République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université de Mohamed El Bachir El Ibrahimi de Bordj Bou Arréridj
Faculté des Mathématiques et d'Informatique
Département d'informatique

MEMOIRE

Présenté en vue de l'obtention du diplôme

Master en informatique

Spécialité : Technologies de l'Information et de la Communication (TIC)

THEME

Classification Des Textes Par Deep Learning

Présenté par **:**

- Belguissi Mohammed El Amine
- Touati Hani

Soutenu publiquement le : 22/06/2023

Devant le jury composé de :

Président : Bendiaf Messaoud

Examinateur : Boumaza Farid

Encadreur : Mouhoub Blaazoug

2022/2023


بسم الله الرحمن الرحيم

Dédicace

Tout d'abord, je tiens à remercier DIEU

De m'avoir donné la force et le courage de mener

À bien ce modeste travail.

Je tiens à dédier cet humble travail à :

À mes chers parents qui n'ont jamais cessé de m'aimer et m'encourager durant tout mon parcours et qui m'ont permis d'être aujourd'hui ce que je suis.

À Toi ma chère mère **:** Ghania

À Toi mon cher Père **:** Mohamed Seghir

À me sœur et mon frère **:** Sara, Youcef

À toute la famille.

A mon binôme **:** Mohammed El Amine

À mon encadrant Mr Mouhoub Blaazoug.

A tous mes collègues de deuxième année master. Et à tous

MERCI A TOUS.

TOUATI HANI

Dédicace

Tout d'abord, je remercie Dieu de m'avoir permis de terminer cet humble travail

Je dédie également ce travail à **:**

Ma chère maman **:** Fatma

Cher Père **:** Belkacem

Mon oncle **:** Saleh, le « professeur » qui m'a enseigné, et sa femme

Mes chers frères et sœurs

Et toute ma famille d'ici et d'ailleurs

Je n'oublie pas non plus tous ceux qui m'ont enseigné dans mon parcours scolaire

À mon binôme **:** Hani Touati

À mon encadrant Mr Mouhoub Blaazoug.

A tous mes collègues de deuxième année master.

MERCI A TOUS.

BELGUISSI MOHAMMED EL AMINE

Remerciement

Tout d’abord on remercie « DIEU » pour nous avoir donné la force, la capacité, la volonté et le courage afin de mener à bien et à terme ce travail. Nous adressons également nos sincères remerciements à : Notre promoteur Mr BELAZOUG Mouhoub pour ses précieux conseils, son dévouement, pour son suivi et pour nous avoir aussi bien encadrées tout au long de la réalisation de ce projet. Nous tenons aussi à lui adresser notre gratitude pour tout le temps qu’il nous a consacrée, sa disponibilité ainsi que ses encouragements. Aux membres du jury pour avoir accepté de bien vouloir lire notre travail, l’examiner, l’évaluer et nous corriger. à mener à bien la réalisation de ce modeste travail.

Résumé

Au cours des dernières décennies, l'utilisation du courrier électronique s'est généralisée, entraînant des spams ou des messages frauduleux. L'intelligence artificielle (IA) et surtout le machine learning est une solution prometteuse pour classer ces messages en deux catégories**:** les messages (SPAM) et les HAM. Cependant, cette approche de classification montre des performances insatisfaisantes en raison du faible taux de réussite de la classification des messages valides. Pour améliorer cette situation, le deep learning, branche du machine learning, donne des résultats très satisfaisants.

Dans ce travail, nous allons expérimenter les algorithmes les plus utilisés pour la classification de texte ces dernières années et découvrir quel algorithme est le meilleur pour résoudre ce problème.

Mots clés : Deep learning, Machine learning, la classification des messages ,spam, ham, L'intelligence artificielle .

Abstract

In recent decades, the use of email has become widespread, leading to spam or fraudulent messages. Artificial intelligence (AI) and especially machine learning is a promising solution to classify these messages into two categories: messages (SPAM) and HAM. However, this classification approach shows unsatisfactory performance due to the low success rate of valid message classification. To improve this situation, deep learning, a branch of machine learning, gives very satisfactory results.

In this work, we will experiment with the most used algorithms for text classification in recent years and find out which algorithm is the best to solve this problem.

Keywords: Deep learning, Machine learning, message classification, spam, ham, Artificial intelligence.

