Table des matières

Liste des figures

Liste des abréviations

PU Primary User

SU Secondary User

SDR Software Defined Radio

SNR Signal Noise Ratio

IA Intelligence Artificielle

LSTM Long Short Term Memory

QoS Quality of Service

TOPSIS Technique for Order of Preference by Similarity to Ideal Solution

CNN Convolutional Neural Networks

RNN Recurrent Neural Networks

DBN Deep Belief Networks

WRAN Wireless Regional Area Network

RN Réseaux de neurones

PMC Perceptron multicouches

RBM Restreint Boltzmann machine

ANFIS Adaptive Neuro-Fuzzy Inference System

Introduction générale

1. Contexte

La radio cognitive (RC) est une radio intelligente qui peut être programmée et configurée de manière dynamique pour utiliser pleinement les ressources de fréquence qui ne sont pas utilisées par les utilisateurs sous licence. Il définit les appareils radio capables d'apprendre et d'adapter leur transmission à l'environnement radio externe, ce qui signifie qu'il dispose d'une certaine intelligence pour surveiller l'environnement radio, apprendre l'environnement et prendre des décisions intelligentes.

Le concept de l'intelligence artificielle (IA) est de faire penser les machines « comme des humains » ; en d'autres termes, effectuer des tâches telles que raisonner, planifier, apprendre et comprendre notre langage. L'utilisation de l'IA dans la RC est très utile, en effet, elle est utilisée dans la mise en œuvre de l'architecture des réseaux RC. Ces derniers doivent pouvoir coexister pour rendre les systèmes de la RC pratiques, ce qui peut générer des interférences aux autres utilisateurs. Afin de traiter ce problème, l'idée de la coopération entre les utilisateurs pour détecter et partager le spectre sans causer d'interférences est mise en place.

2. Problématique

Le réseau RC peut être divisé en deux catégories : utilisateurs primaires (PU) possédant une licence sur le spectre et un utilisateur secondaire (SU) qui va allouer des canaux sur ce spectre.

Ceci veut dire que la présence d'un seul SU qui a besoin de couverture ne pose aucun problème. Mais lorsqu'il y a plusieurs SU au même endroit et au même moment, ceci peut engendrer des problèmes d'encombrement : qui entre les différents SU présents va accéder aux canaux libres du spectre et qui va céder sa place ? Là est la problématique principale de notre travail. Autrement, comment un PU peut choisir de travailler avec un certain SU.

3. Contributions

Notre contribution consiste à améliorer la fonctionnalité de partage du spectre dans les réseaux RC. Généralement, les utilisateurs ont une connaissance limitée de leur environnement, par conséquent, nous prétendons que le comportement coopératif peut leur fournir les informations nécessaires pour résoudre les problèmes.

Nous pensons que l'utilisation des techniques du deep learning dans le partage du spectre est très prometteuse car elle permet de classifier les SU selon des classes bien précises de façon efficace.

Afin de proposer une méthode évolutive, il nous est apparu pertinent d'étudier en détail le deep learning et ses différents types et algorithmes et d'imaginer un scénario sur lequel nous allons mettre en pratique cette technique dans le cadre d'un réseau RC.

Nous avons choisi d'utiliser les réseaux de neurones récurrents (RNN) et plus précisément Long-Short Term Memory (LSTM) pour avoir une bonne prédiction sur les SU dans un scénario où il y a un seul PU et plusieurs SU.

4. Organisation

Ce mémoire contient trois chapitres est organisé comme suit :
Dans le premier chapitre, nous présentons les concepts de la RC. Nous commençons d'abord par la définition des différents concepts comme la radio logicielle, la radio logicielle restreinte et sa relation avec la RC, l'architecture, les différentes phases du cycle de cognition et les fonctions la radio cognitive, enfin nous citerons quelques domaines d'application de la RC.

Le deuxième chapitre est consacré à la présentation de l'IA et les différents types d'apprentissage automatique en détaillant leur principe de façon très brève, nous passerons ensuite à l'apprentissage profond et nous passerons en revue ses différents types.

Le dernier chapitre décrit le scénario que nous avons proposé ainsi que les contributions que nous avons apportés avec leurs résultats accompagnés de graphes de comparaison explicatifs.

Chapitre I

La radio cognitive

I. CHAPITRE I : La radio cognitive

I.1 Introduction

Nous assistons actuellement à la multiplication des normes et des standards de télécommunication vu les progrès récents dans ce domaine. Le nombre croissant de standards normalisés permet d'élargir l'éventail des offres et des services disponibles pour chaque consommateur, d'ailleurs, la plupart des radiofréquences disponibles ont déjà été allouées.
Il est aujourd'hui largement reconnu que les systèmes sans fil de communications numériques n'exploitent pas l'intégralité de la bande de fréquence disponible. Les systèmes sans fil de futures générations seront donc amenés à tirer parti de l'existence des bandes de fréquence inoccupées, grâce à leur faculté d'écouter et de s'adapter à leur environnement.

La radio cognitive est un système qui permet à un terminal de pouvoir interagir avec son environnement. Cela signifie que celui-ci sera capable de percevoir son environnement, de le modéliser et de s'y adapter. Il pourra donc détecter les fréquences libres et les utiliser, contribuant ainsi à une meilleure efficacité spectrale. [1]
Nous allons étudier dans ce chapitre la radio cognitive dans ses différents aspects : principes, architecture, fonctions et les différents domaines d'application...

I.2 Radio logicielle (software radio)

C'est grâce aux travaux de Joseph Mitola que le terme Radio logicielle est apparu en 1991 pour définir une classe de radio reprogrammable et reconfigurable. La radio logicielle est une radio dans laquelle les fonctions typiques de l'interface radio généralement réalisées en matériel, telles que la fréquence porteuse, la largeur de bande du signal, la modulation et l'accès au réseau sont réalisés sous forme logicielle. La radio logicielle moderne intègre également l'implantation logicielle des procédés de cryptographie, codage correcteur d'erreur, codage source de la voix, de la vidéo ou des données.[2]
Le concept de radio logicielle doit également être considéré comme une manière de rendre les usagers, les fournisseurs de services et les fabricants plus indépendants des normes. Ainsi, avec cette solution, les interfaces radio peuvent, en principe, être adaptées aux besoins d'un service particulier pour un usager particulier dans un environnement donné à un instant donné.
La radio logicielle est le but ultime intégrant toute les fonctionnalités en logiciel, mais elle impose des phases intermédiaires combinant anciennes et nouvelles techniques, on parle alors de radio logicielle restreinte (software defined radio). Les contraintes de puissance de calcul, de consommation électrique, de coûts, etc. imposent actuellement de passer par cette phase intermédiaire [3].

I.2.1 Radio logicielle restreinte (Software Defined Radio: SDR)

La radio logicielle restreinte est un système de communication radio qui peut s'adapter à

n'importe quelle bande de fréquence et recevoir n'importe quelle modulation en utilisant le même matériel. Les opportunités qu'offre le SDR lui permettent de résoudre des problèmes de la gestion dynamique du spectre. Les équipements SDR peuvent fonctionner dans des réseaux sans fil hétérogènes c'est-à-dire qu'un SDR idéal peut s'adapter automatiquement aux nouvelles fréquences et aux nouvelles modulations [4].

I.3 Radio cognitive

I.3.1 Historique

L'idée de la radio cognitive a été présentée officiellement par Joseph Mitola III à un séminaire à KTH, l'Institut royal de technologie, en 1998, publié plus tard dans un article de Mitola et Gerald Q. Maguire, Jr en 1999.[5]
Mitola combine son expérience de la radio logicielle ainsi que sa passion pour l'apprentissage automatique et l'intelligence artificielle pour mettre en place la technologie de la radio cognitive. D'après lui : **« Une radio cognitive peut connaître, percevoir et apprendre de son environnement puis agir pour simplifier la vie de l'utilisateur. »** [5]

I.3.2 Définitions

La « cognition » est un processus par lequel on acquiert des connaissances, elle regroupe les divers processus mentaux allant de l'analyse perceptive de l'environnement à la commande motrice (en passant par la mémorisation, le raisonnement, les émotions, le langage…). [6]
Le terme radio cognitive est utilisé pour décrire un système ayant la capacité de détecter et de reconnaître son cadre d'utilisation, ceci afin de lui permettre d'ajuster ses paramètres (fréquence, puissance, modulation, bande passante) de fonctionnement radio de façon dynamique et autonome et d'apprendre des résultats de ses actions et de son cadre environnemental d'exploitation, comme le souligne Dr. Mitola.
Cette capacité permet d'adapter chaque appareil aux conditions spectrales du moment et offre donc aux utilisateurs un accès plus souple, efficace et complet à cette ressource. Cette approche peut améliorer considérablement le débit des données et la portée des liaisons sans augmenter la bande passante ni la puissance de transmissions. La radio cognitive offre également une solution équilibrée au problème de l'encombrement du spectre en accordant d'abord l'usage prioritaire au propriétaire du spectre, puis en permettant à d'autres de se servir des portions inutilisées du spectre. [6]

I.3.3 Principe

Puisque la majeure partie du spectre est déjà assignée, le défi le plus important est de partager le spectre autorisé. La radio cognitive permet l'utilisation de spectre temporellement inutilisé, qui est mentionné comme trou de spectre ou espace blanc. Si cette bande est encore utilisée par un utilisateur autorisé, la RC offre des mouvements à un autre espace blanc, en changeant son niveau de puissance de transmission ou arrangement de modulation pour éviter l'interférence. De part cette vision, deux éléments sont omniprésents dans chaque

communication radio cognitive. [6]

a Utilisateurs primaires

Ces utilisateurs (dit utilisateurs licenciés) sont des utilisateurs qui disposent d'une licence qui leur permet d'opérer sur des bandes spectrales qui leurs sont réservées, ainsi ils ont le droit de communiquer en toute liberté à tout instant sur leurs bande de fréquence. [6]

b Utilisateurs secondaires

Ces utilisateurs ne possédant pas de licence, ils accèdent au spectre de façon opportuniste, mais ils doivent veiller à ne pas gêner les utilisateurs primaires, en effet ils doivent prendre la responsabilité de ne jamais interférer avec les utilisateurs primaires.[6]

En effet, à un temps donné et à un emplacement géographique spécifique, il arrive qu'un utilisateur primaire n'utilise plus sa bande de fréquence, ainsi d'autres utilisateurs secondaires peuvent exploiter ces fréquences grâce à des trous dans le spectre comme le montre la **Figure I.1** ci-dessous, et ça, sans perturber les communications des utilisateurs primaires, néanmoins, les systèmes sans fil existants ont été conçus pour fonctionner sur des fréquences dédiées, ce qui fait qu'ils ne pourront pas profiter de cette flexibilité prévue.[6]

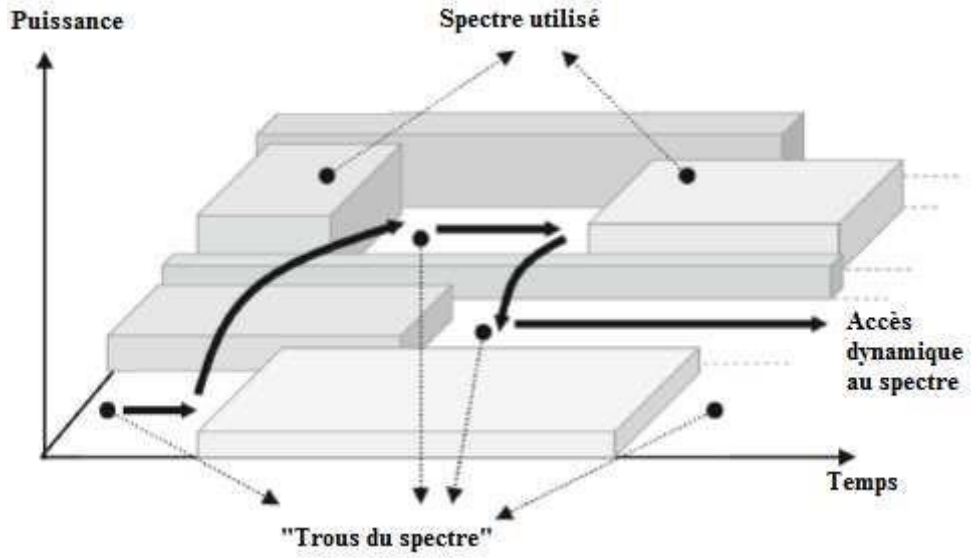

Figure I.1: Le concept des trous du spectre

I.3.4 Architecture

Mitola a défini l'architecture d'une RC par un ensemble cohérent de règles de conception par lequel un ensemble spécifique de composants réalise une série de fonctions de produits et de services. [2]

L'architecture la plus simple d'une radio intelligente est l'ensemble minimaliste des composantes fonctionnelles montré dans de la **Figure I.2**. [7]

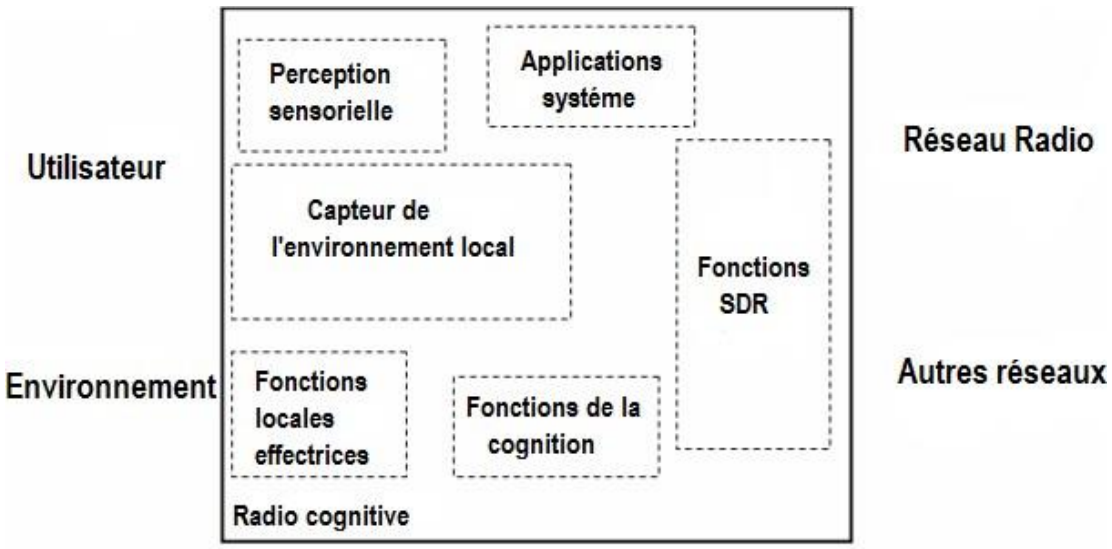

Figure I.2: Architecture de la radio cognitive.

