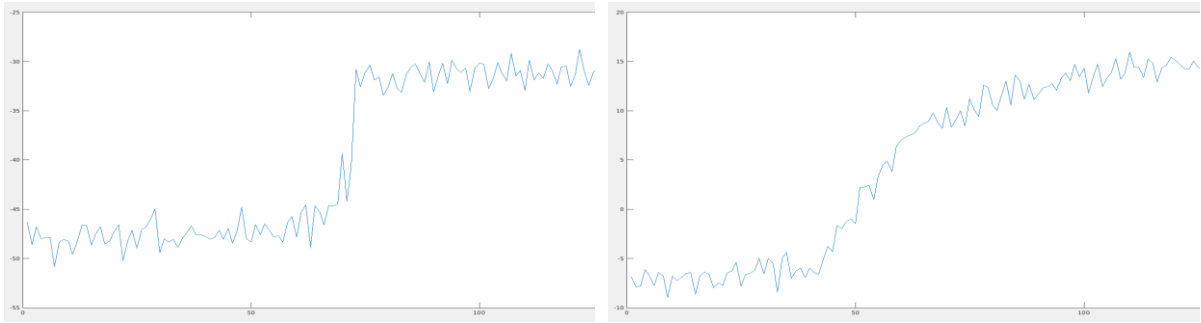


Figure 3 : Comparaison des fronts montants OFDM et 3G

2.3. Null sub-carriers

Dans la section précédente, nous avons constaté qu'il était possible de distinguer les technologies 4G et 5G, qui utilisent toutes deux la modulation OFDM, des autres technologies présentes sur le spectre. Maintenant, nous devons trouver une méthode permettant de classifier ces deux technologies entre elles. Pour cela, nous pouvons utiliser différentes transformations du signal pour trouver des caractéristiques permettant leur différentiation. Une expertise des signaux réalisés conjointement entre l'équipe projet et l'entreprise nous a permis de remarquer une chute de puissance au sein des spectres de certaines cellules. Ce drop de puissance apparait uniquement sur les signaux 4G d'après la norme. Les équations ci-dessous prouvent que le DC carrier n'est pas présent en 4G (LTE) mais qu'il est présent en 5G (NR) :

$$s_l^{(p)}(t) = \sum_{k=-\lfloor N_{RB}^{DL} N_{sc}^{RB}/2 \rfloor}^{-1} a_{k^{(-)},l}^{(p)} \cdot e^{j2\pi k\Delta f\left(t-N_{CP,l}T_s\right)} + \sum_{k=1}^{\lceil N_{RB}^{DL} N_{sc}^{RB}/2 \rceil} a_{k^{(+)},l}^{(p)} \cdot e^{j2\pi k\Delta f\left(t-N_{CP,l}T_s\right)} \quad : \text{LTE}$$

$$k^{(-)} = k + \lfloor N_{RB}^{DL} N_{sc}^{RB}/2 \rfloor \qquad k^{(+)} = k + \lfloor N_{RB}^{DL} N_{sc}^{RB}/2 \rfloor - 1$$

The point at k=1 is not used for wavefor generation because it is DC carrier

Figure 4 : 4G waveform equations

$$s_l^{(p,\mu)}(t) = \sum_{k=-\lfloor N_{RB}^{\mu} N_{sc}^{RB}/2 \rfloor}^{\lceil N_{RB}^{\mu} N_{sc}^{RB}/2 \rceil - 1} a_{k',l}^{(p,\mu)} \cdot e^{j2\pi k\Delta f\left(t-N_{CP,l}\cdot T_s\right)} \quad : \text{NR}$$

The point at k=0 is used for wavefor generation

Figure 5 : 5G waveform equations

Les équations de la génération de la forme d'onde (*waveform equations*) peuvent sembler très différentes à première vue, mais en vérité la différence est minime. En LTE (4G), l'équation de génération de la forme d'onde (essentiellement l'équation IFFT) est divisée en deux parties uniquement pour supprimer le point situé à la position DC (c'est-à-dire à la fréquence = 0 dans la bande de base). En NR (5G), cette suppression du DC n'est plus nécessaire et l'équation IFFT complète est combinée en une seule. Par conséquent, nous allons pouvoir nous baser sur cette différence dans la forme d'onde pour effectuer une classification entre 4G et 5G.

Concrètement, sur le spectre, la différence entre les deux technologies est assez visible :

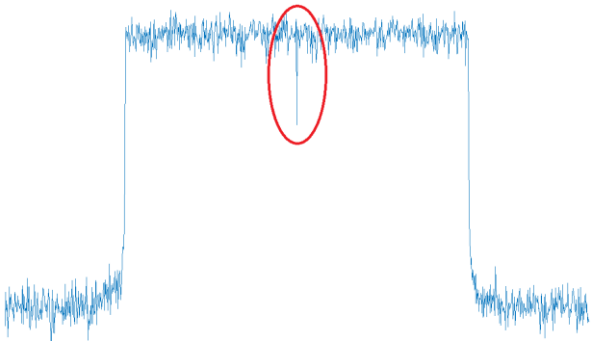


Figure 6 : Spectre d'une cellule 4G

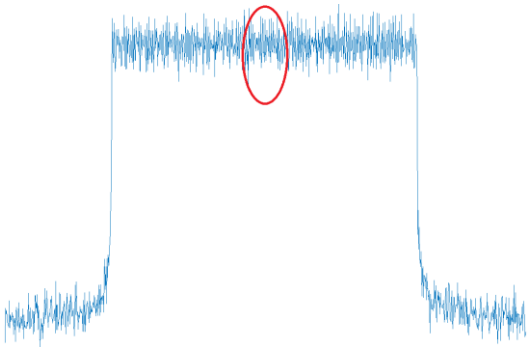


Figure 7 : Spectre d'une cellule 5G

Il est rapidement observable qu'il y a une chute d'amplitude au centre du spectre dans le cas de la technologie 4G, ce qui n'est pas le cas pour la 5G. Cette différence peut donc être utilisée pour classifier ces deux technologies distinctement.

2.4. Conception

Le scanner fonctionne en 6 phases distinctes : le paramétrage du scan, l'acquisition du signal, la reconnaissance des pics de puissance, le filtrage, la classification de technologie, la récupération des blocs d'informations. Son fonctionnement est détaillé dans le schéma fonctionnel suivant :

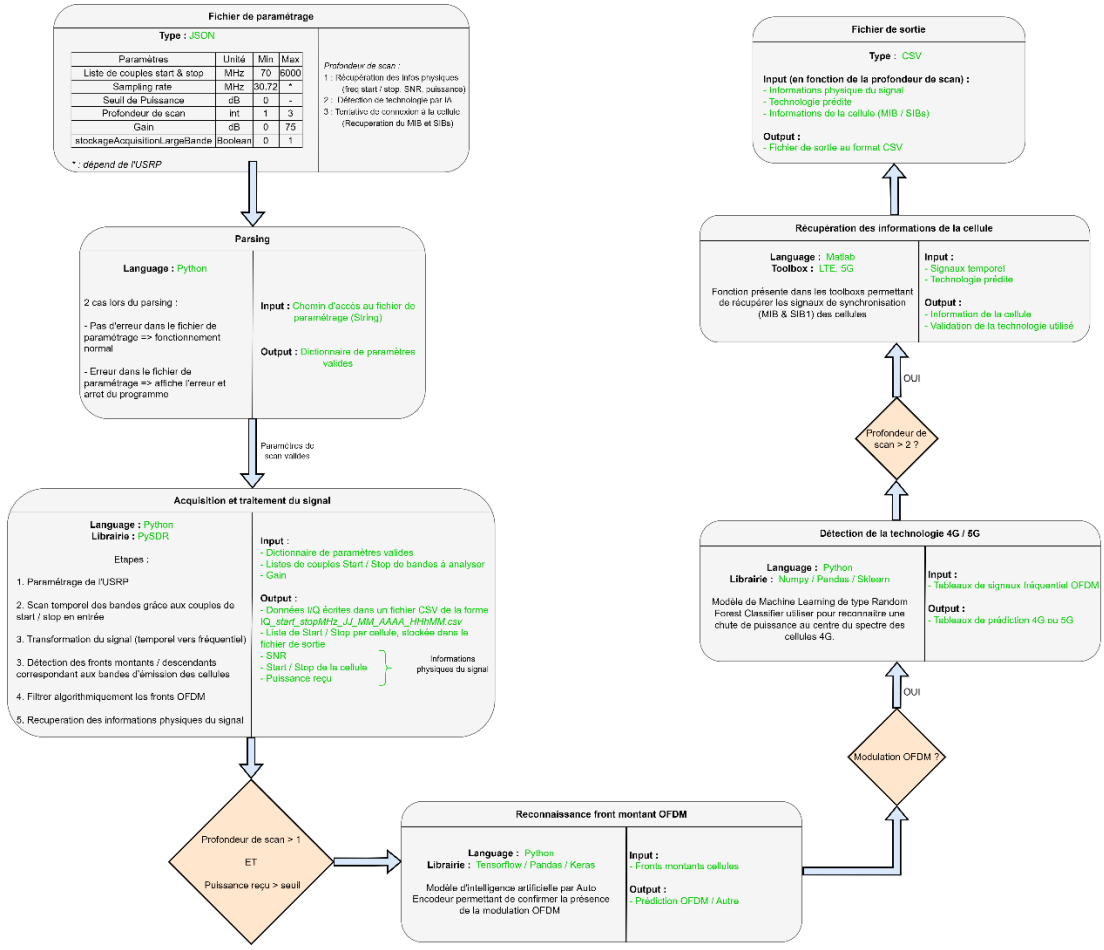


Figure 8 : Schéma de conception

Le paramétrage est représenté par les deux premiers blocs du schéma. Dans un premier temps, nous présentons le fichier de paramétrage puis nous en vérifions le contenu avec le bloc de « parsing ».

Les paramètres récupérés sont transmis au bloc d'acquisition du signal. Celui-ci réalise l'acquisition de l'environnement radiofréquence grâce à un USRP (Universal Software Radio Peripheral). Dans un second temps, le signal va être mis en forme afin d'y appliquer différents filtres dans le but de faire ressortir les fronts montants et descendants spécifiques à la modulation OFDM. Différentes informations physiques du signal sont récupérées sur les bandes de fréquences pertinentes. Ces bandes sont ensuite transmises aux blocs d'intelligence artificielle afin d'en valider leur pertinence.

Le premier bloc d'IA fonctionne à la manière d'un détecteur d'anomalie avec un modèle d'auto-encodeur. Celui-ci, se concentrant sur les fronts montants observés, vérifie leur correspondance avec la forme générique d'un signal OFDM.

Si le signal observé est OFDM, alors celui-ci sera analysé par le second bloc d'IA, se concentrant sur le centre des cellules afin de repérer le drop de puissance (null sub-carrier) présent en 4G. Si la chute de puissance n'est pas repérée au centre de la cellule, alors le signal observé est considéré comme étant une technologie 5G.

Une fois la technologie déterminée, nous récupérons les blocs d'informations (MIB & SIB1) en utilisant les fonctions présentes dans les toolboxs Matlab fournis par Avantix.

Toutes les informations récupérées sont stockées dans un fichier de sortie au format CSV.

3.Scanner

3.1. Acquisition du signal

Afin de détecter les cellules présentes dans l'environnement radio, il est nécessaire de faire une acquisition de cet environnement. Pour cela, nous utilisons un USRP (Universal Software Radio Peripheral), carte électronique qui communique avec l'ordinateur par connexion USB ou Ethernet pour interagir généralement avec une radio logicielle (SDR). Dans ce projet, la communication avec l'ordinateur ce fait uniquement par USB et l'USRP utilisé est un B200. Cette carte comporte quatre convertisseurs numérique-analogique haute vitesse, quatre convertisseurs analogique-numérique haute vitesse, un FPGA et quelques composants glue logic. Dans la perspective future, il est important que le scanner soit compatible avec des USRPs plus performant. C'est pourquoi nous avons dû tenir compte de cette compatibilité tout au long du projet. Pour faire fonctionner un USRP avec un ordinateur, il est nécessaire d'installer le driver libUHD. Le développement de ce scanner se faisant sur Python, nous avons utilisé la bibliothèque pySDR qui fournit des fonctions essentielles à l'acquisition telle que la fonction recv_num_samps() responsable de l'acquisition des signaux temporels de l'environnement radio. L'utilisateur choisit différents paramètres comme la fréquence d'échantillonnage, le nombre de points ou encore le gain pour que la fonction recv_num_samps() puisse faire l'acquisition avec ces paramètres. Ces acquisitions temporelles sont stockées sous la forme de signaux temporels « I/Q ». Cette étape d'acquisition est faite par la fonction 'Scan1()', seule fonction du fichier « Scan1.py » (cf documentation ScriptScanner). Une fois l'acquisition temporelle effectuée, le signal est transformé afin que l'algorithmie puisse repérer les cellules présentes dans l'environnement.