ملخص

في العقود الأخيرة ، انتشر استخدام البريد الإلكتروني على نطاق واسع ، مما أدى إلى ظهور رسائل غير مرغوب فيها أو رسائل احتيالية. يعد الذكاء الاصطناعي (AI) وخاصة التعلم الآلي حلاً واعدًا لتصنيف هذه الرسائل إلى فئتين: الرسائل (SPAM)و .HAM ومع ذلك ، يُظهر نهج التصنيف هذا أداءً غير مرضٍ بسبب انخفاض معدل نجاح تصنيف الرسائل الصحيحة. لتحسين هذا الموقف ، يعطي التعلم العميق ، وهو فرع من فروع التعلم الآلي ، نتائج مرضية للغاية.

في هذا العمل ، سنقوم بتجربة الخوارزميات الأكثر استخدامًا لتصنيف النص في السنوات الأخيرة ومعرفة الخوارزمية الأفضل لحل هذه المشكلة.

الكلمات المفتاحية : التعلم العميق ، التعلم الآلي ، تصنيف الرسائل ، الرسائل المرغوب فيها ،الرسائل الغير مرغوب فيها ، الذكاء الاصطناعي.

Table des matières

Liste des abréviations

AI	Intelligence Artificielle
DL	Deep Learning
LSTM	Long Short Term Memory
CNN	Convolutional Neural Network
ANN	Artificiel Neural Network
RNN	Recurrent Neural Network..
ML	Machine Learning
SVM	Support Vector Machine
NB	Naïve Bayes
KNN	K Nearest Neighbor
DT	Decision Tree
ADAM	Adaptive Moment Estimation
NADAM	Nesterov-accelerated Adaptive Moment Estimation
SGD	Stochastic Gradient Descent

Liste des figures

Liste des tableaux

Introduction générale

Au cours des dernières décennies, le gonflement des données textuelles sur Internet et les médias sociaux ont vu une explosion de l'utilisation du courrier électronique, entraînant l'envoi de milliers, voire de millions de messages. Cependant, cette croissance a également causé des problèmes tels que l'apparition des messages malveillants, indésirables, ou ce qu'on appelle aussi : spam.

Pour résoudre ce problème, l'intelligence artificielle (IA), en particulier l'apprentissage automatique, offre une solution prometteuse. Les modèles d'apprentissage automatique peuvent classer ces messages en deux catégories distinctes : SPAM et HAM.

L'apprentissage en profondeur, ou Deep Learning en anglais est un sous-domaine de l'apprentissage automatique, présente une opportunité intéressante d'améliorer la performance de la classification des e-mails.

Dans ce travail on va présenter une étude concernant des méthodes de classification que ce soit d'apprentissage automatique et d'apprentissage profond. Cette étude est appliqué sur une base de test SMS-SPAM dans laquelle on va essayer différentes architectures comme : CNN, ARNN et Lstm du côté deep learning et de l'autre côté on va choisir les classificateurs à savoir : SVm, DTree et NB de l'approche apprentissage Automatique.

Ce mémoire est organisé comme suit **:**

Chapitre 1 : Ce chapitre commence par définir la classification des textes et explique comment obtenir une représentation mathématique ou numérique de ces textes, ce qui contribue à une meilleure classification.

Chapitre 2 : L'objectif de ce chapitre est de présenter différents algorithmes d'apprentissage automatique tels que SVM, DT et KNN, ainsi que des algorithmes d'apprentissage en profondeur tels que CNN, LSTM et ANN.

Chapitre 3 : Ce chapitre décrit l'approche que nous proposons et explique la méthodologie utilisée dans notre étude comparative des modèles de classification.

Chapitre 4 : Ce dernier chapitre est principalement consacré à la conduite et à la réalisation d'expériences. Il définit les outils et le langage de programmation utilisés dans leur mise en œuvre, et fournit une analyse détaillée des résultats obtenus.

On termine notre travail par une conclusion.

Chapitre I : Classification des textes

1. Introduction

Le progrès technologique a entraîné une augmentation des capacités de stockage numérique, ce qui a conduit à une quantité croissante de documents textuels à traiter. Aujourd'hui, avec la disponibilité croissante de textes numériques, la classification de texte est devenue un domaine en plein essor. Les techniques utilisées pour regrouper des documents similaires selon leur contenu proviennent de disciplines telles que la linguistique informatique, l'analyse de données et l'intelligence artificielle. La classification de texte peut être supervisée ou non supervisée, selon qu'elle utilise des classes préexistantes ou qu'elle découvre des classes de documents sans a priori. L'intelligence artificielle, en particulier l'apprentissage automatique, joue un rôle clé dans le développement de cette discipline. L'apprentissage automatique permet d'analyser de grandes quantités de données pour créer des modèles de prédiction qui peuvent apprendre à partir des données d'entraînement. Le chapitre 1 de ce mémoire porte sur les bases de la classification de texte, y compris le processus de catégorisation, les techniques d'apprentissage automatique, les mesures d'évaluation et l'apprentissage profond.