- L'interface de perception sensorielle de l'utilisateur comprend les fonctions de capture (sensing) et de perception, visuelles et acoustiques.
- Les capteurs de l'environnement local (position, température, accélération, etc.).
- Les applications système (les services médias indépendants comme un jeu en réseau).
- Les fonctions de la radio logicielle restreinte (SDR) (qui incluent la détection RF et les applications radio de la SDR).
- Les fonctions de la cognition (contrôle, planification, apprentissage).
- Les fonctions locales effectrices (synthèse de parole, texte, graphiques, affichages multimédias). [2]

I.3.5 Cycle de cognition

La **Figure I.3** illustre le cycle de cognition ainsi appelé par Mitola et Maguire [1] qui synthétise les fonctions de cognition de l'architecture d'une radio intelligente. Une telle radio cognitive observe l'environnement, s'oriente, crée des plans, décide, et puis agit.

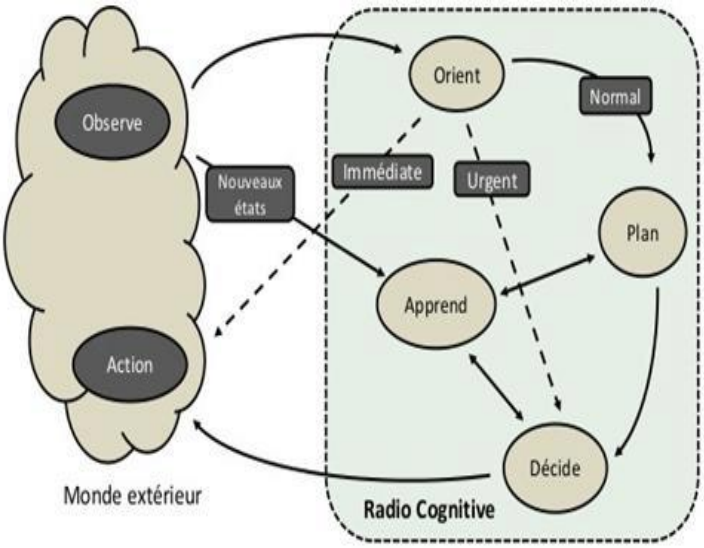

Figure I.4: Le cycle de cognition de Mitola et Maguire.

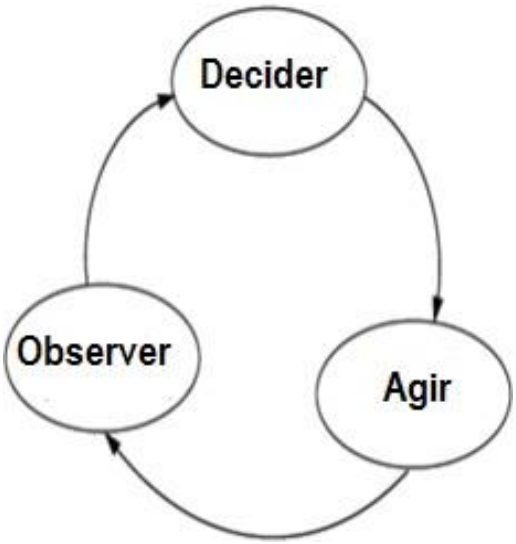

Figure I.3: Le cycle de cognition simplifié.

L'observation (**observer**) consiste en la compréhension de l'environnement radio, des besoins des utilisateurs, des contraintes existantes. L'orientation (**orienter**) fait référence à la façon de déterminer s'il y a besoin d'une action urgente ou si une planification à long terme est plutôt nécessaire. La planification (**planifier**) consiste en la prise de décisions à long terme. Le processus de décision (**décision**) se sert des observations pour produire une action ou un ensemble d'actions (**agir**). L'apprentissage quant à lui dépend des phases d'observation, de décision, d'action et de planification. L'apprentissage initial est réalisé à travers les phases d'observation et d'action dans lesquelles toutes les perceptions sensorielles ainsi que les actions sont continuellement comparées à l'ensemble des expériences antérieures. Il est possible de simplifier le schéma de la **Figure I.3** [8], sans perte de généralité si l'on intègre à "**décider**" tout ce qui sous-entend une notion d'intelligence, comme "**orienter**", "**planifier**", "**apprendre**". Ceci revient au cycle montré dans la **Figure I.4.** [8]

I.3.6 Composantes de la radio cognitive

Les différentes composantes d'un émetteur/récepteur radio cognitive qui mettent en œuvre ces fonctionnalités sont présentées dans la **Figure I.5** [4].

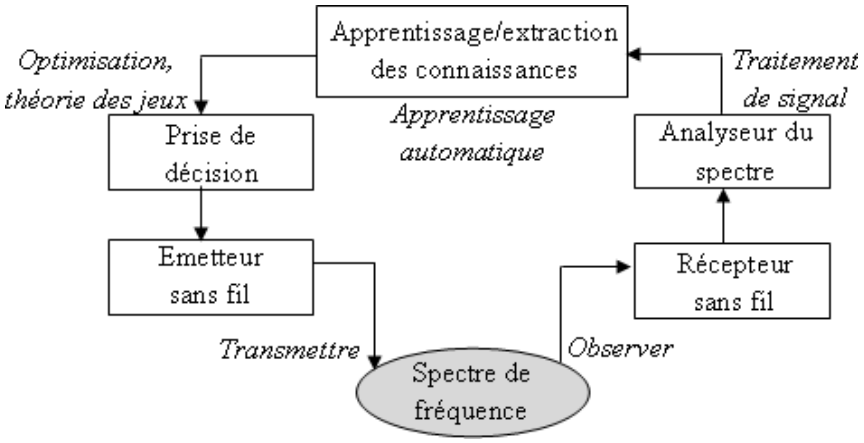

Figure I.5: Composantes de la radio cognitive.

Emetteur / Récepteur: un émetteur/récepteur SDR sans fil est le composant majeur avec les fonctions du signal de transmission de données et de réception. En outre, un récepteur sans fil est également utilisé pour observer l'activité sur le spectre de fréquence (spectre de détection). Les paramètres émetteur/récepteur dans le nœud de la RC peuvent être modifiés dynamiquement comme dicté par les protocoles de couche supérieure.

Analyseur de spectre*:* l'analyseur de spectre utilise les signaux mesurés pour analyser l'utilisation du spectre (par exemple pour détecter la signature d'un signal provenant d'un utilisateur primaire et trouver les espaces blancs du spectre pour les utilisateurs secondaires). L'analyseur de spectre doit s'assurer que la transmission d'un utilisateur primaire n'est pas perturbée si un utilisateur secondaire décide d'accéder au spectre. Dans ce cas, diverses techniques de traitement du signal peuvent être utilisées pour obtenir des informations sur l'utilisation du spectre.

Extraction de connaissances et apprentissage: l'apprentissage et l'extraction de connaissances utilisent les informations sur l'utilisation du spectre pour comprendre l'environnement ambiant RF (par exemple le comportement des utilisateurs sous licence). Une base de connaissances de l'environnement d'accès au spectre est construite et entretenue, qui est ensuite utilisée pour optimiser et adapter les paramètres de transmission pour atteindre l'objectif désiré sous diverses contraintes. Les algorithmes d'apprentissage peuvent être appliqués pour l'apprentissage et l'extraction de connaissances.

Prise de décision: après que la connaissance de l'utilisation du spectre soit disponible, la décision sur l'accès au spectre doit être faite. La décision optimale dépend du milieu ambiant, elle dépend du comportement coopératif ou compétitif des utilisateurs secondaires. Différentes techniques peuvent être utilisées pour obtenir une solution optimale.

Par exemple, la théorie d'optimisation peut être appliquée lorsque le système peut être modélisé comme une seule entité avec un seul objectif. En revanche, les modèles de la théorie des jeux peuvent être utilisés lorsque le système est composé d'entités multiples, chacun avec son propre objectif. L'optimisation stochastique peut être appliquée lorsque les états du système sont aléatoires.

I.3.7 Fonctions de la radio cognitive

Les principales fonctions de la radio cognitive sont les suivantes:

a Détection du spectre (Spectrum sensing)

Détecter le spectre non utilisé et le partager sans interférence avec d'autres utilisateurs. La détection des utilisateurs primaires est la façon la plus efficace pour détecter les espaces blancs du spectre.

L'un des objectifs de la détection du spectre, en particulier pour la détection des interférences, est d'obtenir le statut du spectre (libre /occupé), de sorte que le spectre peut être consulté par un utilisateur secondaire en vertu de la contrainte d'interférence. Le défi réside dans le fait de mesurer l'interférence au niveau du récepteur primaire causée par les transmissions d'utilisateurs secondaires.

b Gestion du spectre (Spectrum management)

Capter les meilleures fréquences disponibles pour répondre aux besoins de communication des utilisateurs.

Les radios cognitives devraient décider de la meilleure bande de spectre pour répondre aux exigences de qualité de service sur toutes les bandes de fréquences disponibles, donc les fonctions de gestion du spectre sont nécessaires pour les radios cognitives. Ces fonctions de gestion peuvent être classées comme suit:

- **Analyse du spectre**

Les résultats obtenus de la détection du spectre sont analysés pour estimer la qualité du spectre. Une des questions ici est de savoir comment mesurer la qualité du spectre qui peut être accédée par un utilisateur secondaire. Cette qualité peut être caractérisée par le rapport signal/bruit, la durée moyenne et la corrélation de la disponibilité des espaces blancs du spectre. Les informations sur cette qualité de spectre disponible à un utilisateur par radio cognitive peuvent être imprécises et bruyantes. Des algorithmes d'apprentissage de l'intelligence artificielle sont des techniques qui peuvent être employées par les utilisateurs de la radio cognitive pour l'analyse du spectre.

- **Décision sur le spectre**

- ***Modèle de décision*****:** un modèle de décision est nécessaire pour l'accès au spectre. La complexité de ce modèle dépend des paramètres considérés lors de l'analyse du spectre.
  Le modèle de décision devient plus complexe quand un utilisateur secondaire a des objectifs multiples. Par exemple, un utilisateur secondaire peut avoir l'intention de maximiser son rendement tout en minimisant les perturbations causées à l'usager primaire.
- ***Compétition / coopération dans un environnement multi utilisateurs :*** lorsque plusieurs utilisateurs (à la fois primaires et secondaires) sont dans le système, leur préférence va influer sur la décision du spectre d'accès. Ces utilisateurs peuvent être coopératifs ou non coopératifs dans l'accès au spectre.

Dans un environnement non-coopératif, chaque utilisateur a son propre objectif, tandis que dans un environnement coopératif, tous les utilisateurs peuvent collaborer pour atteindre un seul objectif. Par exemple, plusieurs utilisateurs secondaires peuvent entrer en compétition les uns avec les autres pour accéder au spectre radio (par exemple, O1, O2, O3, O4 dans la **Figure I.6**) de sorte que leur débit individuel soit maximisé. Au cours de cette concurrence entre les utilisateurs secondaires, tous veillent à ce que l'interférence causée à l'utilisateur primaire est maintenue en dessous de la limite de température de brouillage correspondante. La théorie des jeux est l'outil le plus approprié pour obtenir la solution d'équilibre pour le problème du spectre dans un tel scénario.

Dans un environnement coopératif, les radios cognitives coopèrent les unes avec les autres pour prendre une décision pour accéder au spectre et de maximiser une fonction objective commune en tenant compte des contraintes. Dans un tel scénario, un contrôleur central peut coordonner le spectre de gestion.