3.2. Transformation du signal

Après s'être renseigné sur les caractéristiques signantes d'un signal OFDM, nous avons choisi d'effectuer la détection par reconnaissance de la forme du spectre d'un signal OFDM. En effet, dans le spectre fréquentiel, les cellules OFDM se remarquent par leur forme rectangulaire. Ce sont les seuls signaux du spectre à avoir cette forme particulière. Ainsi, on applique une transformée de Fourier sur les données temporels I/Q afin d'obtenir le spectre de l'environnement radiofréquence. Par la suite, un processus de filtrage sera appliqué sur le spectre pour mettre en évidence les pics de puissance semblables à des fronts OFDM.

3.3. Détection de puissance

Une fois l'acquisition du signal temporel et sa transformation effectués par la fonction 'Scan1()', le script détecte chaque front montant et chaque front descendant du spectre à l'aide d'une dérivée sur le signal fréquentiel. Les pics de puissance du spectre sont relevés, et un seuil est fixé pour filtrer le bruit sur la dérivée. Un seuil volontairement bas est utilisé pour permettre que chaque pic puisse être inclus dans le processus de filtrage. Cette détection des zones de puissance se trouve dans la fonction analyseScan1() dans le fichier « fonctions_scan1.py ».

3.4. Filtrage

Le processus de filtrage est divisé en deux parties distinctes. La première partie se concentre sur les transitions ascendantes et descendantes détectées grâce à la détection de puissance. La deuxième partie se focalise sur les bornes supposées représenter les cellules identifiées.

Au début du filtrage, les fronts montants et descendants sont très nombreux, c'est pourquoi il est nécessaire d'effectuer un tri. Dans certains cas, le même front montant apparait comme plusieurs pics distincts sur la dérivée. Par conséquent, le premier filtre consiste à regrouper les pics correspondant au même front sur le spectre. De plus, sur les fronts montants, les pics détectés se situent au milieu de celui-ci. Afin d'éviter de couper le début de la cellule, un décalage permettant de déplacer les fronts montant avant le début de la cellule est appliqué.

Afin de former des couples de fronts montants et descendants pouvant constituer une cellule, le script de filtrage effectue des associations en fonction des largeurs des cellules OFDM connues. Pour ce faire, un filtre analyse la largeur spectrale entre chaque front montant et descendant, puis forme des couples en fonction des largeurs possibles. Dans le PoC (Proof of Concept), les largeurs de cellules prises en compte sont 5MHz, 10MHz, 15MHz et 20MHz. Cependant, une marge d'erreur est prévue pour ces largeurs car les cellules n'émettent jamais exactement avec la bonne largeur, notamment à cause de la bande de garde entre les cellules. C'est pourquoi la bande 5MHz accepte une largeur allant de 4MHz à 5,6MHz, la bande 10MHz accepte une largeur allant de 9MHz à 10,6MHz, la bande de 15MHz accepte une largeur allant de 13,5 à 15,6 MHz et la bande de 20MHz accepte une largeur allant de 18MHz à 20,6 MHz.

Une fois ces couples formés, une analyse est effectuée sur la différence d'amplitude entre le front montant et les premiers points du plateau de la cellule. Si cette différence est inférieure au seuil de puissance du fichier de paramétrage alors le couple est supprimé de la liste. Cela veut dire que la cellule ne ressort pas assez du plancher de bruit. Un second filtre vérifie qu'entre le front montant et le front descendant aucun point ne descende en dessous de l'amplitude minimale entre ces deux fronts. Si un point ou plus rempli ce critère alors le couple est supprimé de la liste des cellules. Un autre filtre, se rapprochant du précédent, s'assure qu'il n'y ait pas de chute d'amplitude trop importante entre le front montant et le front descendant. Un seuil à 30% de l'espacement entre le plancher et le plafond de la cellule est fixé, si le nombre de points en dessous du seuil est supérieur à 3 alors le couple et supprimé de la liste. Un filtre intervient juste après, il compare chaque couple de la liste avec le couple suivant et il calcule un pourcentage de ressemblance entre eux. Si ce pourcentage est supérieur à 80% alors on va garder le couple le plus large. Sinon, on garde les deux couples. Cela permet d'éviter la superposition de cellules. Enfin, les couples passent dans un dernier filtre qui compare les fronts montants de chaque couple avec celui d'après. Si ces deux fronts montants consécutifs sont les mêmes alors le plus large des couples est conservé. De manière similaire, le même processus est effectué sur les fronts descendants.

3.5. Récupération des blocs d'informations

Afin de récupérer les blocs d'informations, il est possible d'utiliser des logiciels open-source, comme SRSRAN, ou Open Air Interface. Il est aussi possible d'utiliser des scripts MATLAB accessibles via des toolboxs payantes. La récupération de ces blocs d'informations se décompose en plusieurs étapes : la récupération du MIB (Master Information Block), le décodage du MIB qui permet de trouver les SIB (System Information Block) puis la récupération des SIBs et leur décodage.

Pour récupérer ces blocs d'informations, il faut s'assurer que le sampling rate soit à une valeur suffisante par rapport à la bande passante de la cellule. Dans la majorité des cas, une valeur de sampling rate de 30.72 Msamples/s suffit.

La bande passante de la cellule n'étant pas connue, seule la partie centrale du signal va être démodulée. Les signaux de synchronisation sont d'abord récupérés, et une corrélation dans les domaines temporels et fréquentiels va être réalisée afin d'obtenir l'identité de la cellule, ce qui permet ensuite de savoir quel est le « duplexing » utilisé et la longueur du préfixe cyclique. Avec ces informations, la démodulation peut être réalisée, ce qui permet de trouver le MIB. Le décodage du MIB permet ensuite d'avoir les informations permettant de trouver les SIB et la bande passante de la cellule.

Avec les informations du MIB obtenues, le signal entier est alors démodulé. On observe alors les canaux de contrôle pour savoir si un SIB est présent, et avoir les éléments de décodage du SIB le cas échéant. Le SIB est alors récupéré et décodé grâce aux informations obtenues.

MIB Content	
Downlink Channel Bandwidth	
PHICH Configuration	PHICH Duration
	PHICH Resource
System Frame Number (SFN)	

SIB1 Content	
Cell Access Information	PLMN, Cell Identity, Cell Barred, CSG
Cell Selection Information	
Pmax	
Frequency Band Indicator	
Scheduling Information List	Periodicity, Mapping, Window length, Tag

Figure 9 : Contenu du MIB et du SIB1

4.IA

4.1. Concept général

L'expression Intelligence Artificielle décrit toutes les méthodes mises en place par des opérateurs pour automatiser une prise de décision en vue de simuler celle d'un humain. L'intelligence artificielle (IA) comprend différentes approches pour simuler l'intelligence humaine. Deux concepts clés sont le Machine Learning et le Deep Learning.

Le Machine Learning (ou apprentissage automatique) fait appel à des études statistiques réalisées par l'opérateur en vue de sélectionner les informations capitales du problème (l'expertise est menée par l'opérateur, et l'apprentissage par la machine). Le Deep Learning en revanche utilise d'autres méthodes plus coûteuses (les réseaux de neurones profonds – CNN, RNN, LSTM…), comme des algorithmes très complexes permettant de simuler l'expertise de l'utilisateur. Moyennant une complexité bien plus élevée de son algorithme, l'opérateur n'a pas besoin de sélectionner les informations pertinentes du problème auquel il est confronté, l'algorithme le fait lui-même. Cette méthode nécessite cependant beaucoup plus de données et de ressource matérielle.

Il existe plusieurs types d'apprentissage dans l'intelligence artificielle. L'apprentissage supervisé et l'apprentissage par renforcement sont tous deux des approches puissantes de l'intelligence artificielle, mais ils diffèrent dans leur mode d'apprentissage. L'apprentissage supervisé nécessite des exemples étiquetés pour effectuer l'apprentissage. En utilisant ces données d'entraînement labellisées, le système apprend à généraliser et à faire des prédictions précises sur de nouvelles données non étiquetées en repérant les caractéristiques signantes de la donnée. En comparaison, l'apprentissage non supervisé va effectuer son entrainement sur des données non labellisées. L'algorithme va extraire les caractéristiques similaires des données et les regrouper par similarité pour définir les classes.

D'un autre côté, l'apprentissage par renforcement est basé sur un modèle d'interaction entre l'utilisateur et l'environnement. L'algorithme apprend à prendre des décisions en maximisant une récompense ou en minimisant une pénalité lorsqu'il interagit avec l'environnement. Au fil du temps, il développe une stratégie qui lui permet de prendre les meilleures décisions possibles dans cet environnement spécifique.

De plus, les hyperparamètres jouent un rôle crucial dans la configuration et l'optimisation des modèles d'IA. Les hyperparamètres sont des paramètres externes qui ne sont pas appris par le modèle lui-même, mais qui déterminent comment il apprend et se comporte pendant le processus d'entraînement. Ils peuvent inclure des choix tels que le taux d'apprentissage, le nombre d'itérations, la taille du lot (batch size), la régularisation et bien d'autres. Ces hyperparamètres influencent directement la performance du modèle en termes de précision, de vitesse de convergence et de capacité à généraliser à de nouvelles données. Trouver les bonnes valeurs d'hyperparamètres est une tâche d'optimisation qui nécessite souvent une expérimentation itérative pour déterminer les réglages optimaux, notamment par parcours de grille.

4.2. Workflow

L'un des livrables du projet est le Workflow de détection de technologie par IA. C'est une suite d'étapes menant à la réalisation du projet. Il permet d'assurer la reproductibilité et une meilleure organisation des tâches.

La création et l'utilisation d'un workflow permet de normaliser tout le processus de conception, de création et d'implémentation d'un modèle d'intelligence artificielle. Nous avons pu répéter ce workflow en changeant certains paramètres dans le but de comparer les résultats et de comprendre l'impact de chaque variable.

Ci-dessous, le schéma de notre workflow :

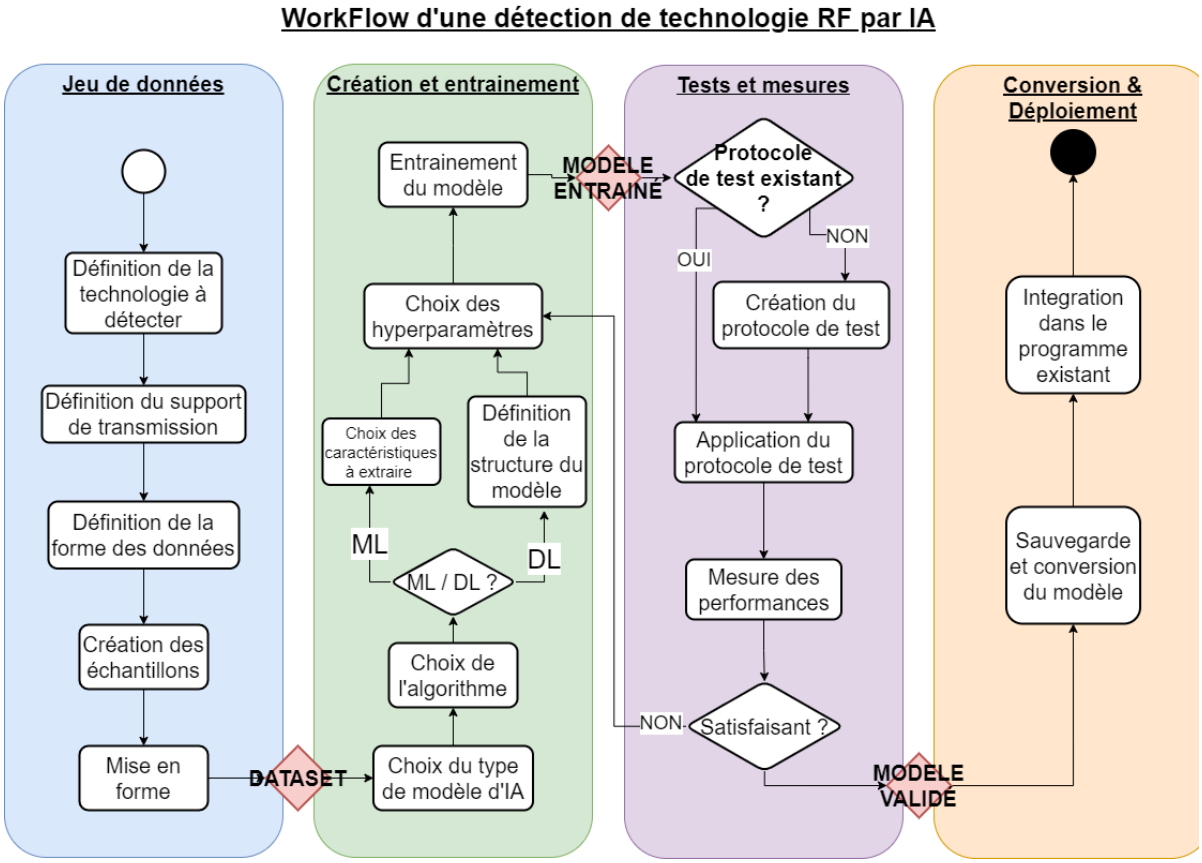


Figure 10 : Workflow d'une détection d'une technologie cellulaire assisté par IA

Ce workflow est découpé en 4 phases principales, nous allons donc les décrire plus en détails.