2. Définition de la catégorisation de textes

La catégorisation automatique de textes est d'apprendre à une machine à classer un texte dans la bonne catégorie en se basant sur son contenu.

Dans une catégorisation de texte : la classification s'apparente au problème de l'extraction de la sémantique d'un texte, puisque l'appartenance d'un document à une catégorie est étroitement liée à la signification de ce texte. **[1]**

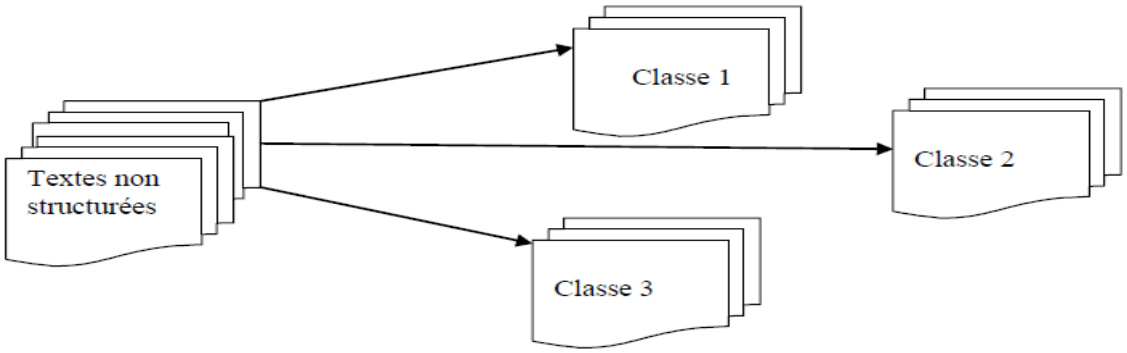


Figure 1 : Processus de classification illustré

Définition formelle : La catégorisation de texte consiste à attribuer une valeur booléenne à chaque couple (dj ,ci) ∈ D × C, D est un ensemble des textes et C = {c1, ... , C|c|} est un ensemble de catégories

fonction ϕ : D × C → {T, F} (qui décrit comment les documents doivent être classés) au moyen d'une **fonction ϕ** : D × C → {*vrai, faux*} a appelé le classificateur. **[2]**

3. Applications de la catégorisation de texte

La classification automatique est une technique largement utilisée dans différents domaines, en raison de sa rapidité et de son efficacité prédictive. Elle trouve de nombreuses applications, telles que le filtrage de spam, qui consiste à trier les messages électroniques textuels en messages désirés ou non désirés, en identifiant leurs caractéristiques.

3.1. Catégorisation des textes : un support pour différentes applications

Une autre application consiste à déterminer automatiquement le sujet d'un texte pour le classer, de manière à notifier les personnes intéressées par ce sujet de la présence d'un nouveau texte. D'autres applications incluent la catégorisation des documents multimédias, l'organisation des documents, l'indexation automatique des textes, l'analyse des sentiments et la fouille d'opinion, qui prend une importance croissante de nos jours. En utilisant cette technique de classification, il est également possible de résoudre des problèmes tels que l'identification de la langue d'un document et la résolution d'ambiguïtés dans les termes.

4. Le processus général de catégorisation de textes

4.1. L'apprentissage

Qui comprend plusieurs étapes et aboutit à un modèle de prédiction :

a) nous disposons d'un ensemble de textes étiquetés (pour chaque texte nous connaissons sa catégorie)
b) à partir de ce corpus, nous extrayons les k descripteurs (mots, termes) (t1; ..; tk) les plus pertinents au sens du problème à résoudre.
c) nous disposons alors d'un tableau « descripteurs × individus », et pour chaque texte nous connaissons la valeur de ses descripteurs et son étiquette ;
d) nous appliquons un algorithme d'apprentissage sur ce tableau afin d'obtenir un Modèle de prédiction Φ.

4.2. La phase de validation

L'objectif de la phase de validation est d'ajuster automatiquement les paramètres des algorithmes utilisés lors de la phase précédente. En effet, la plupart des algorithmes d'apprentissage, tels que les classificateurs, dépendent d'un ensemble de paramètres qui doivent être correctement fixés.