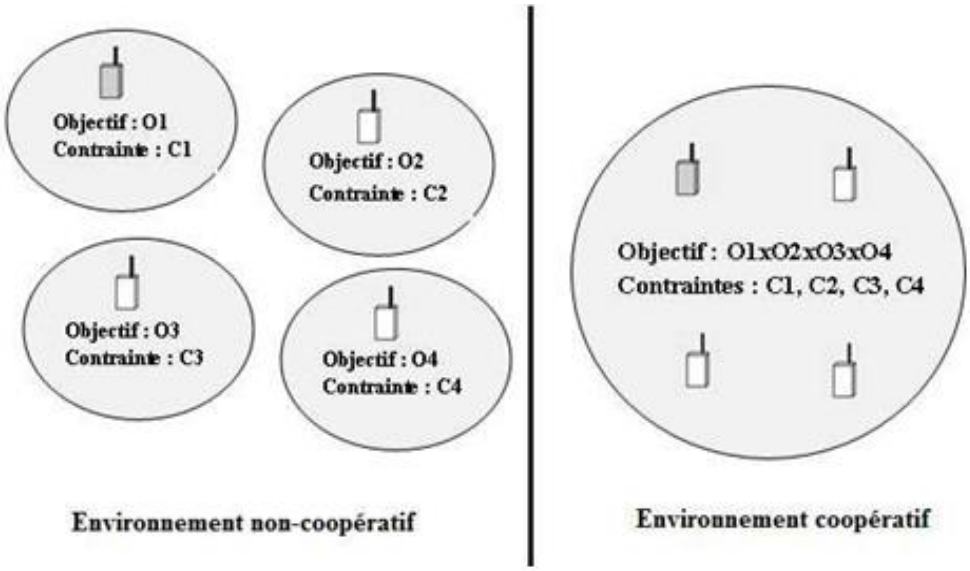

Figure I.6: Accès au spectre coopératif et non-coopératif [4]

- ***Mise en œuvre distribuée du contrôle d'accès au spectre*** : dans un environnement multi utilisateur distribué, pour un accès non-coopératif au spectre, chaque utilisateur peut parvenir à une décision optimale de façon indépendante en observant le comportement (historique / action) des autres utilisateurs du système. Par conséquent, un algorithme distribué est nécessaire pour un utilisateur secondaire pour prendre la décision sur l'accèsau spectre de manière autonome.

c Mobilité du spectre (Spectrum mobility)

C'est le processus qui permet à l'utilisateur de la radio cognitive de changer sa fréquence de fonctionnement.

Les réseaux radio cognitifs essayent d'utiliser le spectre de manière dynamique en permettant à des terminaux radio de fonctionner dans la meilleure bande de fréquence disponible, de

maintenir les exigences de communication transparentes au cours de la transition à une meilleure fréquence.

- **Recherche des meilleures bandes de fréquence**

La radio cognitive doit garder une trace des bandes de fréquence disponibles de sorte que si nécessaire (par exemple, un utilisateur autorisé est détecté), il peut passer immédiatement à d'autres bandes de fréquences. Lors de la transmission par un utilisateur secondaire, l'état de la bande de fréquences doit être respecté.

- **Auto-coexistence et synchronisation**

Quand un utilisateur secondaire effectue un transfert du spectre, deux questions doivent être prises en compte. Le canal cible ne doit pas être actuellement utilisé par un autre utilisateur secondaire (l'exigence d'auto-coexistence), et le récepteur de la liaison secondaire correspondant doit être informé de la non-intervention du spectre (la demande de synchronisation) [4].

I.3.8 Domaines d'application de la radio cognitive

Parmi les domaines d'application de la radio cognitive, on peut citer :

Les réseaux sans fil de prochaine génération :
La radio opportuniste (RC) est apparue comme une technologie clé pour la prochaine génération des réseaux sans fil hétérogènes.

Coexistence de différentes technologies sans fil :
IEEE 802.22, basée sur les utilisateurs WRAN peut utiliser efficacement la bande TV quand il n'y a pas d'utilisation du téléviseur à proximité ou quand une station de télévision ne diffuse pas.

Services de cyber santé (eHealth services) :
Depuis que les équipements médicaux et les capteurs bio-signal sont sensibles aux interférences électromagnétiques, la puissance d'émission des appareils sans fil doit être soigneusement contrôlée.
En outre, différents dispositifs biomédicaux (équipement et appareils chirurgicaux, de diagnostic et de suivi) utilisent la transmission RF. L'utilisation du spectre de ces dispositifs doit être choisie avec soin pour éviter toute interférence avec l'autre. Dans ce cas, les concepts de la radio cognitive peuvent être appliqués.

Réseaux d'urgence :
Les réseaux de sécurité publique et d'urgence peuvent profiter des concepts de la radio cognitive pour fournir la fiabilité et la flexibilité de communication sans fil.

Réseaux militaire :
Avec la RC, les paramètres de la communication sans fil peuvent être adaptés de manière dynamique en fonction du temps et de l'emplacement ainsi que de la mission des soldats.

I.4 Conclusion

Dans ce chapitre, nous avons présenté les caractéristiques principales de la radio cognitive, nous avons introduit plusieurs concepts comme la radio logicielle, la SDR, l'architecture de RC, le cycle de cognition, les composantes de la RC, les fonctions de la RC et les domaines d'application.
Afin de pouvoir tirer le maximum de profit de la bande passante globale disponible, une gestion optimisée du spectre s'impose. La RC est, avant tout, un système radio qui offre aux utilisateurs un débit supérieur et une meilleure qualité de service, et finalement, une augmentation du confort dans les communications.

Dans le chapitre suivant, nous allons présenter le deep learning avec ses différents algorithmes et domaines.

CHAPITRE II

Deep Learning

II. Chapitre II: Deep Learning

II.1 Introduction IA

L'objectif de la recherche en intelligence artificielle (IA) est de doter un système informatique de capacités de réflexion similaires à celles des humains. Il y a donc un défi dans la compréhension du raisonnement humain, mais surtout dans la modélisation et dans la reproduction de celui-ci.[9]

L'IA est devenue un sujet en vogue dans les médias et magazines scientifiques en raison des nombreuses réalisations, dont beaucoup sont le fruit des progrès accomplis dans le domaine de l'apprentissage automatique. De grandes entreprises dont Google, Facebook, IBM, Microsoft mais aussi des constructeurs automobiles à l'instar de Toyota, Volvo et Renault, sont très actifs dans la recherche en IA et prévoient d'y investir davantage encore dans le futur. Plusieurs scientifiques spécialisés dans l'IA dirigent désormais les laboratoires de recherche de ces grandes entreprises et de nombreuses autres. La recherche en IA a permis de réaliser d'importants progrès dans la dernière décennie, et ce dans différents secteurs. Les avancées les plus connues sont celles réalisées dans l'apprentissage automatique, grâce notamment au développement d'architectures d'apprentissage profond, des réseaux de neurones convolutifs multicouche dont l'apprentissage s'opère à partir de gros volumes de données sur des architectures de calcul intensif. Parmi les réalisations de l'apprentissage automatique, il convient de citer la résolution de jeux Atari (Bricks, Space invaders, etc.) par Google DeepMind, utilisant les pixels images affichés à l'écran comme données d'entrée afin de décider quelle action adopter pour atteindre le plus haut score possible à la fin de la partie.[10]

Dans ce chapitre, nous allons parler des différents types d'apprentissage automatique et détailler leur principe de façon très brève, nous passerons ensuite à l'apprentissage profond et nous passerons en revue ses différents types.

II.2 Apprentissage automatique

L'apprentissage automatique fait référence au développement, à l'analyse et à l'implémentation de méthodes qui permettent à une machine d'évoluer grâce à un processus d'apprentissage, et ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classiques. [11]

Apprentissage Supervisé : Cette approche a pour objectif la conception d'un modèle reliant des données d'apprentissage à un ensemble de valeurs de sortie (un comportement) [5]. Un expert est employé pour étiqueter correctement des exemples. L'apprenant doit alors trouver ou approximer la fonction qui permet d'affecter la bonne étiquette à ces exemples. [11]

La **Figure II. 1** représente le schéma de l'apprentissage supervisé, le processus se déroule en deux phrases. La première est appelée la phase d'apprentissage durant laquelle le professeur fournit une étiquette à chaque exemple disponible dans l'environnement en les associant à une classe : x1,x2,...,xm,... et xn, xO,... dans le cas présenté ici. Ensuite il classifie les exemples en élaborant un échantillon de données étiquetées noté: Sm =(x1, U1), (x2,U2),…, (xm,Um). Il guide ainsi l'apprenant en lui fournissant des exemples sur ce qu'il doit apprendre à savoir faire.

La seconde étape est nommée phase de test. Elle consiste à tenter de prédire l'étiquette d'un nouvel exemple en utilisant une fonction apprise à partir de Sm notée h. Avec cette hypothèse, l'apprenant peut associer Yn=h(xn) à chaque xn, Yo=h(xo) pour chaque xo, et etc... Plus ces valeurs seront proches de celles que le professeur aurait fournies à sa place, meilleur sera l'apprentissage. L'apprenant apprend donc par lui-même à classer ou à décider d'une action comme le fait le professeur. [12]

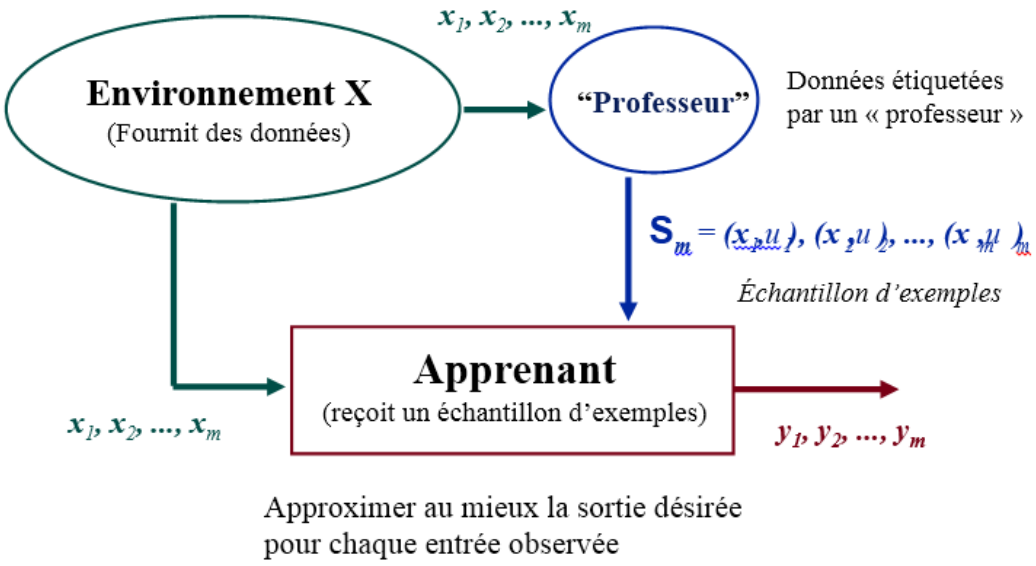

Figure II.1: schéma de l'apprentissage supervisé

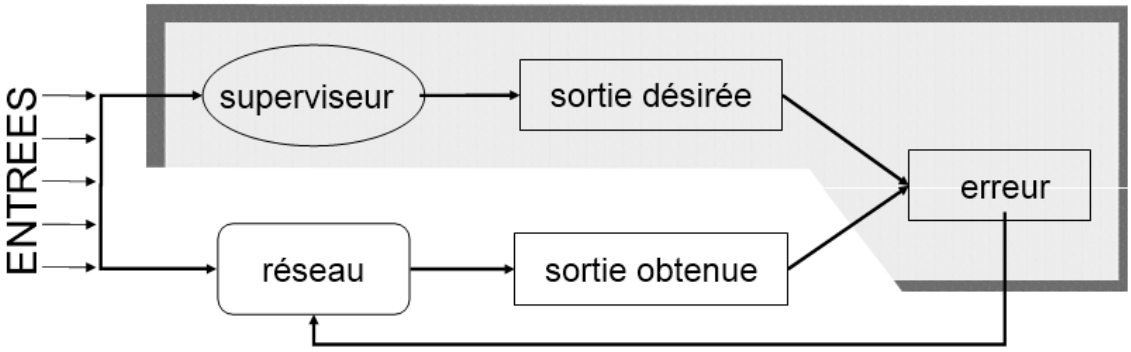

Figure II.2: Schéma d'un modèle supervisé. [3]

Apprentissage Non-Supervisé : Aucun expert n'est disponible. Il vise à concevoir un modèle structurant l'information. La différence ici est que les comportements (ou catégories ou encore les classes) des données d'apprentissage ne sont pas connus, c'est ce que l'on cherche à trouver. [13]

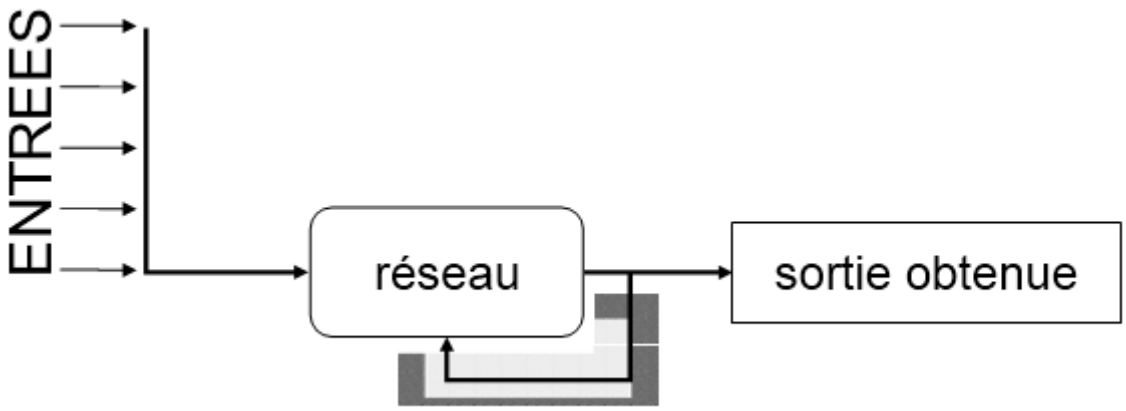

Figure II.3: Schéma d'un modèle non supervisé. [3]

Apprentissage par Renforcement : Les données en entrée sont les mêmes que pour l'apprentissage supervisé, cependant l'apprentissage est guidé par l'environnement sous la forme de récompenses ou de pénalités données en fonction de l'erreur commise lors de l'apprentissage. [13]

II.3 Réseaux de neurones (RN)

Les réseaux de neurones fonctionnent en répartissant les valeurs des variables dans des automates (les *neurones*). Ces unités sont chargées de combiner entre elles leurs informations pour déterminer la valeur du paramètre de discrimination. C'est de la connexion de ces unités entre elles qu'émerge la capacité de discrimination du RN.