Lors de la phase « ***Jeu de données*** » nous choisissons quelles technologies doivent être détectées, par exemple, la 4G et la 5G. C'est aussi lors de cette étape que nous choisissons la forme des données contenues dans le dataset. Par exemple, nous pouvons choisir de représenter les signaux sous leur forme spectrale et isoler des points autour du front montant. À la fin de cette étape, nous obtenons un dataset exploitable pour l'entrainement d'un modèle d'intelligence artificielle.

Lors de la phase « ***Création et entrainement*** », nous choisissons toutes les caractéristiques du modèle d'IA que nous voulons entrainer. Ces caractéristiques vont beaucoup affecter la structure des données d'entrée et la forme de la sortie. Par exemple, si nous choisissons un algorithme de Deep Learning, il faut concevoir la structure des neurones composant le modèle. Il faut ensuite choisir les

hyperparamètres du modèle, par exemple le nombre de cycles d'entrainement. Nous obtenons un modèle d'IA entrainé dont il faut maintenant évaluer les performances.

C'est donc lors des « ***Tests et mesures*** » que nous allons pouvoir estimer la qualité du modèle. Pour cela, nous appliquons un protocole de test avec une grande variété d'échantillons. Attention, pour éviter tout biais, il ne faut qu'aucun échantillon ne soit présent à la fois dans les datasets d'entrainement et dans ceux de test. Posséder des échantillons venant de différentes sources permet de pousser le modèle dans ses limites.

Au cours de la phase « ***Conversion et déploiement*** » le modèle d'intelligence artificielle est intégré à l'application finale. Il faut donc d'abord le sauvegarder avec toutes les variables utiles à son exploitation comme un seuil dans le cas de la détection de front montant OFDM. Il faut aussi créer ou adapter les fonctions permettant la mise en forme des données ou de traitement des résultats.

4.3. Génération de jeu de données

Le travail de génération de données va avoir un impact important sur la qualité de la prédiction des modèles d'intelligence artificielle. Il est donc nécessaire d'y accorder beaucoup de temps et d'importance. Dans le cadre du projet, une extraction de caractéristiques signantes des signaux 4G et 5G a été réalisée afin d'en simplifier la prédiction. Une connaissance fine de la donnée permet l'exploitation de modèle d'IA plus fiable et moins gourmand en ressource.

Dans le domaine temporel, la forme du signal était reconnaissable, avec une amplitude plus faible pendant la transmission des données de contrôle, mais il était difficile de savoir si elle était unique aux signaux OFDM. De plus, cette forme avait des risques d'être non reconnaissable sur des signaux bruités ou sous diverses autres conditions.

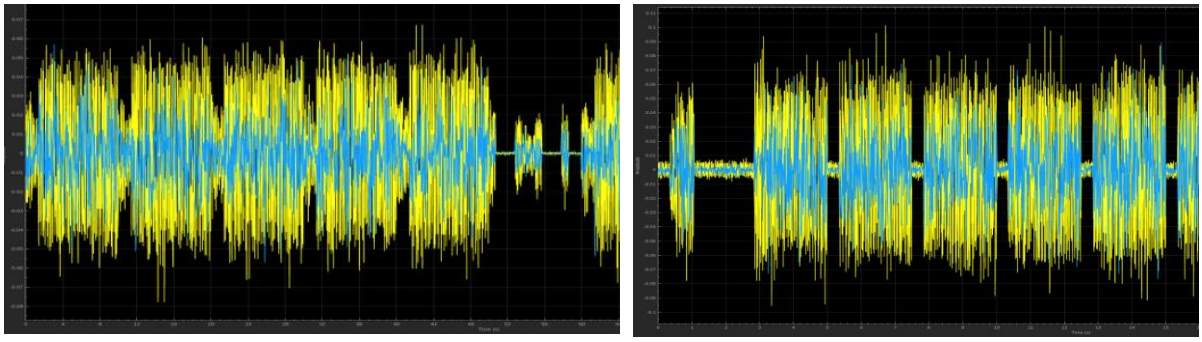


Figure 11 : Signaux temporel IQ 4G et 5G

La transformée SCF (Spectral Correlation Function) permet d'afficher les caractéristiques cyclo-stationnaires du signal étudié. Elle est donc particulièrement adaptée aux signaux de télécommunication. Au cours d'une pré-étude, il a été constaté que la 4G et la 5G se distinguaient clairement des autres signaux. Cependant, cette transformée demande une puissance de calcul trop importante (algorithme par accumulation de FFT), ce qui n'aurait pas permis de respecter la contrainte temporelle du scanner.

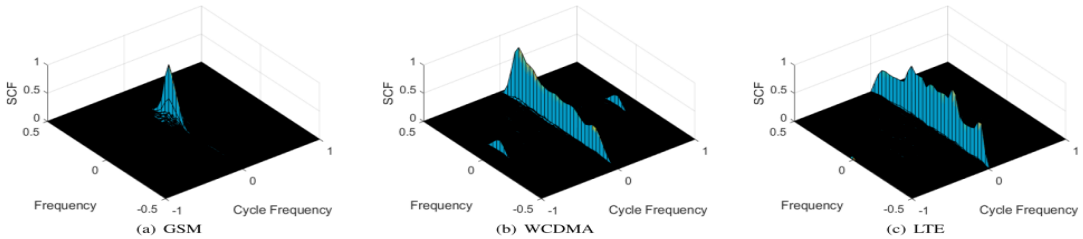


FIGURE 1. SCFs estimated by FAM algorithm in bi–frequency plane after mapping for various cellular communication signals.

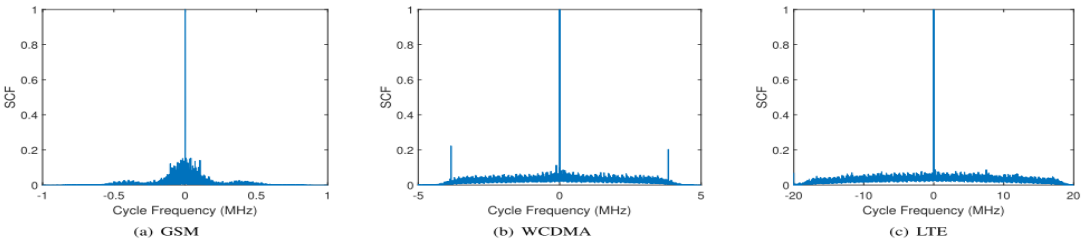


FIGURE 2. α–domain profiles evaluated by maximizing SCFs over spectral frequency.

Figure 12 : Transformée SCF pour la 2G, 3G et 4G

Dans le domaine fréquentiel, plusieurs éléments du signal sont reconnaissables à des instants précis, comme les signaux d'égalisation ou de contrôle. Cependant, cela requiert de se synchroniser correctement et très précisément, ce qui rendrait le développement du scanner trop compliqué. Quant aux fronts montants et aux DC Carriers, ils sont reconnaissables dans de très nombreuses instances de signaux 4G et 5G, c'est donc cette approche qui a été choisie.

Après avoir choisi la représentation fréquentielle pour l'entrée de l'intelligence artificielle, il a été nécessaire de déterminer le nombre de points qui seraient donnés. 100 points espacés de 5 kHz est un bon compromis entre résolution fréquentielle et largeur fréquentielle analysée. C'est donc ce qui a été choisi dans un premier temps. Avec les toolboxs MATLAB, des spectres de signaux 4G et 5G ont été générés, et l'influence des différents paramètres a été étudiée. Du côté de la cellule, les paramètres qui affectent principalement le signal sont la largeur de la bande, la modulation numérique, et l'espacement entre les sous-porteuses pour les signaux 5G. Du côté du canal de transmission, c'est principalement le niveau de bruit qui est une donnée maitrisée ayant une influence sur le signal. Il est aussi possible de faire varier différents autres paramètres du canal, mais ils ne sont pas maitrisés (connaissance des valeurs possibles dans le monde réel, probabilité de trouver un canal paramétré comme dans la simulation, …).

Environ 16400 signaux avec les différentes combinaisons de paramètres possibles ont été générés, et leurs fronts montants et leurs centres ont été extraits pour pouvoir générer un dataset. Après le premier entraînement du modèle d'analyse des fronts montants, il a été observé un phénomène de surapprentissage. Cela signifie que le modèle a obtenu de bons résultats uniquement sur les données d'entraînement, mais a eu du mal à généraliser et à obtenir de bons résultats sur de nouvelles données, ce qui a introduit un biais dans les résultats obtenus. Le nombre de signaux dans les datasets a alors été réduit à 1 000 pour ce modèle. Les résultats obtenus sur les datasets de test étaient alors bien meilleurs.

Lors de la création du modèle d'intelligence artificielle étudiant les fronts montants, il a été observé que le scanner ne capturait pas toujours correctement le front montant lorsqu'il était limité à 100 points. Par conséquent, le nombre de points a été augmenté à 150 afin d'augmenter la largeur fréquentielle analysée.

4.4. Reconnaissance de front montant OFDM

4.4.1. Mise en forme des échantillons

Comme mentionné dans la partie 2.4, la reconnaissance des cellules 4G et 5G se fait en 2 parties. Le scanner commence par séparer la 4G et la 5G des autres technologies avant de différencier la 4G de la 5G (voir 4.5). Dans cette démarche, l'objectif est de reconnaître la forme spectrale de l'OFDM, qui représente la modulation utilisée par les deux technologies.

Nous avons sélectionné le front montant de l'OFDM (forme spectrale) car il est très particulier comparé aux autres technologies que nous essayons d'écarter. En effet, les fronts montants de la 4G et la 5G sont très droits, on passe du plancher de bruit jusqu'au plafond de la cellule en quelques dizaines de kHz alors que la 3G par exemple met plusieurs centaines de kHz.

Voici un exemple de front montant OFDM et un front montant de 3G (UMTS) :

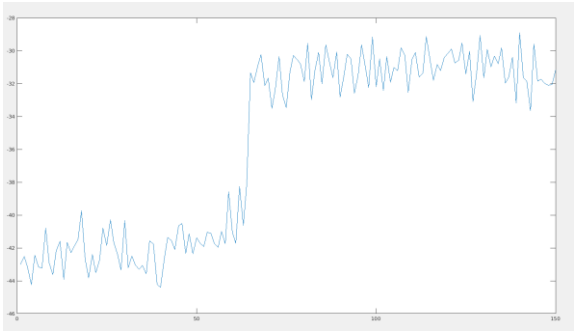


Figure 13 : Front montant 4G (OFDM)

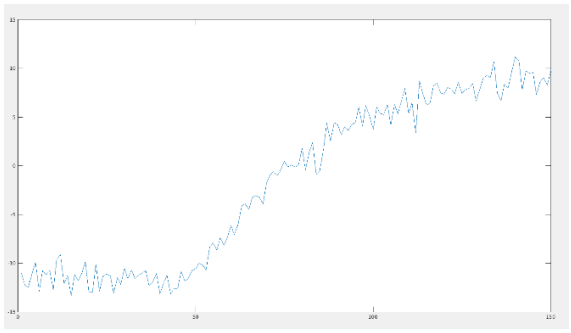


Figure 14 : Front montant 3G (UMTS)

Une grande partie de la performance des modèles vient de la qualité des données d'entrainement. La forme de ces données doit être choisie de sorte à faire ressortir les caractéristiques que l'on cherche à détecter et à réduire la variance non signante.