4.3. Le test

La phase de test sert à évaluer de manière définitive le processus de catégorisation à l'aide d'un jeu de test. Comme pour le jeu de validation, le jeu de test consiste en un ensemble de couples (document, catégorie) dont l'information de catégorisation n'est utilisée que pour l'évaluation du système. Étant donné que la signification d'un document est une notion subjective, l'évaluation d'un processus de catégorisation se fait empiriquement sur un jeu de test. Une fois le jeu de test catégorisé, les affectations fournies par le système sont comparées à celles définies préalablement par un expert. En conclusion, l'objectif d'un processus de catégorisation est de maximiser une fonction de score.

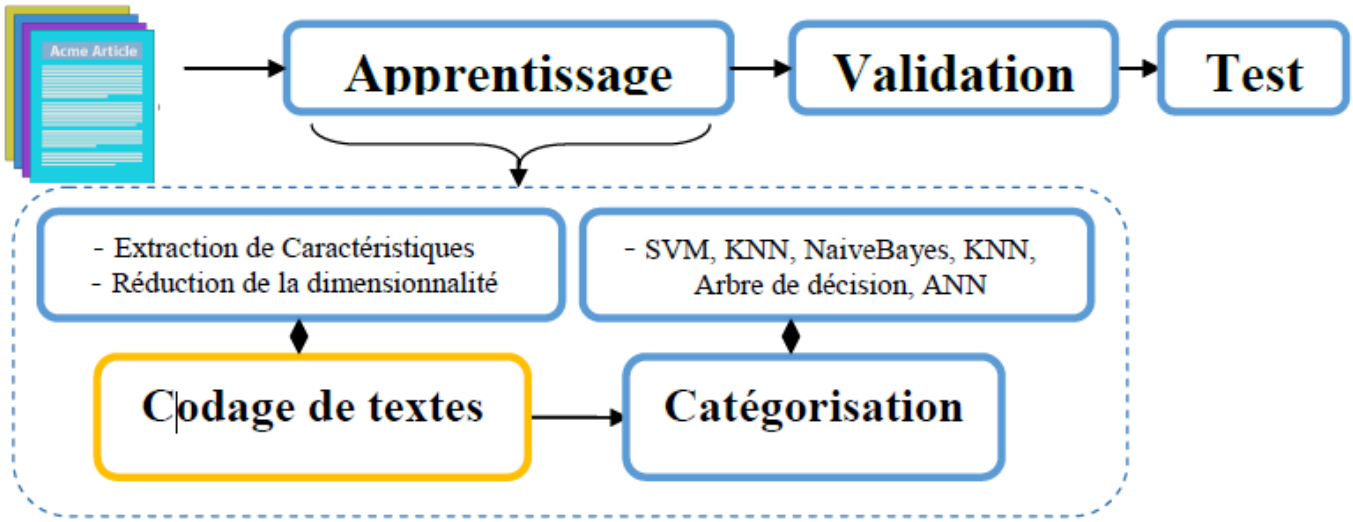


Figure 2 : Le processus de Catégorisation de textes

5. Problèmes de la catégorisation de textes

Plusieurs difficultés peuvent s'opposer au processus de catégorisation de textes. Des problèmes connus dans la discipline liés à l'apprentissage automatique supervisé comme la subjectivité de la décision prise par les experts, le sur-apprentissage, etc.. mais aussi des problèmes particuliers liés à la nature des données traitées à savoir des données textuelles comme la polysémie, la redondance, Les variations morphologiques ou même L'homographie, etc.. Dans ce qui suit nous allons signaler les dix principales difficultés qui s'opposent à la catégorisation de textes **:**

5.1. Redondance (Synonymie)

La redondance et la synonymie permettent d'exprimer le même concept par des expressions différentes, plusieurs façons d'exprimer la même chose.

Cette difficulté est liée à la nature des documents traités exprimés en langage naturel contrairement aux données numériques .Il est alors important de rassembler ces terme sen un groupe sémantique commun**.**

[03]

5.2. Polysémie (Ambiguïté)

A la différence des données numériques, les données textuelles sont sémantiquement riches, du fait qui sont conçues et raisonnées par la pensée humaine. Contrairement des langages informatiques, le langage naturel, autorise des violations des règles grammaticales engendrant plusieurs interprétations d'un même propos. Un mot possède, dans différents cas, plus d'un sens et plusieurs définitions lui sont associées**. [03]**

Par conséquent, à cause de la polysémie, les mots seuls sont parfois de mauvais descripteurs.. Le mot livre peut désigner une unité monétaire, ou un bouquin. Le mot avocat peut désigner le fruit, le juriste, ou même au sens figuré, la personne qui défend une cause. Le mot table de cuisine ce n'est pas le même que dans table de multiplication. Le mot pièce peut correspondre à une pièce de monnaie par exemple, ou à une pièce dans une maison, de même pour pavillon, bloc, glace, etc ..