Chaque neurone reçoit des informations numériques en provenance de neurones voisins ; à chacune de ces valeurs est associé un *poids* représentatif de la force de la connexion. Chaque neurone effectue localement un calcul dont le résultat est transmis ensuite aux neurones avals. [14]

Dans la littérature, on parle de perceptron simple et de perceptron multicouche.

Perceptron simple

Il s'agit d'un neurone binaire, c'est-à-dire dont la sortie vaut 0 ou 1. Pour calculer cette sortie, le neurone effectue une somme pondérée de ses entrées (chaque entrée possède un poids) :

$Y = f(W1 \times \mathrm{X1} + \mathrm{W2} \times \mathrm{X2})$, puis applique une fonction d'activation à seuil : si la somme pondérée dépasse une certaine valeur, la sortie du neurone est 1, sinon elle vaut 0. Il ne peut donc faire que de la classification, puis de la prédiction. [15]

La figure II.4 montre le schéma d'un perceptron simple.

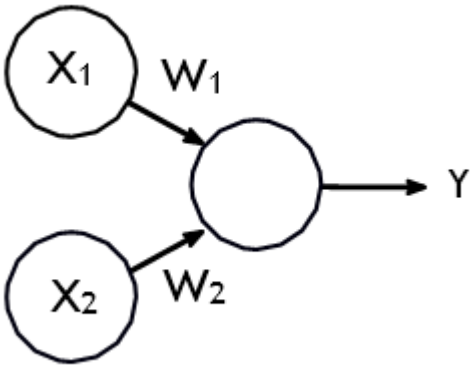

Figure II.4 : schéma du perceptron simple.

Perceptron multicouches (PMC)

La famille de réseau de neurones majoritairement employée est le perceptron multicouches.
À lui seul ce type de réseau recouvre plus de 95 % des applications scientifiques et industrielles. Il comporte quelques dizaines à quelques centaines de neurones dans les cas usuels, voir plusieurs milliers pour les applications graphiques.
Le PMC est un modèle de *réseau à propagation par couche* (**Figure II. 1**). Les neurones y sont organisés en couches successives : une couche d'entrée, une couche de sortie et entre les deux une ou plusieurs couches intermédiaires, appelées aussi *couches cachées*. Il n'existe pas de connexion entre les neurones d'une même couche, en revanche tout neurone d'une couche est connecté à tous les neurones de la couche suivante.
La « couche » d'entrée n'est pas une réelle couche de neurones car elle se contente de coder les variables d'observation. La couche de sortie code la variable de discrimination. Les valeurs d'activité des neurones sont propagées dans le réseau, de l'entrée vers la sortie, sans retour arrière. La présence d'une couche cachée permet de modéliser des relations non linéaires entre les entrées et la sortie. En théorie une seule couche cachée suffit, mais le fait de disposer d'une seconde couche cachée permet de modéliser plus facilement une fonction de discrimination non continue. En pratique, la plupart des problèmes sont résolus avec un ou deux niveaux, trois au maximum. [14]

La figure II.5 illustre l'estimation de l'âge au décès à partir de l'observation de critères osseux de la surface sacro-pelvienne iliaque. Les entrées correspondent à l'observation de critères morphologiques sur la surface sacro.

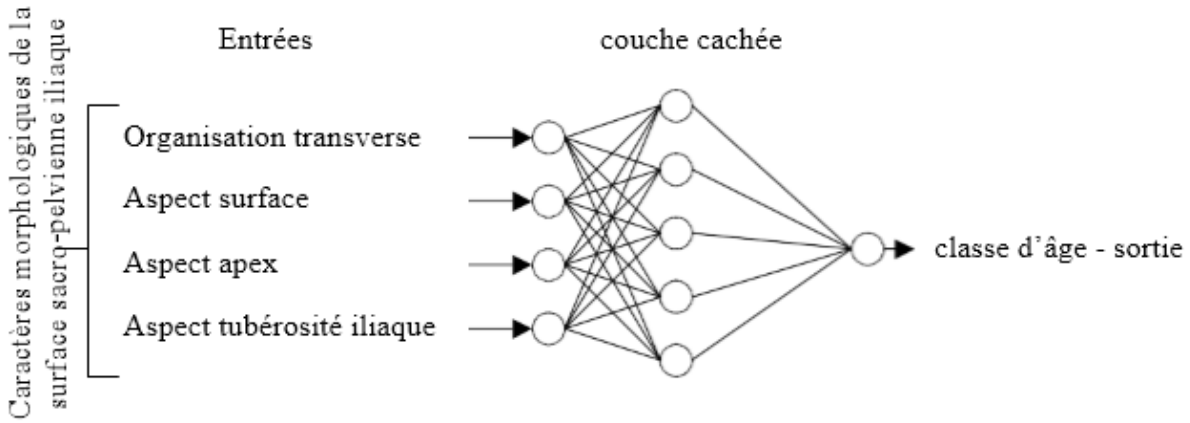

Figure II.5 : Schéma d'un perceptron multicouches

II.4 Apprentissage profond

Au sein de la révolution que connait l'Intelligence Artificielle aujourd'hui, l'apprentissage profond (deep learning) occupe une place de tout premier plan. Il est au cœur des technologies qui permettent aujourd'hui de réaliser des tâches encore impensables il y a quelques années, comme la conduite automatique ou la conversation avec des assistants vocaux. La réussite du deep learning repose à la fois sur des avancées en terme de modélisation des représentations de l'information, sur des méthodes d'apprentissage statistiques exploitant des données massivement annotées, et sur des calculateurs très puissants adaptés à ces architectures. [16]
Les modèles de Deep Learning sont bâtis sur le même modèle que les perceptrons multicouches précédemment décrits. Cependant, il convient de souligner que les différentes couches intermédiaires sont plus nombreuses.
Chacune des couches intermédiaires va être subdivisée en sous partie, traitant un sous problème, plus simple et fournissant le résultat à la couche suivante, et ainsi de suite. Ainsi, dans le cadre du traitement du langage, imaginons un extrait audio parlant de l'équipe d'Algérie de football, les différentes couches pourraient se hiérarchiser de cette façon :

- Couche 1 : Sport.
- Couche 2 : Sport collectif.
- Couche 3 : Football.
- Couche 4 : Equipe d'Algérie.

Cette manière d'ordonner les déductions amènent les modèles de Deep Learning à se rapprocher un peu plus du mode de fonctionnement du cerveau humain, en ajoutant au fur et à mesure un contexte de plus en plus précis au sujet sur lequel le modèle opère. [17]

Il existe différents algorithmes de Deep Learning. Nous pouvons ainsi citer :

II.4.1 Les réseaux de neurones récurrents (RNN ou Recurrent Neural Networks).

Les humains ne commencent pas leur réflexion à partir de zéro chaque seconde. En lisant cet essai, vous comprenez chaque mot en fonction de votre compréhension des mots précédents. Vous ne jetez pas tout et recommencez à penser à partir de zéro. Vos pensées ont de la

persévérance.
Les réseaux de neurones traditionnels ne peuvent pas le faire, et cela semble être une lacune majeure. Par exemple, imaginez que vous souhaitiez classer le type d'événement qui se produit à chaque étape d'un film. On ne sait pas comment un réseau de neurones traditionnel pourrait utiliser son raisonnement sur des événements antérieurs dans le film pour les informer plus tard.

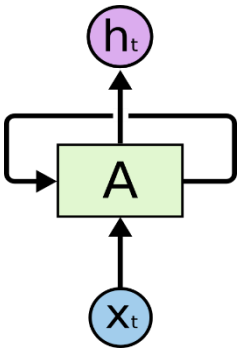

Figure II.6 : Les réseaux neuronaux récurrents ont des boucles.

Les réseaux de neurones récurrents résolvent ce problème. Ce sont des réseaux avec des boucles qui permettent à l'information de persister.
Dans la figure II.6 ci-dessus, un segment de réseau neuronal; "**A**" regarde une entrée $\mathbf{X_t}$ et fournit une valeur h_t. Une boucle permet de passer des informations d'une étape du réseau à l'autre.
Ces boucles rendent les réseaux neuronaux récurrents un peu mystérieux. Cependant, si vous pensez un peu plus, il s'avère qu'ils ne sont pas tous différents d'un réseau de neurones normal. Un réseau de neurones récurrent peut être considéré comme des copies multiples du même réseau, chacune transmettant un message à un successeur comme le montre la figure II.7. Considérez ce qui se passe si nous déroulons la boucle:

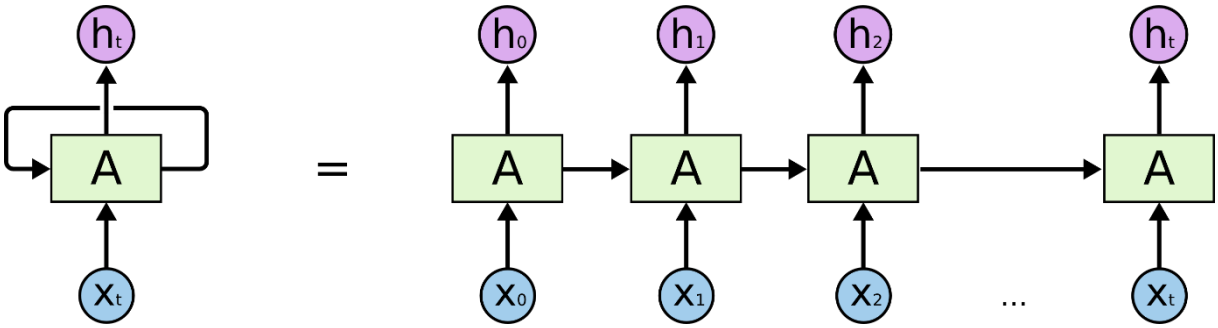

Figure II.7: Un réseau neuronal récurrent déroulé.

Cette nature en chaîne révèle que les réseaux neuronaux récurrents sont intimement liés aux

séquences et aux listes. Ils sont l'architecture naturelle du réseau de neurones à utiliser pour de telles données.

Au cours des dernières années, il y a eu un succès incroyable en appliquant les RNN à une variété de problèmes: la reconnaissance de la parole, la modélisation du langage, la traduction, le sous-titrage des images ...etc. [18]

II.4.2 Les réseaux de neurones convolutionnels (CNN ou Convolutional Neural Networks)

Les réseaux de neurones convolutifs sont à ce jour les modèles les plus performants pour classer des images [19]. Désignés par l'acronyme CNN, de l'anglais Convolutional Neural Network, ils comportent deux parties bien distinctes. En entrée, une image est fournie sous la forme d'une matrice de pixels. Elle a 2 dimensions pour une image en niveaux de gris. La couleur est représentée par une troisième dimension, de profondeur 3 pour représenter les couleurs fondamentales [Rouge, Vert, Bleu].

La première partie d'un CNN est la partie convolutive à proprement parler. Elle fonctionne comme un extracteur de caractéristiques des images. Une image est passée à travers une succession de filtres, ou noyaux de convolution, créant de nouvelles images appelées cartes de convolutions. Certains filtres intermédiaires réduisent la résolution de l'image par une opération de maximum local. Au final, les cartes de convolutions sont mises à plat et concaténées en un vecteur de caractéristiques, appelé code CNN. La figure II.8 illustre l'architecture standard d'un CNN.

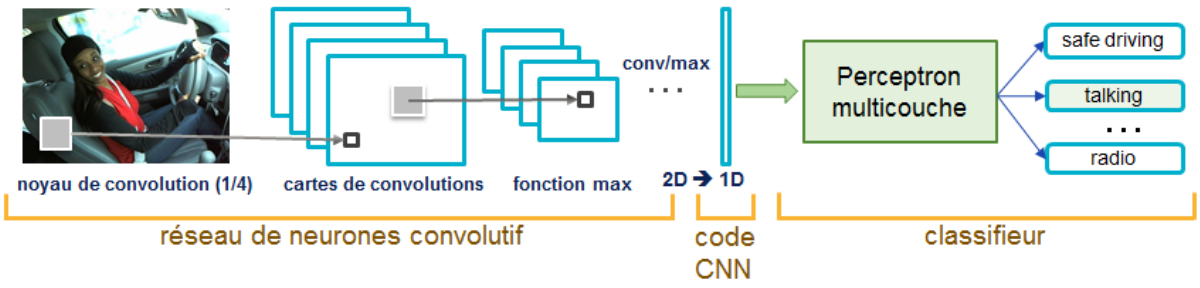

Figure II.8: Architecture standard d'un réseau de neurone convolutionnel.

Ce code CNN en sortie de la partie convolutive est ensuite branché en entrée d'une deuxième partie, constituée de couches entièrement connectées (perceptron multicouche). Le rôle de cette partie est de combiner les caractéristiques du code CNN pour classer l'image.
La sortie est une dernière couche comportant un neurone par catégorie. Les valeurs numériques obtenues sont généralement normalisée entre 0 et 1, de somme 1, pour produire une distribution de probabilité sur les catégories. [19]

II.4.3 La machine de Boltzmann profonde (DBN ou Deep Belief Network)

DBN est un réseau de neurones constitué de plus d'une machine Boltzmann restreinte (RBM). RBM et DBN ont été conçus par Geoffrey Hinton. [20]

RBM est un algorithme utile pour la réduction des dimensions, la régression, le filtrage collaboratif, classification, apprentissage de fonctionnalités et modélisation de thèmes.