Une première solution est de normaliser simplement en changeant l'échelle de l'amplitude pour qu'elle soit entre 0 et 1. Le but étant d'éliminer l'information de la puissance reçue qui n'est pas signante. Pour cela, on utilise une fonction de la librairie sklearn : minmax_scale. Voici un exemple de front montant de 4G avant et après normalisation :

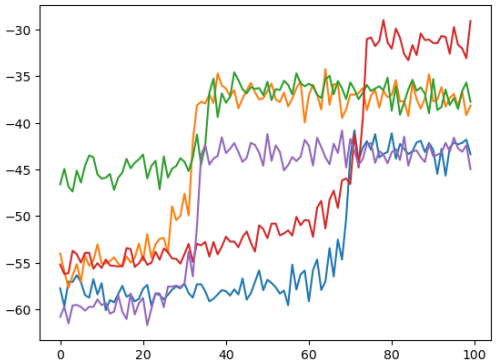


Figure 15 : Échantillons 4G avant normalisation

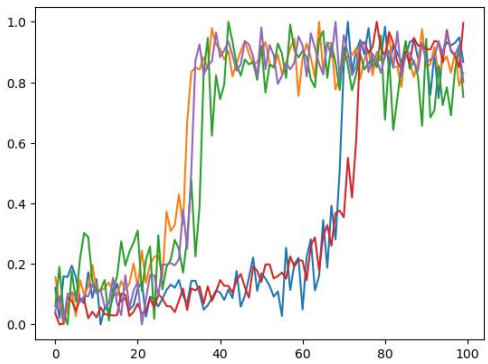


Figure 16 : Échantillons 4G après normalisation

L'effet de la normalisation sur un grand nombre d'échantillons (8400) peut aussi être visualisé à l'aide d'une heatmap. Voici donc des fronts montants 4G avant et après normalisation :

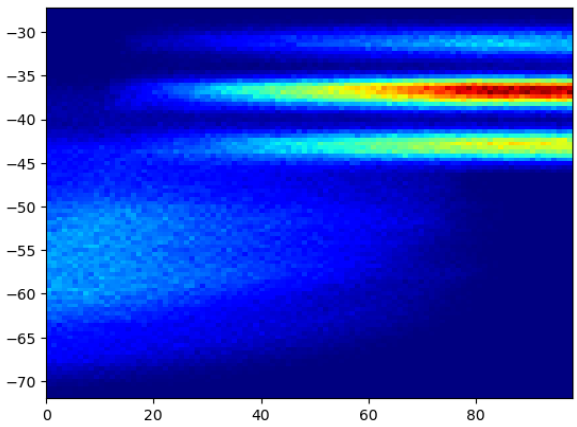


Figure 17 : Heatmap 4G avant normalisation

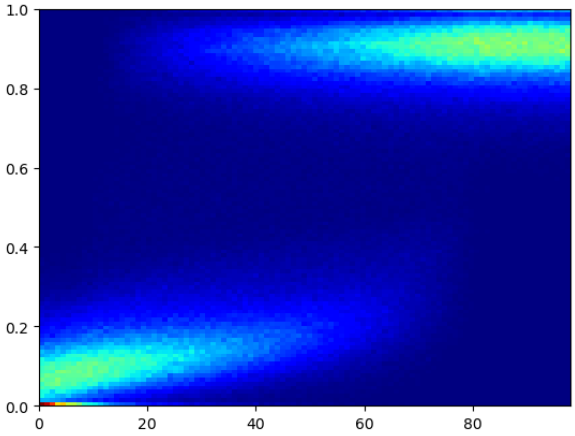


Figure 18 : Heatmap 4G après normalisation

L'uniformisation des échantillons par la normalisation est très significative. Le travail sur les différentes normalisations « classiques » est disponible dans le notebook nommé *« Normalisation Front Montant V1 »*. Il contient des visualisations d'échantillons de 2G, 3G, 4G et 5G normalisée avec différentes méthodes.

Sur les échantillons présentés plus haut, on remarque que la position du front montant varie. En effet, en sortie de la phase d'acquisition, la précision n'est pas assez bonne pour être sûrs que le front sera centré sur l'échantillon. Pour que cela ne crée pas un biais, une solution est d'ajouter cette variance sur les données d'entrainement. Le modèle comprendra donc que la position du front ne fait pas varier les résultats. Éliminer cette variance peut simplifier le problème est donc améliorer la prédiction. Une représentation des données qui élimine cette variation a été retenue, le format histogramme.

L'avantage de la représentation histogramme est qu'elle permet de mettre en avant les spécificités de l'OFDM (front montant très droit) par rapport aux autres technologies tout en supprimant la variation de position du front montant. De plus, on peut choisir un nombre de bins (nombre de colonnes de l'histogramme) constant peu importe la taille de l'échantillon d'entrée. Voici un exemple de fronts montants 4G et leur transformation en histogrammes :

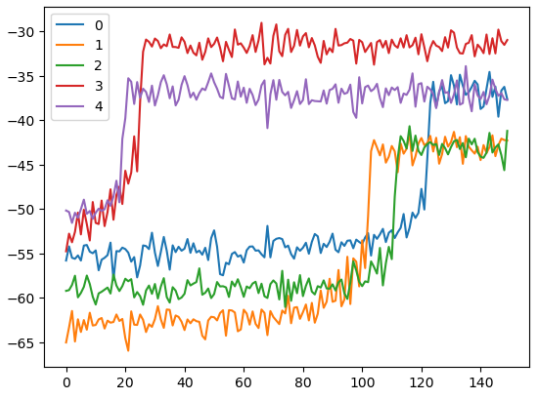


Figure 19 : Échantillons 4G avant normalisation

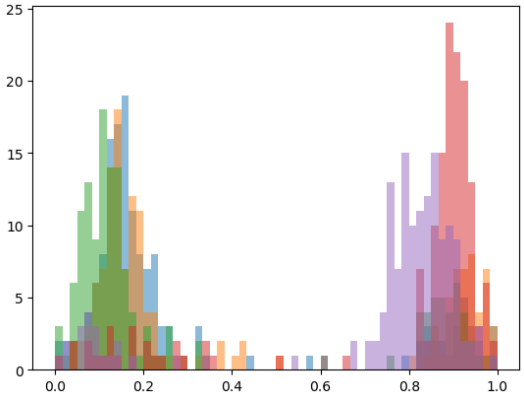
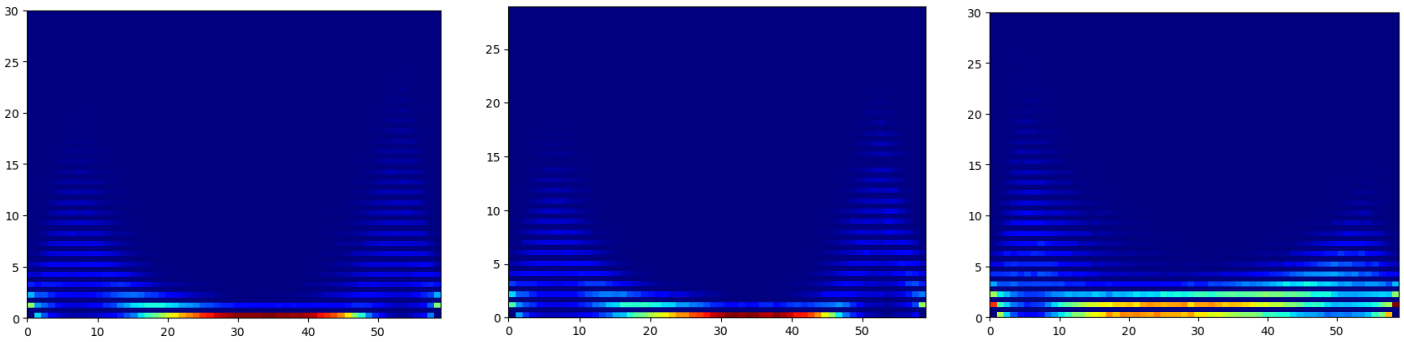


Figure 20 : Échantillons 4G après leur transformation en histogrammes

Il a été observé de manière constante que les bins centraux de l'histogramme sont généralement vides, indépendamment de l'échantillon. Par conséquent, seules ces parties de l'histogramme, contenant les bins centraux, sont fournies au modèle d'intelligence artificielle. Cette représentation permet aussi de réduire l'impact du bruit. En effet, le seul effet du bruit sera d'élargir les pics de gauche et droite sans trop impacter les bins centraux. Voici 3 heatmaps représentant respectivement des échantillons OFDM (4G et 5G) et UMTS (3G) :



4G (OFDM) *5G (OFDM)* *3G (UMTS)*

Sur les bins centraux (entre 20 et 40) on observe une nette différence entre l'OFDM et l'UMTS. Tout le travail sur la transformation en histogramme se trouve dans le notebook nommé *« Normalisation Front Montant V2 »*.

Ainsi, le problème a été considérablement simplifié pour l'intelligence artificielle. La simplification s'est effectuée d'une reconnaissance de pente sur 150 points, avec de nombreuses variations dues au bruit et aux décalages de pente, à un échantillon de 20 points où l'objectif est simplement de vérifier la présence d'une quantité excessive de données. En conséquence, des modèles très simples (~100 neurones) ont été créés, permettant un entraînement rapide.

4.4.2. Modèle Auto-Encodeur

Le principal problème de la reconnaissance de front montant OFDM dans notre projet est l'impossibilité de décrire de manière exhaustive tout ce qui n'est « pas OFDM ». En effet, un modèle classique de type « classifier » va s'entrainer tous les types de données qu'il doit classifier puis estimer à quelle classe appartiennent les nouveaux échantillons qu'on lui propose basé sur la ressemblance avec ceux qu'il connait. C'est pourquoi un modèle de classification classique ne peut pas être utilisé.

Après des recherches et consultation d'experts IA au sein de l'ESISAR, il a été décidé d'utiliser un modèle d'auto-encodeur en raison de sa capacité à être utilisé avec le principe de la détection d'anomalie plutôt que la classification. La différence entre ces 2 types de modèles est que l'on cherche à mesurer la différence entre les échantillons présentés et la base d'entrainement plutôt que la ressemblance.

L'auto-encodeur (dont un exemple d'architecture est représenté par la figure ci-dessous) est un type de réseau de neurones particulier pour principalement deux raisons :

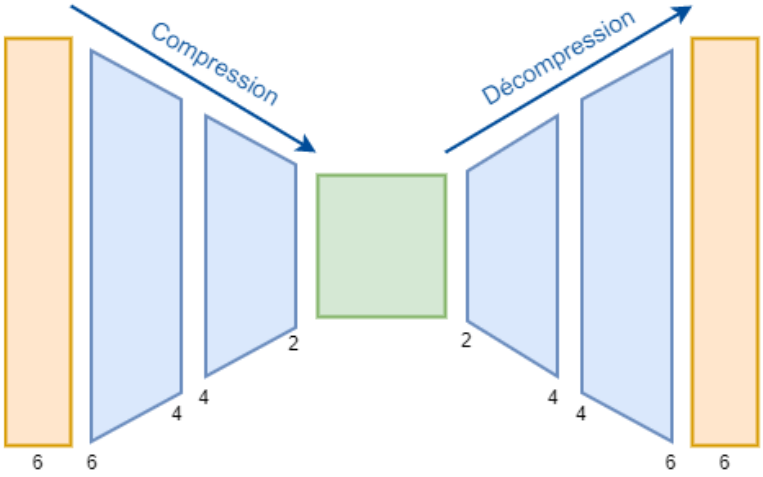


Figure 21 : Structure d'un auto-encodeur

Premièrement, il s'agit d'un algorithme d'apprentissage semi supervisé : seules les données dites « normales » (ici OFDM) sont utilisées pour l'apprentissage. En revanche, au moment de la validation, un mélange de données « normales » et « anormales » (ici UMTS) est requis pour vérifier la détection et observer le placement du seuil de décision.

Sa deuxième spécificité est qu'il utilise un principe simple, mais qui a fait ses preuves en informatique : la compression-décompression. Lorsqu'il est effectué correctement, ce processus permet de dégager les éléments essentiels d'une liste d'indicateurs. Cette astuce va permettre à l'algorithme de séparer efficacement les données qu'il connaît des données anormales.

Ce réseau de neurones est fragmenté en 3 parties :

- Un encodeur, réseau de neurones qui permet de réaliser la compression, constitué de couches dont le nombre de neurones décroît. Cette partie transforme les indicateurs initiaux en sous-ensemble plus significatif.
- Un goulet d'étranglement, couche détenant le moins de neurones du réseau servant de transition, et de centre au réseau global.
- Un décodeur, réalisant l'opération inverse de l'encodeur ; il prend en entrée le goulet d'étranglement (peu d'indicateurs) et le transforme en sortie plus importante. Le décodage, à l'instar de l'encodage permet, elle aussi, d'ajouter une couche de « complexité » au procédé, qui devient à même de mieux séparer les données similaires à celles utilisées pour l'entraînement des autres.

La première couche et la dernière couche contiennent obligatoirement le même nombre de neurones, sans quoi il ne pourrait remplir son objectif de reconstruction. En effet, son rôle est de créer une image la plus fidèle possible de l'entrée du réseau (même si l'on ne peut pas parler de neurones jumeaux, i.e. comme si chaque neurone d'entrée avait sa sortie dédiée). Cette représentation est fidèle du point de vue de l'erreur des moindres carrés.

Ainsi, l'auto-encodeur, qui a une structure optimisée pour dégrader et reconstruire une information, va **très bien savoir comment compresser et décompresser les données normales, mais très mal d'autres types de données.**

En résumé, la sortie sera très similaire à l'entrée si les données sont dites « normales » mais très différentes si les données sont dites « anormales ». Il ne reste plus qu'à calculer la différence entre l'entrée et la sortie du réseau pour obtenir l'erreur de reconstruction.

Il faut ensuite choisir un seuil de décision qui permettra de classifier les échantillons comme des anomalies ou non. Ce seuil doit être déterminé à chaque nouveau modèle, car il est très sensible et dépend énormément de l'entrainement de chaque modèle. Pour trouver une valeur satisfaisante, on se base donc sur l'erreur moyenne que fait le modèle sur les échantillons normaux qu'il a rencontrés lors de son apprentissage auquel on peut ajouter plusieurs écarts types. Après de nombreux essais, nous avons trouvé qu'un seuil fixé à 1 fois la moyenne plus 2 fois l'écart type permettait de bonnes performances et un bon équilibre entre recall et précision. Pour rappel, le recall (ou rappel) et la précision sont deux notions permettant de mesurer les performances d'un modèle. La précision est la proportion d'échantillons réellement pertinents parmi l'ensemble des échantillons désignés comme pertinent. Le recall est la proportion d'échantillons désignés comme pertinent parmi l'ensemble des échantillons pertinent.

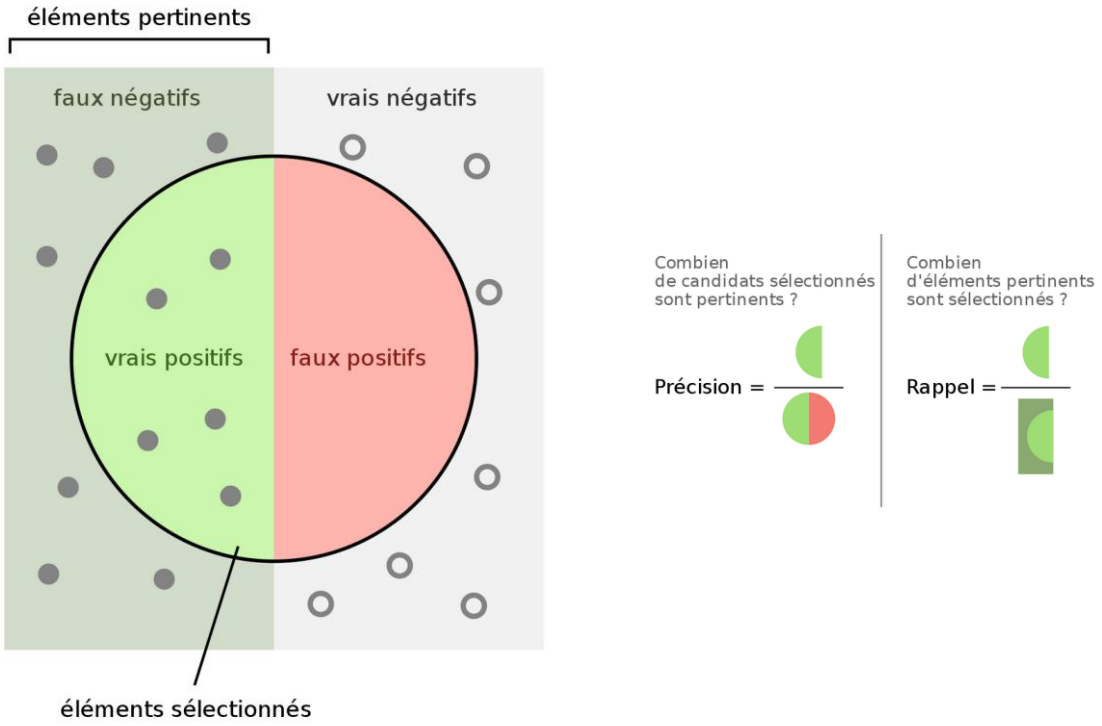


Figure 22 : Explication recall et précision

Nous avons utilisé notre workflow pour améliorer petit à petit nos modèles. Tout ce travail est disponible dans les différentes versions des notebooks *« Front montant OFDM - AutoEncodeur -VX »*. Dans les versions 1 et 2 nous n'utilisions pas la représentation histogramme et l'on peut voir que les résultats sont inexploitables. À partir de la version 3 on introduit cette représentation et l'on observe enfin des résultats utilisables. Les dernières versions utilisent des datasets améliorés et corrigés.

Les versions finales de ces itérations de workflows se trouvent dans les notebooks *« Entrainement modèle recette VX »*. Ils ne contiennent que l'essentiel pour être plus court et donc plus agréable à lire. La version 2 introduit les échantillons à 150 points, car nous nous sommes rendu compte que les scanner n'était pas assez précis pour être sûr qu'un échantillon de 100 points (500kHz de large) contiennent le front montant de la cellule. La version 3 introduit une nouvelle amélioration, au lieu de prendre les 20 bins centraux (sur 60 bins au total) on prend les bins de 30 à 40 seulement. Cette modification a été motivée par l'observation plus poussée d'échantillons réels dans lesquels nous avons remarqué que le départ du front était beaucoup moins raide que la fin. Ça avait pour effet d'ajouter de la donnée dans les bins 20 à 30 qui n'étaient pas ou peu présente dans les datasets générés par MATLAB. C'est un modèle généré par le notebook V3 qui a été utilisé lors de la recette.

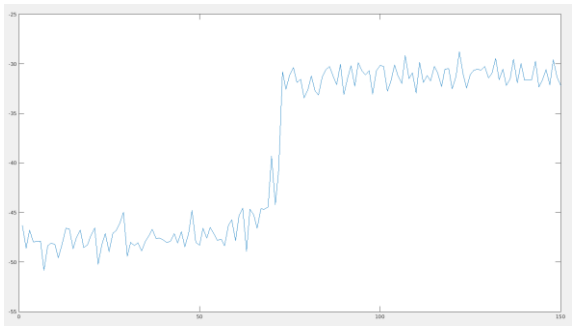


Figure 23 : Front montant généré par MATLAB

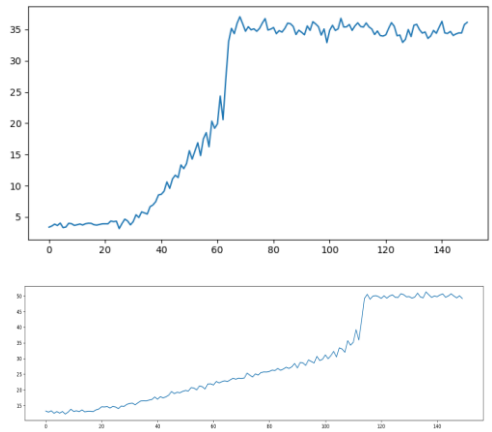


Figure 24 : Fronts montants réels

4.4.3. Résultats & Limites

Les résultats obtenus avec différents tests vont maintenant être observés. Tout d'abord, voici des graphiques montrant les erreurs de reconstructions du modèle sur les différentes technologies ainsi que la matrice de confusion et les statistiques correspondants :

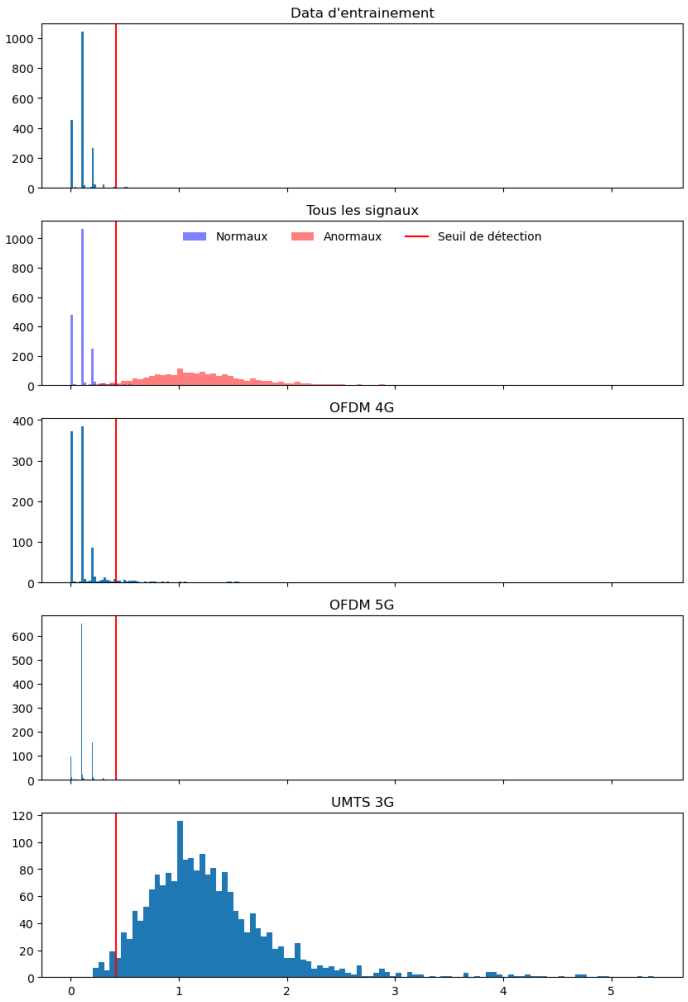
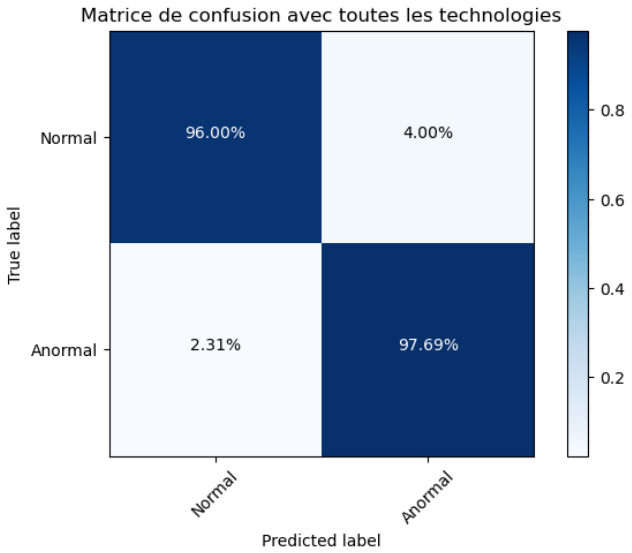


Figure 25 : Histogrammes de l'erreur de reconstruction du modèle sur différentes technologies




```
Accuracy = 0.9683544303797469
Precision = 0.9596977329974811
Recall = 0.9769230769230769
F1-score = 0.9682337992376111

Accuracy par technologies :
- Accuracy OFDM 4G = 0.926
- Accuracy OFDM 5G = 0.994
- Accuracy UMTS = 0.9769230769230769
```

Figure 26 : Matrice de confusion et statistiques du modèle

Le seuil de décision est déterminé à partir des données d'entrainement (histogramme du haut). L'axe des abscisses représente l'erreur de reconstruction et l'axe des ordonnées représente le nombre d'échantillons présentant cette erreur. Les échantillons 4G et 5G sont bien reconnus comme étant normaux, alors que les échantillons 3G sont majoritairement désignés comme étant des anomalies.

La matrice de confusion rend compte de ces bons résultats. Les cases foncées de la diagonale représentent les vrais positif (en haut à gauche) et négatif (en bas à droite), c'est-à-dire les bonnes prédictions, on a une *accuracy* de 96.8%. Pour rappel, l'accuracy permet de mesurer la proportion de bonnes prédictions par rapport au nombre total de prédictions. Les 2 cases claires représentent, elles, les faux positifs (en bas à gauche) et les faux négatifs (en haut à droite). Les faux positifs sont les prédictions les plus gênantes, car elles assurent que le modèle de classification de technologie (voir 4.5) donnera une fausse prédiction puisque l'échantillon n'est pas de l'OFDM (donc pas 4G ni 5G).

Ces résultats permettent une bonne première idée de la qualité de l'entrainement du modèle. Ils ont systématiquement été utilisés durant le projet après chaque entrainement.

Nous avons aussi étudié l'impact du rapport signal / bruit (SNR) sur les prédictions des modèles. Le but n'étant pas d'obtenir une valeur précise de seuil de SNR mais de confirmer la tendance selon laquelle un SNR élevé amène à de meilleurs résultats.

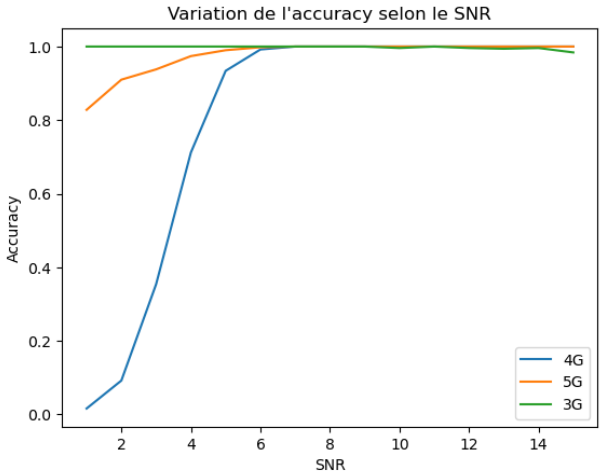


Figure 27 : Evolution des bonnes prédictions en fonction du SNR

Ce graphique montre en effet que les résultats s'améliorent lorsque le SNR qui augmente. Les résultats avec les échantillons 3G ne sont pas impactés par cette variation.

Le travail sur l'évaluation en fonction du SNR se trouve dans le notebook *« Front montant selon SNR V1 »* et surtout dans le notebook *« Test MATLAB – SNR »* de la recette.