Il C'est un réseau de neurones à deux couches peu profond composé d'une couche d'entrée visible et une couche de sortie cachée. Ils sont les blocs de construction de DBN.

Pour chaque RBM la couche visible représente les inputs et la couche cachée reçoit les inputs de la couche visible, fait les calculs ensuite elle envoie les résultats à la couche visible.

Dans la DBN chaque couche cachée du RBM représente la couche visible de la RBM suivante comme le montre la figure II.9. [21]

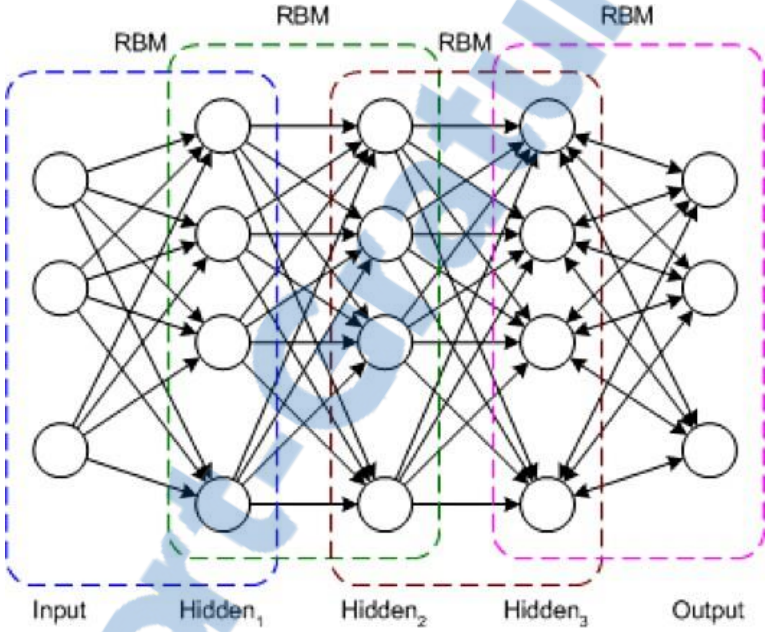

Figure II.9: DBN.

Les DBN ont été utilisés pour générer et reconnaître des images (reconnaissance faciale), des séquences vidéo et des données de capture de mouvement. La NASA utilise DBN pour classer des téraoctets d'imagerie satellitaire à haute résolution et très diversifiée.

II.4.4 Domaines d'applications de Deep Learning

Des applications du Deep Learning sont utilisées dans divers secteurs, de la conduite automatisée aux dispositifs médicaux.

Conduite automatisée : Les chercheurs du secteur automobile ont recours au Deep Learning pour détecter automatiquement des objets tels que les panneaux stop et les feux de circulation. Le Deep Learning est également utilisé pour détecter les piétons, évitant ainsi nombre d'accidents.

Aérospatiale et défense : Le Deep Learning sert à identifier des objets à partir de satellites utilisés pour localiser des zones d'intérêt et identifier quels secteurs sont sûrs ou dangereux pour les troupes au sol.

Recherche médicale : À l'aide du Deep Learning, les chercheurs en cancérologie peuvent dépister automatiquement les cellules cancéreuses. Des équipes de l'Université de Californie à Los Angeles (UCLA) ont conçu un microscope qui génère un ensemble de données de grande dimension afin d'entraîner une application de Deep Learning à identifier avec précision des cellules cancéreuses.

Automatisation industrielle : Le Deep Learning participe à l'amélioration de la sécurité des employés travaillant à proximité d'équipements lourds, en détectant automatiquement les situations dans lesquelles la distance de sécurité qui sépare le personnel ou les objets des machines est insuffisante.

Électronique : Le Deep Learning est utilisé pour la reconnaissance audio et vocale. Par exemple, les appareils d'assistance à domicile qui répondent à votre voix et connaissent vos préférences fonctionnent grâce à des applications de Deep Learning.

II.5 Deep learning dans la Radio cognitive :

Dans [22], la conception et la mise en œuvre d'un algorithme basé sur réseau de neurones récurrents *à* mémoire court-terme et long-terme (LSTM) est proposé afin d'augmenter le pourcentage de succès dans la prévision (présence / absence) de PU dans les canaux de spectre. Le niveau de précision affiché dans les résultats indiquent que LSTM augmente le pourcentage de prédiction par rapport au perceptron multicouche (PMC) et le système d'inférence neuro-floue adaptative (ANFIS) (PMC à 5 couches), ce qui signifie qu'il pourrait être mis en œuvre dans les réseaux cognitifs avec topologies physiques centralisées.

Le scénario de chaque PU est caractérisé par un ensemble unique de caractéristiques tels que la durée du paquet, le délai entre paquets, le canal l'occupation, le modèle de saut de fréquence et la bande passante de transmission.

L'article [23] présente une stratégie pour reconnaître le scénario du PU qui ne nécessite pas l'estimation des paramètres ci-dessus. Au lieu de cela, il forme un modèle CNN profond à effectuer la classification directement à partir du spectrogramme. Le fait que chaque scénario du PU a des caractéristiques uniques est reflété comme un ensemble unique de spectrogrammes par classe. Par conséquent, la détection des scénarios PU devient une tâche de classification d'image qui peut être résolu avec CNN.

L'article [24] étudie la classification et prédiction des agents utilisateurs. Il applique le modèle DBN pour améliorer la précision de la reconnaissance de l'agent utilisateur dans les RC avec un modèle centré sur l'utilisateur. Le modèle DBN fournit la classification d'un agent utilisateur principal, qui est la base de la prédiction aux deux spectres de fréquence au ralenti et tranches de temps. Les résultats expérimentaux montrent que le moteur cognitif trouve un meilleur taux de détection que le moteur RC avec apprentissage superficiel et autre stratégie traditionnelle.

II.6 Conclusion

Nous avons présenté au cours de ce chapitre l'apprentissage automatique avec ses différents types, supervisé, non- supervisé et avec renforcement, les réseaux de neurones tel que le perceptron simple et le perceptron multicouches, ensuite nous avons présenté l'apprentissage profond et ses différentes méthodes : RNN, CNN, DBN et quelques domaines d'applications dont il est utilisé, nous nous sommes aussi intéressés à l'utilisation pratique à travers quelques travaux de l'utilisation des différentes méthodes d'apprentissage profond dans la radio cognitive.

Dans le chapitre suivant, nous allons appliquer le deep learning dans un scénario bien précis de la radio cognitive afin d'assurer un partage de spectre optimal.

CHAPITRE III

Contribution et résultats

III. Chapitre III : Contribution et résultats

III.1 Introduction

L'apprentissage et la cognition illustrent le concept de la connaissance comme une caractéristique commune d'apprentissage et cognition. Ainsi, l'apprentissage est indispensable à tout système cognitif et doit être à la base de radio cognitive.

Dans ce chapitre, nous allons utiliser le deep learning pour résoudre le problème de partage de spectre. En effet, nous montrons les différentes expérimentations menées et nous discutons les résultats obtenus. En premier lieu, nous allons décrire notre scénario. Nous avons implémenté les RNN, et plus précisément LSTM pour faire la prédiction des résultats des SU que nous allons utiliser dans notre scénario.

Ensuite, nous allons faire des variations de paramètres du RNN et expliquer les différents résultats dans chaque cas. Enfin, nous allons faire une comparaison entre l'utilisation du deep learning et sans deep learning pour ce scénario.

III.2 Outils utilisés

III.2.1 Plateforme de développement (TensorFlow)

Nous avons utilisé dans notre travail TensorFlow qui est un framework de programmation pour le calcul numérique, développé par l'équipe Google Brain et qui a été rendu Open Source par Google en Novembre 2015. La version 1.0.0 a été lancée en février 2017. [25]
Depuis sa libération, TensorFlow n'a cessé de gagner en popularité, pour devenir très rapidement l'un des frameworks les plus utilisés pour le Deep Learning, comme le montrent les dernières comparaisons, faites par François Chollet (auteur de la librairie Keras). [26]
La figure ci-dessous représente une comparaison entre les différentes librairies utilisées pour le deep learning.

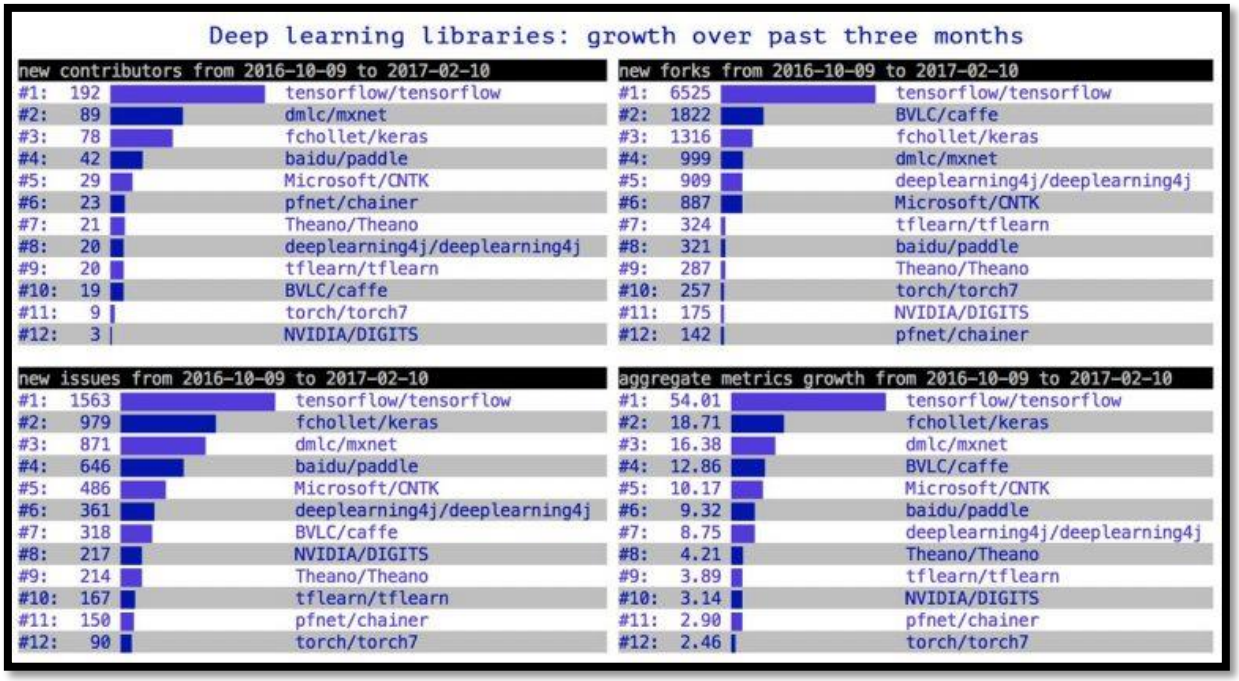

Figure III.1: Comparaison entre les différentes librairies du deep learning.

TensorFlow est très populaire pour ses nombreux avantages qu'on cite ci-dessous:

- Multi-plateformes (Linux, Mac OS, et même Android et iOS).
- APIs en Python, C++, Java et Go (**l'API Python est plus complète cependant, c'est sur celle-ci que nous allons travailler**).
- Temps de compilation très courts.
- Supporte les calculs sur CPU, GPU et même le calcul distribué sur cluster.
- Documentation extrêmement bien fournie avec de nombreux exemples et tutoriels.

TensorFlow est aujourd'hui particulièrement utilisé pour le Deep Learning, et donc les réseaux de neurones. Son nom est notamment inspiré du fait que les opérations courantes sur des réseaux de neurones sont principalement faites via des tables de données multi-dimensionnelles, appelées Tenseurs (*Tensor*). Un *Tensor* à deux dimensions est l'équivalent d'une matrice. [26]

Aujourd'hui, les principaux produits de Google sont basés sur TensorFlow: Gmail, Google Photos, Reconnaissance de voix, etc.

III.2.2 Autres outils de développement

Pycharm: Vu que l'API Python de TensorFlow est la plus complète, nous l'avons choisi pour travailler et donc c'est pour ça on a choisi Pycharm un environnement de développement dédié au langage Python.

ANACONDA: une distribution Python libre qui intègre directement un grand nombre de packages.

III.3 Travail effectué

III.3.1 Construction de la base d'apprentissage

Vu que dans la littérature, il n'y a pas de Benchmark spécifique pour la RC, nous étions obligés de jouer le rôle de l'expert pour construire le DataSet qu'on va utiliser comme entrée pour l'apprentissage.

La taille de notre DataSet en premier lieu a été de 17 entrées avec 7 caractéristiques pour chaque entrée ainsi que son label (classe).

SU	nombre de canaux demandés par le SU	prix	durée d'utilisation (minutes)	fidélité du SU	type de technologie utiliséé	Bruits (SNR)	Taux d'erreurs (%)	Classe
1	4	10	5	1	2	30	5	1
2	6	5	15	5	2	40	10	2
3	3	8	3	4	1	90	20	3
4	2	9	8	8	1	53	3	3
5	2	10	10	5	2	46	12	3
6	3	7	5	6	2	39	54	2
7	5	8	7	12	2	5	5	1
8	1	15	3	4	3	19	31	3
9	3	4	4	6	3	1	45	1
10	9	15	15	5	2	1	10	2
11	8	3	40	34	1	89	20	3
12	12	9	8	18	1	6	3	1
13	4	3	20	15	2	4	12	2
14	8	8	2	16	2	97	54	3
15	7	10	17	1	2	88	5	3
16	10	5	3	14	3	27	31	2
17	6	4	8	1	3	19	45	1

Figure III.2: Le DataSet utilisé.

Ensuite, nous étions obligés de générer des données manuellement pour augmenter la taille de notre DataSet jusqu'à arriver à 46 entrées.