Les modèles ont également été testés sur des échantillons réels de 2 sources différentes. Premièrement, nous avons extrait des fronts montants de cellules 4G, 5G et 3G sur des acquisitions faites par Avantix en mars. Nous avons appliqué un décalage sur le front pour multiplier les échantillons et ajouter de la difficulté de prédiction. Voici les résultats que nous obtenons :

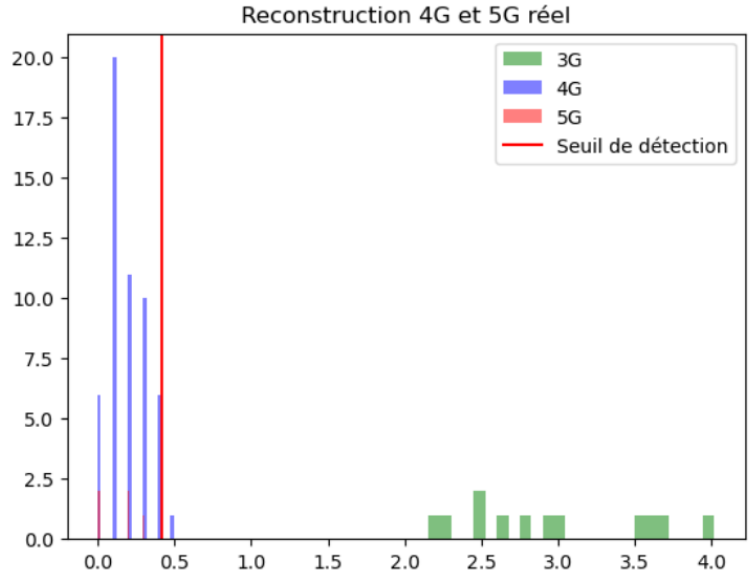


Figure 28 : Résultats échantillons réels Avantix

La 3G est très bien reconnue avec une grosse marge. Les résultats sur la 5G sont assez peu représentatifs, car il n'y a que 5 échantillons et globalement, on peut dire que la 4G est plutôt bien reconnue aussi.

Dans un second temps, nous avons étudié des échantillons réels produit par le scanner. Ils nous ont permis de détecter une faiblesse dans le modèle.

En effet, le choix d'utiliser une représentation sous forme d'histogramme introduit un biais. Certains échantillons qui, sous leur forme spectrale, ne ressemblent pas du tout à des fronts montants OFDM, ont une signature histogramme très proche de la 4G ou de la 5G. Voici quelques exemples :

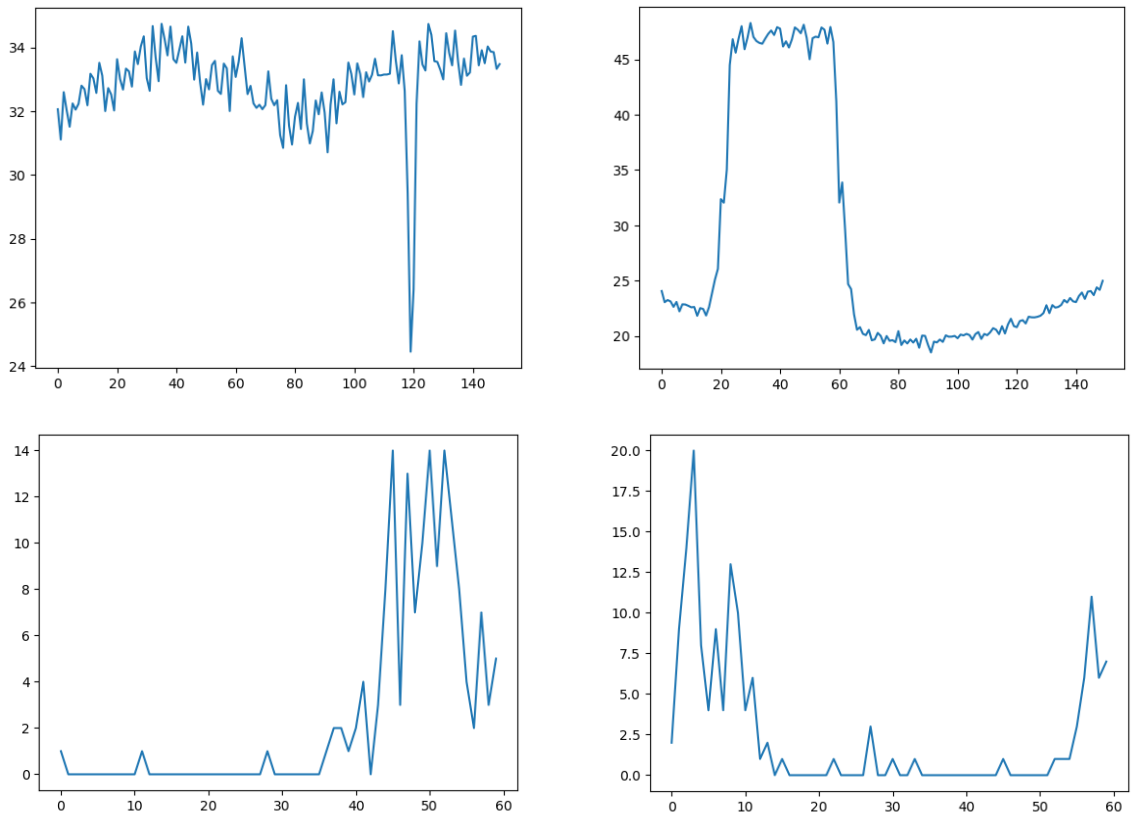


Figure 29 : Échantillons problématiques et leur histogramme

Une autre limitation est que certains échantillons 3G avec des pentes trop raides sont reconnus comme étant OFDM. Voici un exemple :

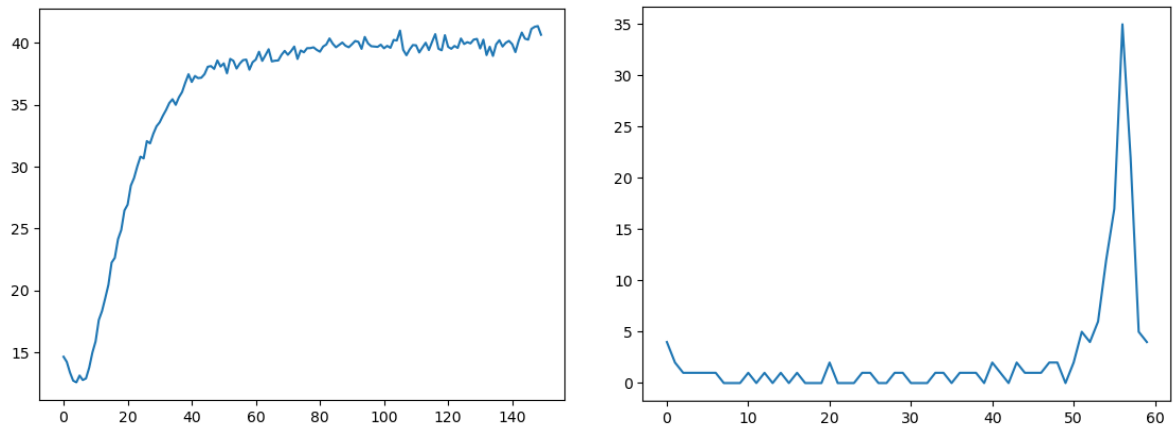


Figure 30 : Échantillon 3G problématique et son histogramme

En conclusion, les résultats obtenus sur les échantillons générés par MATLAB sont très bons, cependant, une fois que nous passons aux échantillons réels, certaines anomalies se manifestent. Si

suffisamment d'échantillons réels étaient disponibles pour entraîner les modèles, cela ne ferait qu'améliorer les résultats obtenus. Toutes les perspectives d'amélioration sont décrites en détail dans la partie 0.

4.5. Classification de technologie 4G & 5G

4.5.1. Mise en forme des échantillons

Comme vu dans la section 2.3, la représentation fréquentielle des signaux permet de faire apparaitre une variation entre les centres des spectres des cellules. C'est donc la transformée de fourrier qui a été retenue pour transformer les signaux temporels acquis par l'USRP. Cette transformation a l'avantage d'être rapide et peu gourmande en puissance de calcul.

Le jeu de données d'entrainement, généré par Matlab, contient 16464 échantillons dont 8400 de technologie 4G et 8064 échantillons de technologie 5G. Chaque échantillon contient 100 points espacés de 5 kHz chacun. Par conséquent, la largeur de la bande analysée est de 500 kHz. Cette largeur est nécessaire afin d'être assuré de récupérer le centre de la cellule lors de l'estimation de la fréquence centrale par le scanner. Si la largeur n'est pas suffisante et que l'estimation n'est pas assez précise alors une mauvaise classification pourrait être effectuée du fait que le centre de la cellule ne soit pas présent dans l'échantillon analysé. L'espacement fréquentiel fixe entre chaque point est donc très importante pour limiter le biais et améliorer les prédictions.

Le dataset d'entrainement prend en compte ce potentiel décentrage du « drop » de puissance dans l'échantillon analysé en effectuant un décalage aléatoire de la fréquence centrale. Ainsi le modèle ne se basera pas sur la position de la chute de puissance pour effectuer sa classification ce qui réduit le biais entre les échantillons générés et ceux observés.

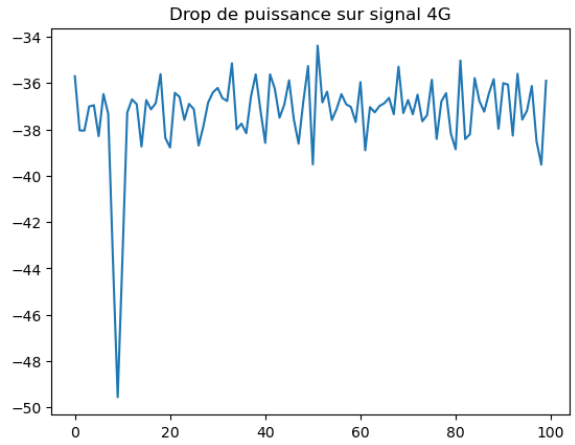


Figure 31 : Echantillon 4G centré non normalisé

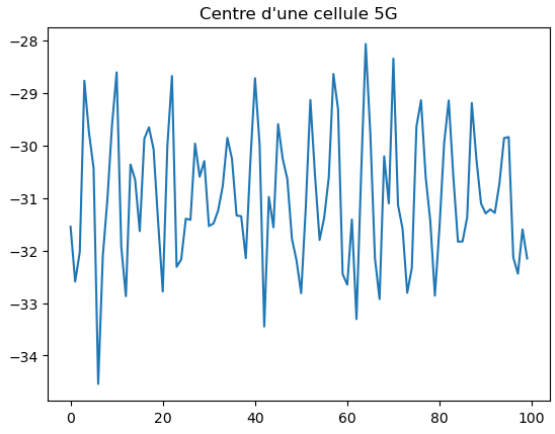


Figure 32 : Echantillon 5G centré non normalisé

De plus, pour limiter le biais dans le jeu de données et dans les acquisitions, il convient de normaliser les données sur l'amplitude. Pour cela, il existe différents types de normalisation. Dans le notebook nommée « *Classification4G&5G / ProcessingDataset4gAnd5g.ipynb* », différents types de normalisation sont passés en revue.

La normalisation MinMax, qui rétablit l'amplitude des points entre 0 et 1, permet d'obtenir la meilleure distinction entre les 2 technologies. En fonction des technologies, les histogrammes suivants sont obtenus :

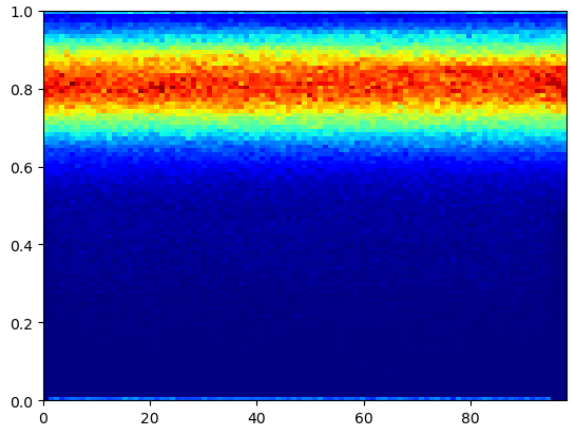


Figure 33 : Histogramme du dataset 4G normalisé

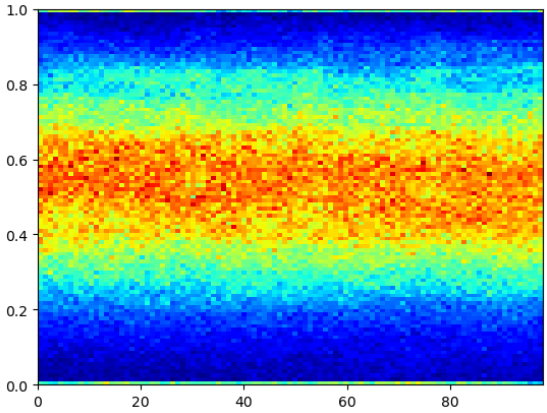


Figure 34 : Histogramme du dataset 5G normalisé

La normalisation permet de recentrer et de relever le plafond de bruit, à 0,8 d'amplitude, en 4G, étant donné que la chute de puissance va décaler l'échelle. En 5G, le plafond va être centré au niveau du centre de l'histogramme, à 0,5 d'amplitude.

4.5.2. Choix de l'algorithme

Le choix entre un algorithme de machine learning et de deep learning dépend généralement de la nature du problème et des données disponibles. Dans le contexte de la reconnaissance d'une chute de puissance dans un échantillon fréquentiel, les raisons qui ont fait que nous avons utilisé un algorithme de machine learning plutôt qu'un algorithme de deep learning sont :

Disponibilité des données : Les algorithmes de deep learning nécessitent souvent une grande quantité de données pour obtenir des performances optimales. Dans notre cas, les échantillons étant générés par l'équipe projet, nous n'étions pas sûr d'être en capacité d'obtenir assez d'échantillon pour obtenir de bonnes prédictions avec un réseau de neurone sans obtenir un surapprentissage sur le dataset.

Complexité du modèle : Le problème de reconnaissance de chute de puissance étant un problème accessible, il peut être résolu efficacement avec un modèle relativement simple, l'utilisation d'un réseau de neurones profonds pourrait être une complication inutile de par la multitude paramètre qu'ils proposent.

Interprétabilité des résultats : Les algorithmes de machine learning traditionnels tels que les arbres de décision, les SVM (Support Vector Machines) ou les réseaux neuronaux peu profonds sont souvent plus faciles à interpréter que les réseaux de neurones profonds utilisés dans le deep learning.

Temps d'entraînement : Les réseaux de neurones profonds, en particulier lorsqu'ils sont entraînés sur de grandes quantités de données, peuvent nécessiter un temps d'entraînement et une puissance de calcul considérablement plus élevé que les algorithmes de machine learning. Le projet étant cadré dans le temps, la phase de tuning des hyperparamètres aurait pu s'avérer chronophage au vu de la multitude de paramètres modifiables.

Les caractéristiques du dataset d'apprentissage nous ont influencé dans le choix des algorithmes de classification utilisé. Ci-dessous, une présentation rapide des algorithmes sur lesquels notre travail s'est concentré :

La régression logistique est un algorithme d'apprentissage automatique utilisé pour la classification binaire, où l'objectif est de prédire l'appartenance à une des deux classes possibles. La régression logistique est relativement rapide à entraîner et à prédire, même sur de grands ensembles de données. Elle est moins sensible aux données aberrantes par rapport à d'autres algorithmes et est plus adaptée aux données linéaires.

Le SVM (Support Vector Machine) est particulièrement efficace dans les cas où les données présentent une séparation linéaire claire entre les classes ou lorsqu'il est utilisé avec des noyaux non linéaires pour des problèmes de classification non linéaires. L'objectif du SVM est de trouver un hyperplan dans un espace de dimensions supérieures (ou le cas échéant, un hyperplan linéaire dans l'espace d'origine) qui sépare de manière optimale les exemples de différentes classes. Les échantillons les plus proches de l'hyperplan sont appelés les vecteurs de support, d'où le nom "Support Vector Machine". Le SVM présente plusieurs avantages, notamment sa capacité à gérer des ensembles de données de petite à moyenne taille, sa robustesse face aux données aberrantes et sa flexibilité grâce à l'utilisation de différents noyaux. Cependant, il peut être sensible au choix des paramètres, ce qui rend important la phase de tuning, et il peut être moins adapté aux ensembles de données de grande dimension.

Le Decision Tree Classifier (classifieur arbre de décision) est basé sur la création d'un arbre de décision qui permet de prendre des décisions basées sur des règles conditionnelles. Cet arbre de décision est construit à partir des caractéristiques des données d'entraînement et des labels de classes correspondantes. Les avantages du Decision Tree Classifier incluent sa simplicité conceptuelle, sa facilité d'interprétation, et sa capacité à capturer des relations non linéaires dans les données. De plus, il peut gérer des données manquantes et des attributs catégoriels. Cependant, les arbres de décision ont également quelques limitations. Ils ont tendance à être sensibles aux variations mineures des données d'entraînement, ce qui peut conduire à un sur-ajustement (overfitting). De plus, les arbres de décision peuvent avoir du mal à capturer des relations complexes dans les données et peuvent être moins performants que d'autres algorithmes lorsque la tâche de classification est plus complexe.

Il existe des variantes du Decision Tree Classifier, comme les Random Forest qui combinent plusieurs arbres de décision pour créer une forêt d'arbres et ainsi améliorer les performances prédictives. Nous étudierons donc ces modèles qui offrent de la robustesse en étant moins sensibles aux variations des données d'entraînement par rapport à un seul arbre de décision. Cela permet de réduire l'overfitting et d'améliorer la généralisation par rapport aux arbres de décision classiques.

XGBoost (eXtreme Gradient Boosting) utilise l'approche du boosting, qui est une méthode d'ensemble où les modèles sont construits de manière séquentielle. Chaque modèle successif est entraîné pour corriger les erreurs du modèle précédent, en se concentrant sur les exemples qui ont été prédits de manière incorrecte. XGBoost utilise des arbres de décision comme modèles de base. Les arbres de décision sont construits itérativement, en ajoutant des arbres à l'ensemble existant. XGBoost est reconnu pour sa performance élevée, grâce à son optimisation efficace et à sa parallélisation. Les techniques de régularisation intégrées dans XGBoost permettent de contrôler l'overfitting et d'améliorer la généralisation du modèle. Cependant, XGBoost peut être plus complexe à mettre en œuvre et à ajuster correctement par rapport à certains autres algorithmes. Il peut également être plus sensible à un mauvais réglage des hyperparamètres, ce qui peut nécessiter plus d'optimisation.

4.5.3. Tuning

La recherche d'hyperparamètres optimaux est une étape importante dans le processus de construction et d'optimisation des modèles de classification par intelligence artificielle. L'objectif de cette recherche d'hyperparamètres est d'obtenir un modèle qui généralise bien sur de nouvelles données et qui donne les meilleures performances possibles pour améliorer sa capacité de prédiction. Cette phase est appelée le tuning. Pour y parvenir nous utiliserons la méthode de recherche en grille (GridSearch) qui évalue le modèle sur différentes combinaisons prédéfinies d'hyperparamètres. Elle fonctionne de la manière suivante :

1. Définition de la grille d'hyperparamètres. Les hyperparamètres peuvent être différents en fonction des modèles à optimiser. Pour chaque hyperparamètre à ajuster, une liste de valeurs est spécifiée.

2. Construction des combinaisons possibles d'hyperparamètres générées à partir de la grille d'hyperparamètres.

3. Entraînement et évaluation : Pour chaque combinaison d'hyperparamètres, le modèle est entraîné sur les données d'entraînement et évalué par validation croisée (k-fold). L'ensemble de données est divisé en k sous-ensembles de taille similaire et le modèle est entraîné et évalué k fois en utilisant différents plis comme ensemble de validation à chaque itération. Cela signifie que chaque pli sera utilisé une fois comme ensemble de validation, tandis que les autres plis sont utilisés pour l'entraînement.

4. Sélection des meilleurs hyperparamètres : La combinaison d'hyperparamètres qui donne les meilleures performances est sélectionnée comme étant la configuration optimale.

Pour les problèmes où l'espace des hyperparamètres est très grand, la recherche d'hyperparamètres optimaux peut également être effectuée en utilisant d'autres méthodes, telles que la recherche aléatoire (RandomSearch) ou la recherche bayésienne (BayesianSearch).

Cette phase de tuning a été réalisée dans le notebook *'Classification4G&5G/Tuning.ipynb'*. Cela peut s'avérer assez long de parcourir l'ensemble de l'espace des hyperparamètres étant donné que les modèles sont entrainés autant de fois qu'il y a de combinaisons multipliées par le nombre de plis du k-fold. De plus, il est peu probable que la qualité de la prédiction soit fondamentalement transformée, et les gains obtenus seront au mieux de quelques pourcents. Dans certains cas, la configuration optimale s'avère être la configuration initiale.

Dans notre cas, il est difficile de valider le choix de ces hyperparamètres étant donné que les tests effectués lors de cette phase de tuning ne sont pas sur des données réelles par manque d'échantillons.

4.5.4. Tests

La phase de test de ces modèles s'effectuera en 3 parties dont la difficulté est grandissante. L'objectif est de faire ressortir le modèle le plus robuste et performant qui sera par la suite intégré dans le scanner. La première sera réalisée avec des données générées par Matlab afin de tester la résistance au bruit des modèles avec 15 niveaux de SNR. La deuxième sera avec des échantillons générés par une plateforme Amarisoft dont l'environnement radiofréquence est maitrisé de même que la configuration de la cellule émettrice. La dernière phase de test s'effectuera sur des données réelles dans les environnements radiofréquence de l'ESISAR et d'AVANTIX.

La phase de test sur des données Amarisoft n'a pas pu se réaliser pour cause de problème de génération et d'utilisation de la plateforme.

Ces phases de test se retrouvent en partie dans la recette. Les autres tests sont effectués en continu et se retrouvent en partie dans le notebook *'Classification4G&5G/TestClassifier.ipynb'*.

Lors des résultats, les modèles qui ressortent sont la SVM et le Random Forest. Ci-dessous, on retrouve les graphiques des bonnes prédictions en fonction du SNR avec des données Matlab :

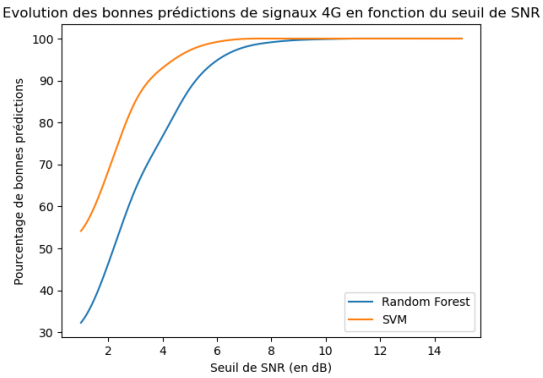


Figure 35 : Graphique des bonnes prédictions en fonction du SNR en 4G

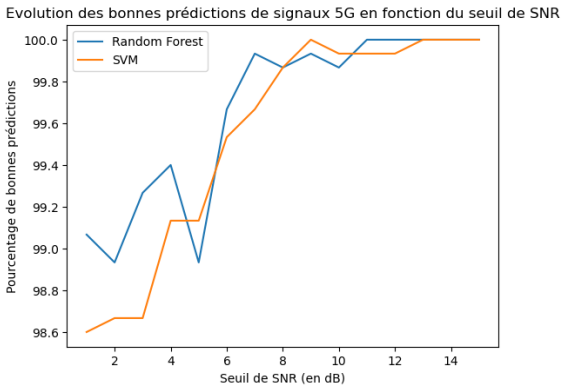


Figure 36 : Graphique des bonnes prédictions en fonction du SNR en 5G

On remarque que les 2 algorithmes auront tendance à effectuer une mauvaise prédiction en 4G lorsque le SNR baisse. Cela s'explique facilement du fait que le drop de puissance sera noyé et confondu avec le bruit donc l'acquisition sera confondue avec une acquisition 5G.

Lors du passage aux tests sur données réelles, l'algorithme de Random Forest obtient des résultats légèrement supérieurs. C'est pourquoi nous choisissons celui-ci pour l'intégration.

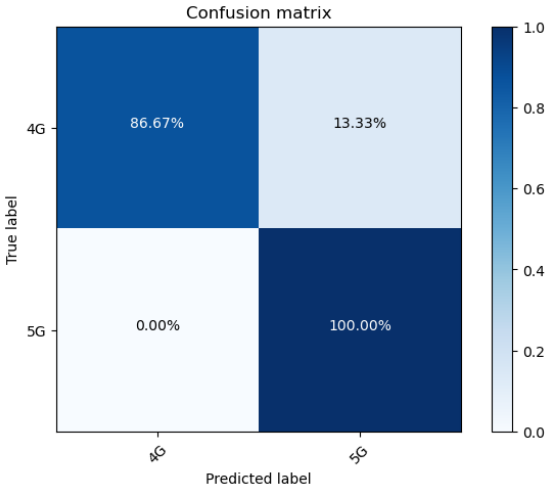


Figure 37 : Prédictions sur données réelles par SVM

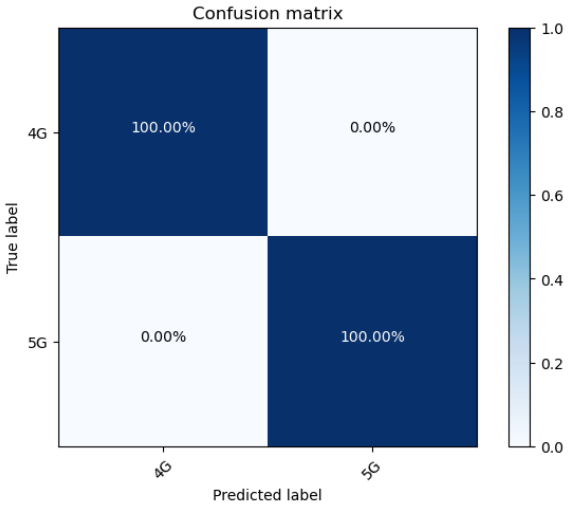


Figure 38 : Prédictions sur données réelles par RFC

5. Intégration

Après le développement unitaire de chacun des blocs logiciels décrit ci-dessus, ils ont tous été regroupés afin de former le produit final du projet : le scanner.