III.3.2 Choix des critères

Dans les travaux de [6], les auteurs ont utilisé pour résoudre le problème du partage du spectre les critères suivants : le nombre de canaux demandés, le prix, la durée d'utilisation). Cependant, en se basant sur le travail effectué dans [27], nous pensons que pour avoir de meilleurs résultats et pour que l'apprentissage se fasse correctement il est nécessaire de considérer d'autres critères qui sont tout aussi importants que les autres.
Dans ce qui suit, nous allons expliquer l'utilité de chaque critère ajouté.

- **Fidélité du SU** : Si le PU a l'habitude de travailler avec un certain SU, une valeur de fidélité lui sera attribuée et qui sera considérée pour faire la décision.
- **Type de technologie utilisé** : Ce paramètre définit les caractéristiques de la communication utilisée par chaque SU. Ce paramètre peut prendre les valeurs suivantes (vidéo, audio, données). Dans notre dataset, les valeurs vont de 1 à 3.
- **Bruits (SNR)** : Ce paramètre est très signifiant pour la détection d'énergie et d'interférences.
- **Taux d'erreurs** : Ce paramètre représente le taux d'erreurs présent dans le canal de communication. Dans le dataset c'est un pourcentage allant de 0 à 100%. Ce paramètre peut aider à évaluer une bande de fréquence qui présente une transmission avec peu d'erreurs.
- **Classe** : peut prendre les valeurs de 1 à 3 qui correspondent à (mieux recommandé, moyen, moins recommandé).

III.3.3 Choix de l'algorithme

Les réseaux de neurones simples tels que le perceptron ou les autres types tels que les CNN n'ont aucune information sur ce qu'ils ont fait auparavant, ils partent toujours de l'état où ils ont été préalablement formés. C'est à dire qu'il n'y a pas d'évolution après la formation initiale. Cependant les RNN ont une forme de mémoire.

En effet, les RNN reçoivent leur propre sortie de l'application précédente (et transitivement de toutes les applications précédentes). De cette façon, ils peuvent transférer des données vers le «futur moi», formant ainsi une sorte de mémoire et évoluant davantage à partir de nouvelles instances d'application.

Au cours de l'entrainement du RNN, l'information circule en boucle encore et encore, ce qui entraîne de très grandes mises à jour dans le modèle de réseau neuronal et qui résulte en un réseau instable. Pour éviter ce problème, nous avons utilisé **LSTM** car ils incluent une

«cellule mémoire» qui peut conserver des informations en mémoire pendant de longues périodes. Un ensemble de portes est utilisé pour contrôler quand l'information entre dans la mémoire, quand elle est sortie, et quand elle est oubliée. Cette architecture leur permet d'apprendre des dépendances à plus long terme.

III.3.4 Scénario proposé

Figure III.3: Scénario proposé.

Dans notre scénario, nous allons traiter le cas d'avoir 1 PU et plusieurs SU qui veulent travailler avec lui; le PU veut choisir le meilleur SU. Dans ce cas, le problème qui se pose c'est comment classer ces SU afin de pouvoir choisir le meilleur entre eux ? chaque SU a des caractéristiques différentes des autres, donc il est très difficile de savoir si un SU est mieux recommandé ou bien moyen ou bien le moins recommandé.
Pour cela, nous avons choisi d'utiliser le Deep learning dans ce scénario afin de pouvoir classifier les SU et par la suite faciliter le choix pour le PU.

III.3.5 Résultats obtenus

a Démarche suivie

Pour commencer nous avons normalisé les DataSets qui sont les inputs de notre modèle dans un intervalle de [-1,1] en utilisant une certaine fonction, la normalisation est effectuée pour avoir la même plage de valeurs pour chacune des entrées. Cela peut garantir une convergence stable et précise du modèle, en plus toutes les fonctions d'activations travaillent sur l'intervalle [-1,1].

La 2ème étape constitue à préparer le modèle, pour cela nous avons utilisé la librairie Keras qui est une librairie Python qui encapsule l'accès aux fonctions proposées par plusieurs librairies d'apprentissage automatique, en particulier Tensorflow.

Après cela, nous avons créé une couche LSTM qui contient 30 neurones ensuite les neurones de cette 1ère couche sont connectés avec une deuxieme couche qui contient 128 neurones, ensuite nous avons ajouté une couche 'Dense' qui signifie que chaque neurone dans une couche dense sera entièrement connecté à chaque neurone de la couche précédente.

Nous avons utilisé aussi la fonction 'Sigmoid' comme fonction d'activation, elle est utilisée comme la fonction de déclenchement la plus utilisée dans le LSTM pour les 3 portes (in, out, forget), car elle produit une valeur entre 0 et 1, elle peut soit ne laisser aucun flux passer ou compléter le flux d'informations à travers les portes. Nous avons également utilisé l'algorithme d'optimisation 'Adam' pour mettre à jour les poids de réseau itératifs en fonction des données d'apprentissage.

Pour finir nous avons utilisé 80% des SU pour l'apprentissage (Training) et les 20% restants pour le test.

Dans le code, nous avons utilisé une certaine variable **'nb_epochs'** qui signifie nombre d'époques, une "époque" décrit le nombre de fois que l'algorithme voit l'ensemble de données en entier. Ainsi, chaque fois que l'algorithme a vu tous les échantillons dans l'ensemble de données, une époque est terminée.
1 Epoch = 1 Passe avant + 1 Passe arrière, pour TOUS les échantillons d'entraînement.

Nous avons aussi utilisé la variable ' Batch size ' qui définit le nombre d'échantillons qui vont être propagés à travers le réseau.

b Résultats

SU

								Résultat
29	0.359768	-0.019809	-0.180851	0.782324	-0.468085	-0.125000	-0.008761	-0.457447
2	-0.276596	0.014674	-0.049272	-0.038189	-0.468085	0.166667	-0.342094	0.542553
42	0.087041	0.221570	0.134938	-0.012548	0.031915	0.135417	-0.106800	0.042553
13	0.177950	0.049156	0.187570	-0.217676	0.031915	0.531250	-0.302879	0.542553
19	0.177950	-0.226706	-0.022956	-0.038189	-0.468085	0.541667	-0.342094	0.542553
39	0.087041	0.393984	-0.128219	-0.140753	-0.468085	-0.177083	-0.204839	-0.457447
4	-0.185687	-0.054292	-0.128219	-0.089471	0.031915	0.020833	0.657906	0.042553
35	-0.094778	0.049156	-0.128219	-0.166394	0.531915	0.291667	0.206925	0.042553
5	-0.003868	-0.019809	-0.075588	0.064375	0.031915	-0.333333	-0.302879	-0.457447
8	0.359768	0.221570	0.134938	-0.115112	0.031915	-0.375000	-0.204839	0.042553

Résultat de prédiction

```
[[-0.05866297]
 [-0.05383388]
 [-0.01747006]
 [ 0.30785814]
 [ 0.45635182]
 [-0.314246  ]
 [ 0.3734171 ]
 [ 0.24135779]
 [-0.43393418]
 [-0.38034362]]
```

Figure III.4: Premiers résultats.

Les SU dans la figure ci-dessus ont été utilisé pour le test, on remarque que certains résultats de prédiction (6 résultats de prédiction sur 10) sont proches des résultats attendus et les restes sont loins.

c Matrice de Confusion

	mieux recommandé	Moyen	moins recommandé
mieux recommandé	2	0	1
Moyen	2	1	1
moins recommandé	0	0	3

Figure III.5 : Matrice de Confusion

En ligne, on lit les labels des individus et en colonne les labels prédits par le modèle. Ainsi, on peut conclure par exemple que :

- Dans 2 situations où le SU est ‘mieux recommandé’, le modèle a bien prédit ‘mieux recommandé’
- Le modèle a prédit 1 fois que le SU est ‘moins recommandé’ alors qu’en réalité il est ‘mieux recommandé’
- La classe ‘moyen’ déduite par le modèle contient 1 observation bien classée et aucune mauvaise prédiction.

Egalement, à partir de cette matrice, on peut calculer deux mesures :

- **Précision :** correspond à la qualité générale du modèle. Pour cela, on va diviser les bonnes prédictions (somme de la diagonale) par le nombre total de prédiction, le tout soustrait de 1. Dans notre exemple, on a une précision égale à 40%.
  Précision = 40%
- **Rappel :** correspond à la qualité d’une classe. Il y en a une par classe donc. On va diviser le nombre d’éléments bien classés dans la classe par le nombre total d’individus appartenant réellement à la classe. Par exemple, il y a une sensibilité de 66% dans la classe ‘mieux recommandé’, 25% dans la classe ‘moyen’ et 100% dans la classe ‘moins recommandé’, ensuite la somme des sensibilités de chacune des classes divisé par le nombre de classe donne **la sensibilité du modèle** qui est égale à 63%.
  Rappel = 63%

d Discussion des résultats

La figure suivante représente la variation moyenne de la "perte" pendant l'époque durant

l'entrainement. On remarque que la perte se réduit durant l'entrainement ce qui prouve que notre modèle apprend car l'erreur diminue.

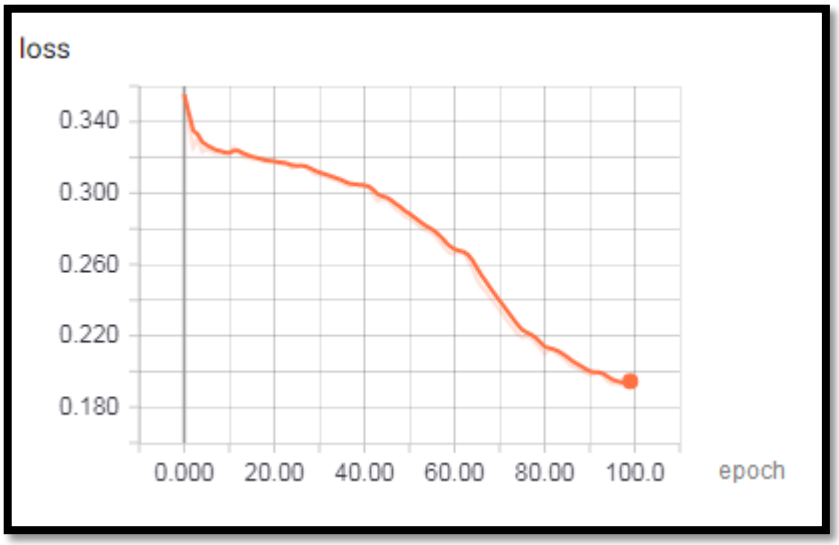

Figure III.6: Graphe de la variation moyenne de la perte.

La figure III.8 représente la variation moyenne de la " validation de perte" après la fin de l'époque durant l'entrainement. Elle diminue aussi ce qui prouve que notre modèle apprend.

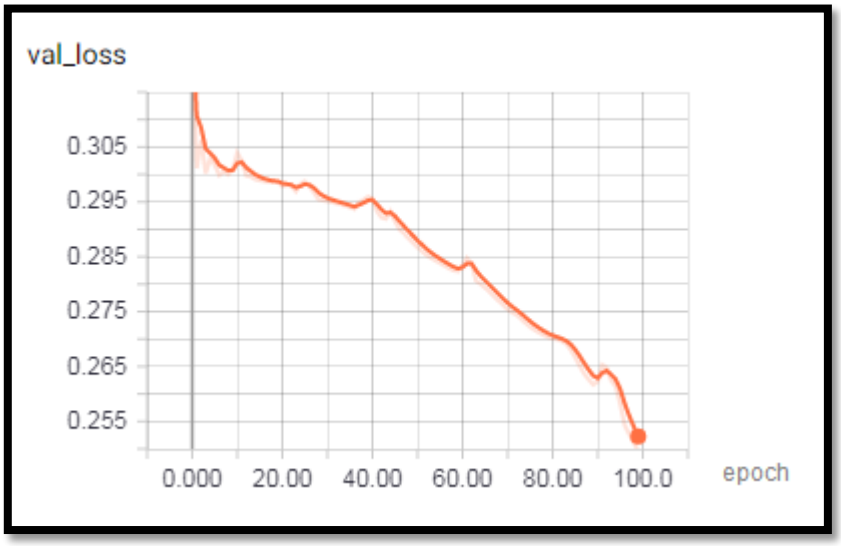

Figure III.7: Graphe de la variation moyenne de la validation de perte.

e TensorBoard

L'enchaînement des opérations peut vite devenir complexe lors de la création des réseaux de neurons et nous allons très vite ressentir le besoin de visualiser graphiquement le résultat, ainsi que de contrôler l'évolution de nos phases d'apprentissage (activités des neurones, etc.). Pour cela, TensorFlow met à disposition un outil, **TensorBoard**. TensorBoard est une réelle force et constitue un vrai élément différenciant de TensorFlow par rapport aux autres

frameworks de Deep Learning.