La base du scanner étant le bloc d'acquisition et de filtrage des cellules, les 2 blocs d'IA lui ont été greffés. Pour mieux appréhender l'utilisation de ce scanner, se référer au manuel utilisateur. Il contient toutes les indications pour paramétrer et lancer le scanner. Le contenu du fichier de sortie y est aussi décrit en détail.

Pour tester le scanner lors du développement et aussi mieux observer les résultats lors de la recette, un affichage complet des acquisitions ainsi que le résultat détaillé des modèles d'IA apparaissent. Le spectre de la bande scannée avec en surbrillance les cellules détectées est affiché. Les couleurs des cellules correspondent aux prédictions de l'IA.

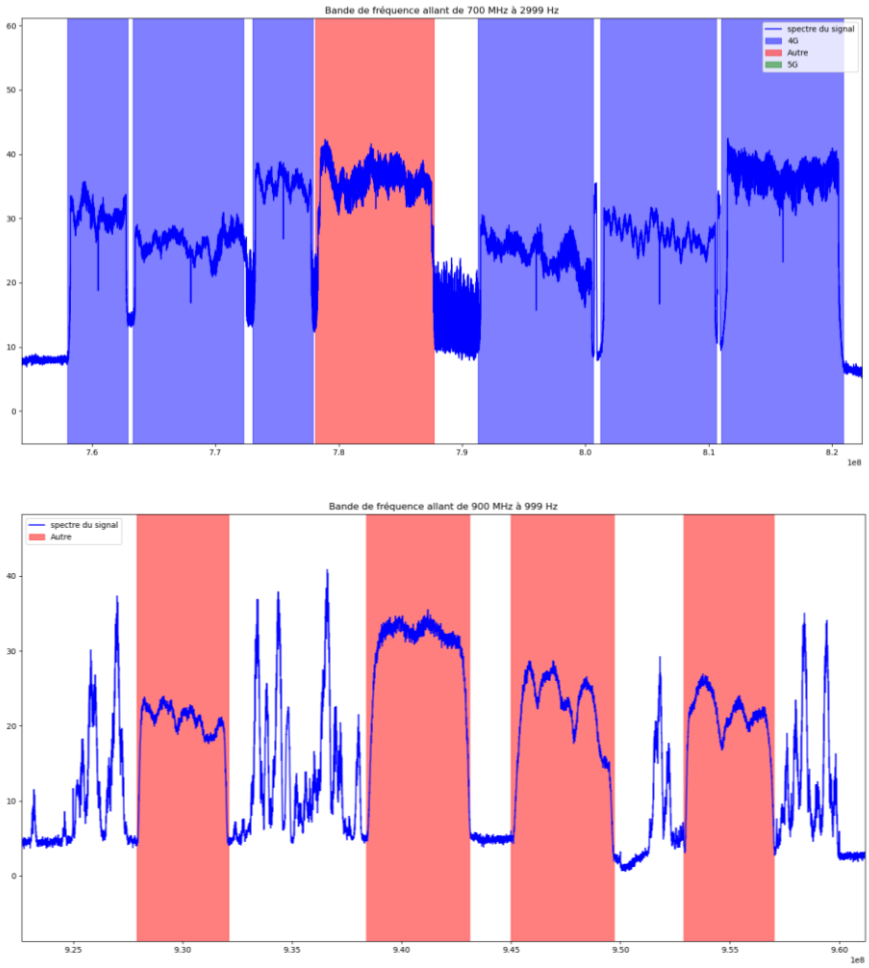


Figure 39 : Exemples d'affichages de 2 bandes contenant des cellules 4G et 3G

Pour plus de précision et de possibilités d'analyse, chaque cellule détectée est affichée dans le détail avec un zoom sur le front montant, et, si la cellule est classifiée OFDM, un zoom sur le drop central.

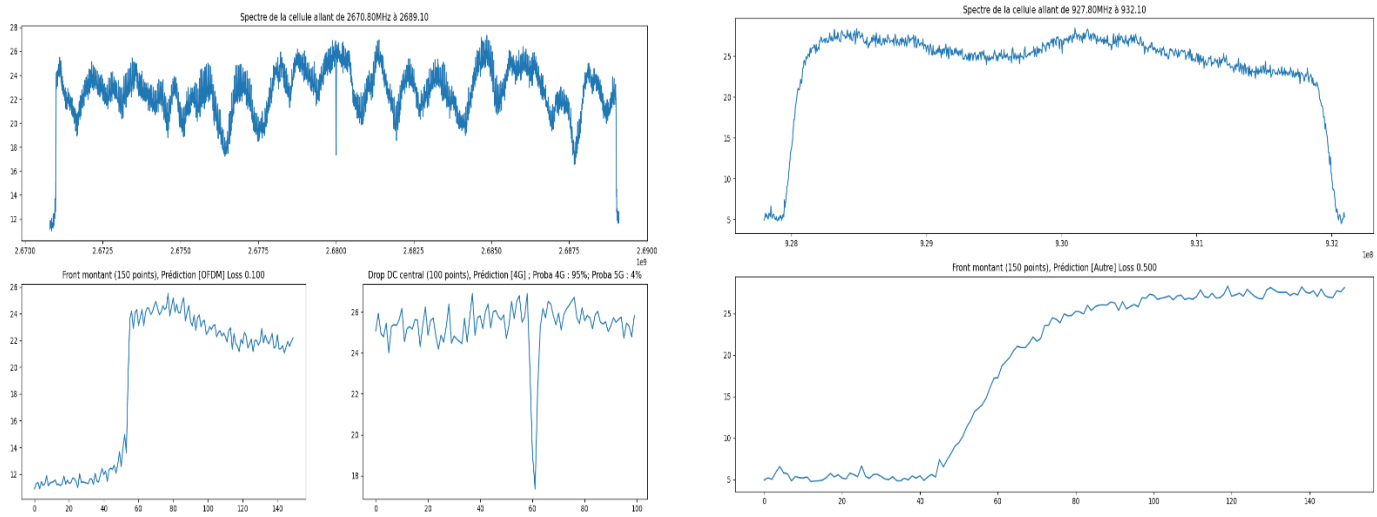


Figure 40 : Exemples d'affichage détaillé d'une cellule 4G et 3G

Grâce à ces différents affichages, nous pourrons visualiser et analyser tous les résultats lors de la recette. Nous pourrons ainsi mieux comprendre les potentielles erreurs du scanner et juger de leur gravité.

6. Perspective d'amélioration

6.1. Scanner

Lors de l'observation du spectre du signal étudié, il est remarqué qu'à haute fréquence, un schéma se répète périodiquement sur les fenêtres d'acquisition. En effet, une chute d'amplitude sur les bordures de la fenêtre ainsi qu'un pic d'amplitude au centre de la fenêtre est observé. Ce pic au point central est un artefact commun dans les récepteurs à conversion directe qui est l'architecture utilisée pour de nombreux USRPs. Dans les récepteurs à conversion directe, un oscillateur local convertit le signal de sa fréquence réelle en bande de base. Par conséquent, les fuites de cet oscillateur apparaissent au centre de la bande passante observée. La fuite de l'oscillateur local est une énergie supplémentaire créée à cause de la combinaison des fréquences. L'élimination de ce bruit supplémentaire est difficile car il est proche du signal de sortie souhaité. De nombreux circuits intégrés RF (RFIC) intègrent une fonction automatique d'élimination du décalage continu.

6.2. IA – Reconnaissance de fronts montants OFDM

Au fil du projet, nous avons fait beaucoup de recherches et avons demandé conseils à différents experts IA leurs idées pour le projet. Différentes solutions ont été proposés mais nous avons dû, par manque de temps, faire des choix sur les types d'algorithmes explorés. Maintenant que nous avons plus de recul et d'expérience sur l'IA dans les radiofréquences, nous pouvons donner un avis objectif sur des solutions que nous n'avons pas choisi d'étudier et essayer d'en imaginer les avantages et inconvénient.

6.2.1. Algorithmes SVM

La SVM est un des algorithmes utilisés pour le classificateur de technologie (voir 4.5). On pourrait également envisager d'entrainer une SVM sur les datasets de front montant 3G, 4G et 5G. Les résultats sur les échantillons générés par MATLAB seront probablement très bons, mais la prédiction sera surement plus nuancée lors de la généralisation avec des configurations non présentes dans les datasets d'entrainement. L'indicateur de confiance des modèles pourrait être utilisé pour essayer de mieux classifier les échantillons inconnus.

6.2.2. Cartographie de l'espace latent

Dans la structure d'un auto-encodeur, le goulot d'étranglement, aussi appelé espace latent, contient une représentation compressée de l'entrée du modèle. En effet, la première partie du modèle, l'encodeur, réduit la donnée jusqu'à obtenir un vecteur à quelques dimensions. L'idée est que ce vecteur contient tous les indicateurs importants de la donnée complexe en entrée. Une approche envisageable serait de réduire ce vecteur à 2 ou 3 dimensions, puis de les représenter sur une "carte" ou une visualisation. Le but serait alors de déterminer des zones correspondant à l'OFDM. Il suffirait alors de placer les nouveaux échantillons sur cette carte et les classifier comme étant des anomalies s'ils ne sont pas dans les zones prédéfinies OFDM.

L'avantage de cette solution est que l'aspect détection d'anomalie est conservé. Les résultats qui pourraient être obtenus ne peuvent pas être prédits avec certitude car c'est une solution très peu documentée.

6.2.3. Utilisations de features

L'extraction de caractéristiques à partir des fronts montants a été envisagée plutôt que de travailler avec les échantillons "bruts". L'idée est de choisir quelques caractéristiques signantes pour simplifier le travail de l'IA. Cela nécessite une expertise de la donnée pour savoir quels sont les meilleurs features.

Voici quelques features envisagées :

- Temps de montée 10-50%
- Temps de montée 50-90%
- Temps de montée 10-90%
- Dérivé max
- Quelques bins centraux de la représentation histogramme

Le désavantage de cette méthode est qu'elle repose intégralement sur le choix des features. Il faut aussi pouvoir les extraire, ce qui peut être compliqué suivant la nature de ces dernières. Par contre, étant donné que la plupart de la complexité du problème se trouve dans le choix des features, le travail de l'IA devrait être plus simple et les résultats généralisables lors du passage au réel.

6.3. IA – Classification 4G / 5G

6.3.1. Apprentissage par renforcement

L'apprentissage par renforcement (cf 4.1) permettrait de maintenir le modèle à jour en fonction des modifications de l'environnement dans lequel il opère et cela permettrait également d'améliorer ses performances sur des données réelles. Pour automatiser cette procédure, il faudrait pouvoir valider (ou invalider) la prédiction effectuée. Cela pourrait être fait avec la récupération des blocs d'informations puisque si on arrive à les récupérer, c'est que la technologie prédite est la bonne donc que l'échantillon peut être réintroduit dans le modèle pour prolonger l'apprentissage. Si les blocs d'informations ne sont pas récupérés, les échantillons ne seraient pas intégrés dans l'apprentissage en continu car il n'y aurait pas de moyen de valider les prédictions.

6.3.2. Librairie TSfresh pour faire ressortir des features

La librairie TSfresh est une bibliothèque Python qui permet d'automatiser la recherche de caractéristiques à partir de séries temporelles. Elle fournit des fonctionnalités puissantes pour extraire automatiquement un large éventail de caractéristiques et les classer par ordre d'importance, ce qui peut être utile dans notre cas. Elle peut donc être utilisée en parallèle d'un modèle de prédiction pour réduire le nombre de données à analyser pour effectuer la prédiction et potentiellement améliorer les performances si les caractéristiques trouvées sont pertinentes.

6.3.3. Représentation du signal

Dans le cadre de notre projet, nous avons exploré la représentation fréquentielle du signal à l'aide d'une transformée de Fourier. Cependant, de nombreuses autres transformations existent. Nous avons rencontré certaines d'entre elles lors de notre phase de montée en compétences sur des datasets publics (IEEE Dataport), cf Notebooks dans le dossier *'Classification4G&5G / DatasetIEEE'*. La

SCF (Spectral Correlation Fonction) semblait intéressante puisqu'elle permettait de faire ressortir les caractéristiques cyclo-stationnaires des signaux. Les signaux cellulaires respectant des normes, ils possèdent notamment des patterns périodiques permettant la synchronisation aux cellules qui se retrouvent dans la SCF. Cependant, la complexité de mise en place d'un tel algorithme ainsi que la puissance de calcul requise pour l'effectuer ont fait que cette solution n'a pas été retenue puisque cela aurait grandement impacté la rapidité de la prédiction (plusieurs minutes pour une SCF sur large bande). La balance entre rapidité, exhaustivité et précision aurait donc été déséquilibrée. Une solution pourrait donc être la mise en place d'un calcul matérielle de la SCF sur une cible FPGA. Cela permettrait une diminution du temps de calcul et rendrait possible l'utilisation de cette transformée pour la prédiction.