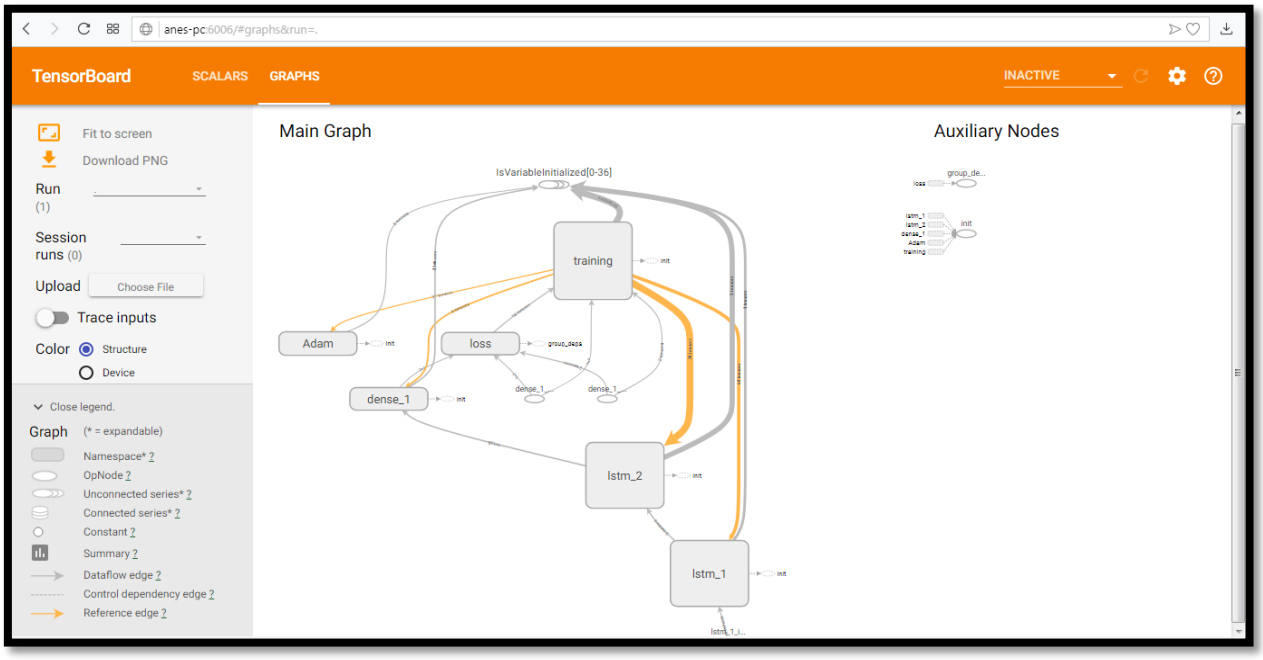

Figure III.8: Interface web de TensorBoard.

f Résultats avec variation

❖ Nombre d'epochs de 100 à 1000

- Nous avons changé la valeur de la variable '**nb_epoch**' de 100 à 1000 et nous avons obtenus les résultats suivants :

SU							Résultats	Résultats de prédiction
9	0.268859	-0.192223	0.792833	0.628478	-0.468085	0.541667	-0.008761	0.542553	[[0.5597679]
44	0.087041	0.738811	0.398096	0.166939	0.531915	0.291667	0.579474	0.042553	[0.08595479]
6	-0.367505	0.221570	-0.180851	-0.140753	0.531915	-0.187500	0.206925	0.542553	[0.3102798]
2	0.177950	0.049156	0.187570	-0.217676	0.031915	0.531250	-0.302879	0.542553	[-0.15881807]
18	-0.003868	-0.192223	-0.049272	-0.166394	-0.468085	-0.125000	-0.008761	0.042553	[0.22436802]
11	-0.094778	-0.192223	0.266517	0.141298	0.031915	-0.343750	-0.165624	0.042553	[-0.41298604]
13	0.177950	0.049156	0.187570	-0.217676	0.031915	0.531250	-0.302879	0.542553	[0.3378243]
16	-0.003868	-0.157740	0.003359	-0.115112	0.031915	-0.072917	-0.302879	0.042553	[-0.14691497]
45	0.087041	0.738811	0.398096	0.166939	0.531915	-0.229167	-0.204839	-0.457447	[-0.22771652]
10	0.632495	0.014674	-0.049272	0.218221	-0.468085	-0.322917	-0.342094	-0.457447	[-0.30196443]]

Figure III.9: Résultats après variations.

Alors ici, nous avons obtenu presque 6 sur 10 résultats de prédiction qui ressemblent aux résultats attendus. Nous remarquons que le nombre de bonnes prédiction est le même que dans le premier exemple (avec 100 epochs). Ceci est du à la taille de notre dataset qui ne contient pas beaucoup d'échantillons et donc 100 epochs sont suffisants pour l'apprentissage.

- Variation de l'erreur avec 1000 epochs :

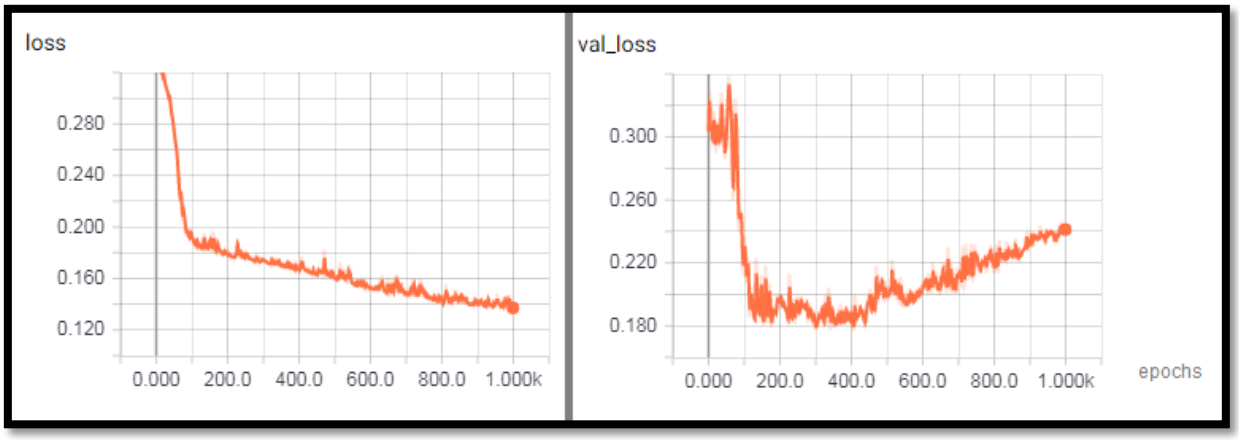

Figure III.10: les graphes après variations

Le graphe de perte nous a donné des résultats acceptable mais le graphe de la validation de perte montre qu'à partir de l'époque 400 le graphe commence à augmenter ce qui montre qu'il y'a eu un sur-apprentissage.

Matrice de Confusion

	mieux recommandé	Moyen	moins recommandé
mieux recommandé	3	1	0
Moyen	1	2	1
moins recommandé	0	0	2

Figure III.11 : Matrice de confusion de la 1ière variation

- **Précision :** on a une précision égale à 33%.
- **Rappel :** il y a une sensibilité de 75% dans la classe 'mieux recommandé', 50% dans la classe 'moyen' et 100% dans la classe 'moins recommandé', ensuite la somme des sensibilités de chacune des classes divisé par le nombre de classe donne **la sensibilité du modèle** qui est égale à 75%.

❖ Nombre d'epochs de 100 à 400 et variation du nombre de neurones

Changement de la variable '**nb_epoch**' de 100 à 400 et le nombre de neurones de chaque couche (couche 1 : 60 neurones et couche 2 : 160 neurones) alors qu'ils étaient (couche 1: 30 neurones et couche 2: 128 neurones).

SU								Résultats	Résultats de prédiction
37	0.087041	-0.226706	-0.128219	-0.140753	-0.468085	-0.177083	-0.204839	0.042553	[[-0.14061828]
4	-0.185687	-0.054292	-0.128219	-0.089471	0.031915	0.020833	0.657906	0.042553	[0.26250845]
5	-0.003868	-0.019809	-0.075588	0.064375	0.031915	-0.333333	-0.302879	-0.457447	[-0.5542627]
8	0.359768	0.221570	0.134938	-0.115112	0.031915	-0.375000	-0.204839	0.042553	[-0.47945094]
24	-0.367505	-0.123258	-0.049272	-0.063830	0.031915	-0.031250	-0.302879	-0.457447	[-0.3777269]
40	0.087041	0.221570	0.134938	-0.012548	0.031915	-0.031250	-0.342094	0.042553	[-0.27156785]
11	-0.094778	-0.192223	0.266517	0.141298	0.031915	-0.343750	-0.165624	0.042553	[-0.34175494]
22	-0.367505	-0.192223	-0.075588	-0.115112	0.031915	-0.291667	-0.302879	-0.457447	[-0.12621486]
27	-0.185687	0.152605	-0.154535	0.141298	0.031915	-0.072917	-0.302879	-0.457447	[-0.42433754]
1	-0.185687	-0.019809	-0.180851	-0.140753	-0.468085	0.552083	-0.008761	0.542553	[0.6417464]]

Figure III.12: Résultats après les 2èmes variations.

Nous avons obtenu 4 résultats de prédiction sur 10 proches des résultats attendus. ce qui montre que l'augmentation du nombre de neuronnes n'a pas donné de bonnes résultats à cause de la petite taille du dataset.

- Graphes :

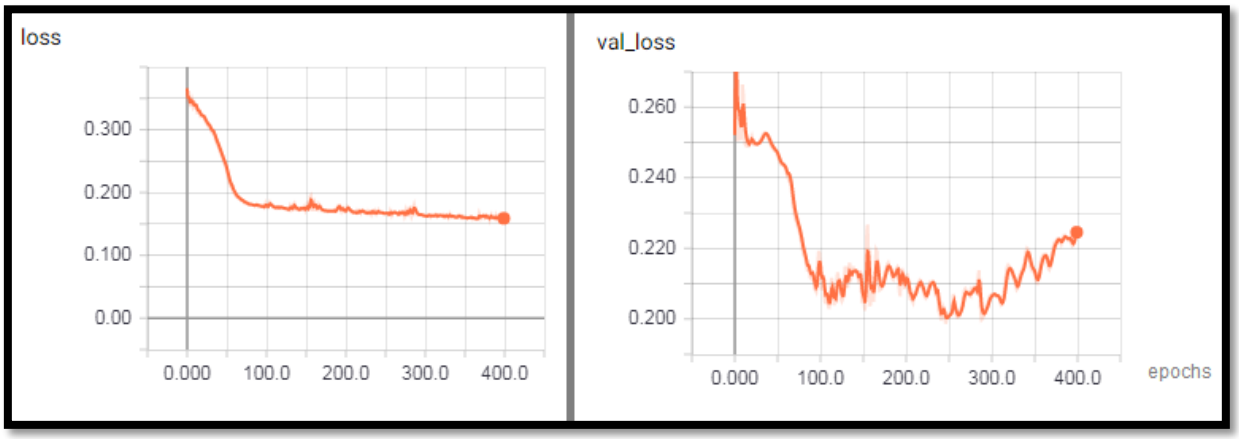

Figure III.13: les graphes après les 2emes variations

Nous remarquons que le résultat ressemble au précédent et donc il y'a eu un sur-apprentissage.

Matrice de Confusion

	mieux recommandé	Moyen	moins recommandé
mieux recommandé	1	0	0
Moyen	0	2	3
moins recommandé	0	1	3

Figure III.14 : Matrice de confusion de la 2ème variation

- **Précision :** on a une précision égale à 40%.
- **Sensibilité (rappel) :** il y a une sensibilité de 100% dans la classe 'mieux recommandé', 40% dans la classe 'moyen' et 75% dans la classe 'moins recommandé', ensuite la somme des sensibilités de chacune des classes divisé par le nombre de classe donne **la sensibilité du modèle** qui est égale à 72%.

g Comparaison avec Topsis

Topsis est un algorithme classique (programmation classique): des conditions, des boucles etc qui donne toujours les mêmes résultats pour une certaine DataSet quelque soit sa taille, il fait des calculs à base de quelques critères et donne le résultat. Alors que le RNN n'est pas un algorithme classique mais un modèle de réseau de neurones qui est capable d'apprendre et de prédire des résultats à partir du traitement des DataSet, donc il fait de la prédiction, les résultats changent car le réseau apprend à chaque entrainement et il évite toujours les erreurs commises dans l'étape précédente.

Le temps d'exécution de Topsis est minime car l'algorithme se base sur des calculs simples.

Pour le RNN, le temps d'exécution n'est pas fixe, la phase d'entrainement consomme beaucoup de temps, car le réseau fait de l'apprentissage sur une DataSet de grande taille (des milliers d'inputs). Après avoir fait un bon apprentissage et un bon modèle, la phase de test ne prend pas beaucoup de temps (millisecondes) pour donner des résultats précise et qui s'adaptent sur les inputs.

Dans notre exemple, vu que la taille du dataset n'est pas assez grande, nous ne voyons pas vraiment de différence entre les deux algorithmes en terme de temps d'exécution.

Nous précisons que les poids utilisés dans TOPSIS sont comme suit :

- Nombre de canaux demandés par le SU : 0.2

- Prix : 0.1
- Durée d'utilisation : 0.2
- Fidélité du SU : 0.1
- Type de technologie utilisé : 0.2
- Bruits (SNR) : 0.1
- Taux d'erreurs : 0.1

SU	nombre de canaux demandés par le SU	prix	durée d'utilisation (minutes)	fidélité du SU	type de technologie utiliséé	Bruits (SNR)	Taux d'erreurs (/.)	Classe	Résultat de Topsis	Classe normalisé	prédiction
29	3	13	4	15	2	30	5	1	0.3316069	-0.457447	-0.05866297
2	6	5	15	5	2	40	10	2	0.4357626	0.542553	-0.05383388
42	6	15	15	9	2	34	3	2	0.4934552	0.042553	-0.01747006
13	4	3	20	15	2	4	12	2	0.4171431	0.542553	0.30785814
19	5	6	5	15	2	13	10	2	0.3646585	0.542553	0.45635182
39	6	2	5	4	1	20	10	2	0.3749558	-0.457447	-0.314246
4	2	9	8	8	1	53	3	3	0.3033786	0.042553	0.3734171
35	2	8	8	2	2	9	54	2	0.3376568	0.042553	0.24135779
5	2	10	10	5	2	46	12	3	0.3000964	-0.457447	-0.43393418
8	1	15	3	4	3	19	31	3	0.3117696	0.042553	-0.38034362

Figure III.15: Résultats de Topsis + résultats prédiction sur quelques SU's.

Nous remarquons qu'il n'y a pas une relation ou bien une correspondance entre les résultats prévus (la classe) et les résultats de Topsis. On sait que pour choisir un résultat parmi ceux obtenus par TOPSIS, nous devons prendre la meilleure valeur et donc si on se réfère à l'exemple montré dans la figure III.14, on remarque que pour TOPSIS, le meilleur résultat est le SU n°42 alors que la classe de ce dernier est 2 (Moyen).

TOPSIS donne comme résultat une valeur supérieure pour le meilleur choix ensuite cette valeur de diminue jusqu'à arriver à la valeur minimum pour le choix le moins recommandé. Alors que le RNN donne des prédictions pour tous les SU. Il classifie chaque SU dans une classe donc on peut avoir plusieurs SU qui ont le même résultat de prédiction comme meilleur choix et autres comme moins recommandé etc.

III.4 Conclusion

Nous avons proposé dans ce chapitre un scénario qui présente un PU qui veut partager son spectre avec plusieurs SU et donc pour faire le choix de la meilleure offre et décider du SU qu'il va travailler avec, nous avons utilisé un algorithme du deep learning qui est le RNN pour faire la prédiction des meilleurs SU, ensuite nous avons présenté les résultats avec les différents graphes et nous avons vu le changement des résultats après avoir fait varier les quelques paramètres du RNN. Nous avons également montré la différence entre l'utilisation du deep learning et sa non utilisation.

Dans les futurs travaux, nous souhaitons avoir un apprentissage complet pour le modèle utilisé en utilisant des Datasets avec un nombre de SU plus important (plus de 1000 SU) pour avoir une meilleure prédiction des résultats.

Conclusion générale

La RC est une technologie émergente récemment proposée pour mettre en œuvre une certaine forme d'intelligence permettant à un terminal d'avoir des capacités d'apprentissage. Elle offre aux utilisateurs un débit et une QoS accrus, globalement une augmentation du confort dans leurs communications.

L'apprentissage est devenu une technique de base des radios cognitives. Cependant, les algorithmes d'apprentissage automatique actuels appliqués dans les RC sont tous dans l'architecture peu profonde. Contrairement à l'architecture peu profonde, l'architecture profonde est plus alignée avec la nature cognitive de l'homme. Dans ce mémoire, nous avons proposé l'utilisation du LSTM l'un des algorithmes du deep learning, dans la prédiction des classes des SU présents.

Le fonctionnement de l'algorithme de caractérisation LSTM a été testé avec un dataSet construite manuellement et comparées à l'algorithme multicritères TOPSIS.

Pour améliorer le travail réalisé dans ce mémoire, il serait intéressant de penser à utiliser les différents algorithmes du deep learning pour améliorer tout le fonctionnement de la RC. Nous aimerions également avoir la possibilité d'utiliser un dataset qui contient des milliers de données réelles pour avoir de meilleurs résultats.

Références bibliographies

[1] L. Ngom, Insa ; Diouf, “LA RADIO COGNITIVE”, mémoire de Master, 2007.

[2] B. Benmammar and A. Amraoui, “Réseaux de radio cognitive Allocation des ressources radio et accès dynamique au spectre”, rapport de recherche, 2014.

[3] A. Metref, “Contribution à l’étude du problème de synchronisation de porteuse dans le contexte de la Radio Intelligente”, thèse de doctorat, université de Rennes, 2010.

[4] E. HOSSAIN, D. NIYATO, and H. Zhu, “Dynamic spectrum access and management in cognitive radio networks”, Cambridge University Press, 2009.

[5] A. Amraoui, W. Baghli, and B. Benmammar, “Amélioration de la fiabilité du lien sans fil pour un terminal radio cognitive mobile”, Les 12èmes Journées Doctorales en Informatique et Réseaux (JDIR'11), Nov 2011, Belfort-Montbéliard, France. 2011.

[6] A. AMRAOUI, “Vers une architecture multi-agents pour la radio cognitive opportuniste”, thèse de doctorat, université de Tlemcen, 2015.

[7] B. Ahlam, L. Mohamed, and M. B. Amel, “Vers l’auto-gestion d’un réseau de radio cognitive”, mémoire de Master, 2014.

[8] B. Benmammar, “Présentation de la radio cognitive”, rapport de recherche, 2012.

[9] C. Beaulac, “Intelligence artificielle avec apprentissage automatique pour l'estimation de la position d'un agent mobile en utilisant les modèles de markov cachés”, Mémoire. Montréal (Québec, Canada), Université du Québec à Montréal, Maîtrise en mathématique 2015.

[10] Bertrand Braunschweig, “Intelligence artificielle - livre blanc,” pp. 1–21, 2016.

[11] B. Benmammar, “Apprentissage automatique”, rapport de recherche, p. 71.

[12] S.Orlhac, A.Magamadova et F.Albaux. “L’apprentissage automatique - TPE sur l’IA.” Blog Google [Online]. Available: https://sites.google.com/site/tpesurlia/accueil/iv-les-ia-dans-le-futur/dans-le-futur/la-philosophie-pour-les-ia. [Accessed: 08-Jun-2018].

[13] Al-albab and Y. Ouli, “Initiation à l’apprentissage automatique,” p. 71.

[14] A. Schmitt and B. Le Blanc, “Les réseaux de neurones artificiels,” vol. 13, 2016.

[15] N. P. Rougier, “Perceptron simple Perceptron multi-couches.”. Cours en ligne pour les Master 2 - Sciences cognitives, université de Bordeaux.

[16] Groupes de Travail, “Rapport De Synthèse: France Intelligence Artificielle,” 2017.

[17] “Une première introduction au Deep Learning – Machine Learning France.” [Online]. Available: https://blogs.msdn.microsoft.com/mlfrance/2016/04/28/une-premiere-introduction-au-deep-learning/. [Accessed: 08-Jun-2018].

[18] “Understanding LSTM Networks -- colah’s blog.” [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 08-Jun-2018].

[19] P.-A. J. Julien Krywyk, “Classification d’images : les réseaux de neurones convolutifs en toute simplicité.” [Online]. Available: https://blog.octo.com/classification-dimages-les-reseaux-de-neurones-convolutifs-en-toute-simplicite/. [Accessed: 08-Jun-2018].

[20] J. W. Kim, “Classification with Deep Belief Networks.”

[21] A. Aragon, “SPEECH ACTIVITY DETECTION IMPLEMENTATION IN HEARING AIDS USING DEEP BELIEF NETWORK,” p. 52, 2017.

[22] D. López, E. Rivas, and O. Gualdron, “Primary user characterization for cognitive radio wireless networks using a neural system based on Deep Learning,” *Artif. Intell. Rev.*, pp. 1–27, 2017.

[23] F. Paisana *et al.*, “Context-Aware Cognitive Radio Using Deep Learning,” *IEEE Int. Symp. Dyn. Spectr. Access Networks (DYSPAN), Spectr. Chall.*, pp. 1–2, 2017.

[24] Y. Cui, S. Songlin, X. Wang, D. Cheng, and H. Huang, “Deep Learning Based Primary User Classification in Cognitive Radios,” pp. 165–168, 2015.

[25] B. Kijsirikul, “Introduction to TensorFlow,” pp. 1–18, 2016.

[26] “TensorFlow & Deep Learning | Blog Xebia - Expertise Technologique & Méthodes Agiles.” [Online]. Available: https://blog.xebia.fr/2017/03/01/tensorflow-deep-learning-episode-1-introduction/. [Accessed: 08-Jun-2018].

[27] C. Hernandez, C. Salgado, H. López, and E. Rodriguez-Colina, “Multivariable algorithm for dynamic channel selection in cognitive radio networks,” *Eurasip J. Wirel. Commun. Netw.*, vol. 2015, no. 1, 2015.

Annexe

Réseaux de neurones récurrents (RNN)

&

Long Short Term Memory (LSTM)

Les réseaux de neurones artificiels traditionnels ne sont pas capables de stocker des informations. Pour ce faire, il est nécessaire de modifier la topologie en créant des structures récurrentes qui rétro-alimentent les neurones et permettent le stockage de l'information. De telles structures sont connues sous le nom de neurones récurrents. Un ensemble de ces neurones est appelé un réseau neuronal récurrent (RNN).

Un RNN permet le stockage des états suivants à différents intervalles de temps où les paramètres sont partagés entre les différentes parties du modèle, ce qui permet une meilleure généralisation. L'un des problèmes d'un RNN est la dépendance à long terme, ce qui suggère le besoin de ne pas toujours étudier les données historiques pour effectuer une tâche en cours. Cela implique qu'un RNN ne stocke que les informations apprises dans le passé et qu'il n'est pas capable de stocker de nouvelles informations à court terme.

LSTM peut être expressément conçu pour éviter le problème de dépendance à long terme en se souvenant de l'information pendant de longues périodes de temps et d'apprendre de nouvelles informations dans le présent. Les blocs LSTM contiennent des cellules de mémoire qui permettent de mémoriser une valeur pour une période de temps arbitraire et de l'utiliser si nécessaire. Il y a aussi une couche d'oubli qui peut effacer le contenu de la mémoire qui n'est pas utile. Tous les composants sont construits pour des fonctions différentiables et sont formés pendant le processus de rétro-propagation.

La structure d'un LSTM peut être représentée comme représenté sur la figure A, où la cellule mémoire est identifiée par une lettre «C», la couche d'oubli, avec un «O», la couche d'entrée, avec un «E», et le couche de sortie, avec un «S».

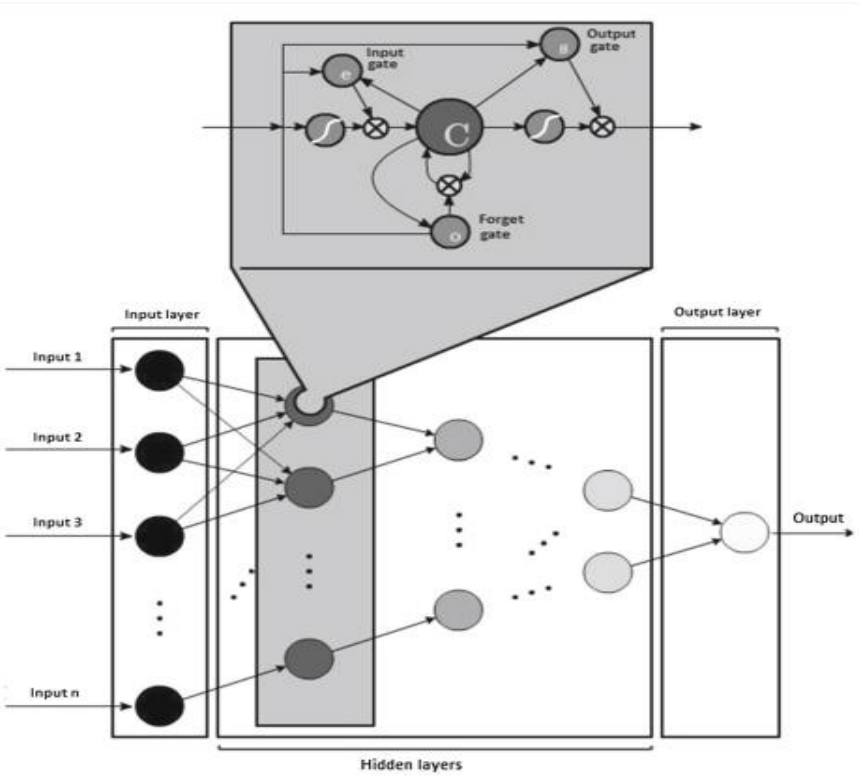

Figure A: Représentation graphique d'un réseau de neurones de type LSTM.

Afin de décrire le comportement d'une cellule, deux éléments doivent être pris en compte. Le premier est la fonction de propagation, qui dépend non seulement des inputs courant, mais aussi des outputs pour l'instant immédiatement précédent des autres blocs dans la couche cachée. Le deuxième est le statut du neurone, qui indique si le neurone garde l'information ou va l'oublier, et dépend des outputs de la porte d'oubli et la porte d'entrée.
Le neurone output indiquera si un nouvel apprentissage a été généré ou l'information stockée est gardée.
La figure B représente l'organigramme d'entrainement LSTM

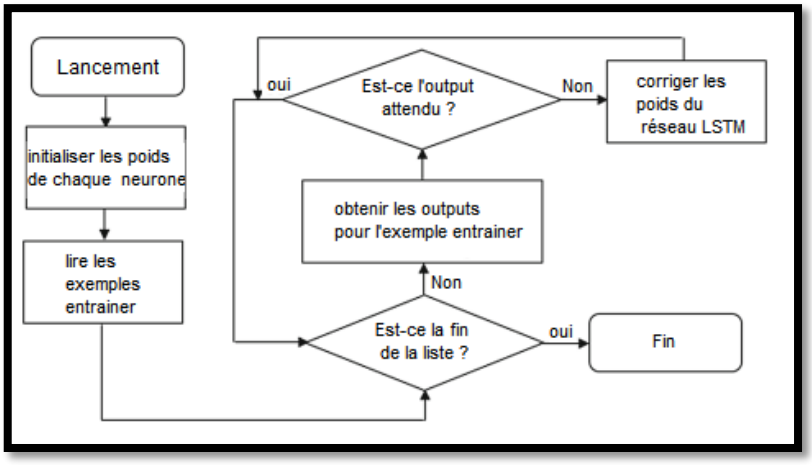

Figure B: Organigramme d'entrainement LSTM.