5.3. L'homographie

Deux mots sont dits homographes si 'ils s'écrivent de la même façon sans forcément avoir la même prononciation. L'homographie est une sorte d'ambigüité supplémentaire. (Ex : avocat en tant que fruit et avocat en tant que juriste). **[3]**

L'homographie et l'ambiguïté génère du bruit qui va causer une dégradation de précision

(indicateur nécessaire pour mesurer la performance du classifieur). Il sera alors préférable d'ôter ces ambigüités. **[3]**

5.4. Grandes dimensions

Le modèle vectoriel proposé par consiste en la représentation de chaque texte par un vecteur dont les composants sont les termes constituant le vocabulaire du corpus. Or, pour un corpus de taille raisonnable, le tableau « termes × textes » peut avoir des centaines de milliers de lignes (textes) et des milliers de colonnes (termes) ; mais ce tableau est souvent creux, c'est-à-dire que la majorité de ses cellules sont vides. Ceci affecte

le processus d'apprentissage en rendant certains algorithmes inopérants et en augmentant le risque de sur apprentissage.**[19]**

5.5. Complexité de l'algorithme

Un texte est représenté généralement sous forme de vecteur contenant les nombres d'apparitions des termes dans ce texte. Or, le nombre de textes qu'on va traiter est très important sans oublier le nombre de termes composant le même texte donc on peut bien imaginer la dimension du tableau (textes * termes) à traiter qui va compliquer considérablement la tâche de classification en diminuant la performance du système. De ce

fait, une réduction de la taille du tableau, comme nous allons voir par la suite, est primordiale avant d'entamer l'apprentissage. **[03]**

5.5.1. Sur-apprentissage

Le sur-apprentissage survient lorsque le modèle de prédiction ne parvient pas à classifier correctement les nouveaux textes, bien qu'il ait réussi à le faire lors de la phase d'apprentissage en classifiant correctement les textes de la base d'apprentissage.

Afin de limiter le sur-apprentissage, il est nécessaire de sélectionner des termes pour réduire la dimensionnalité. Selon les expériences précédentes, le nombre de termes doit être limité par rapport au nombre de textes de la base d'apprentissage.

Certains auteurs recommandent d'utiliser au moins 50 à 100 fois plus de textes que de termes. En général, le nombre de textes d'apprentissage est limité, c'est pourquoi nous cherchons à agir sur le nombre de termes utilisés en les réduisant afin d'éviter ce sur-apprentissage, tout en évitant de pénaliser le système en supprimant des termes pertinents**.[4]**

5.5.2. Subjectivité de la décision

Parmi les problèmes classiques usuels dans le domaine de l'apprentissage supervisé c'est la subjectivité de la décision prise par les experts qui décident de la classe à laquelle le texte va être attribué. **[5]**

Certainement après la lecture du texte à classer, l'expert va trancher à quelle(s) catégorie(s) ce texte appartient en se basant sur le contenu sémantique et le contexte du texte et même en consultant d'autres textes préalablement associés à certaines classes, pour valider la décision prise qui ne peut être que subjective. Les experts humains ne lisent pas de la même manière ! Ne réfléchissent pas de la même manière ! Donc ne classent pas de la même manière ! **[5]**

Ainsi un même document peut être classé différemment par deux experts, ou encore un même document peut être classé différemment par le même expert, soumis à deux instants différents.

6. Les techniques de représentation

6.1. La représentation en sacs de mots << back of words >>

Bag of Words **(BoW),** ou sac de mots en français, est une méthode de traitement de texte qui consiste à représenter un document sous forme d'un vecteur de fréquence de mots. Le principe est de transformer chaque document en un vecteur de taille fixe, où chaque dimension représente un mot unique dans le corpus de texte, et où la valeur de chaque dimension est la fréquence du mot correspondant dans le document.

La méthode **BoW** est souvent utilisée pour des tâches de classification de texte telles que la classification de courriels en spam et non-spam, la classification de commentaires en positif et négatif, ou encore la classification de documents en différentes catégories.

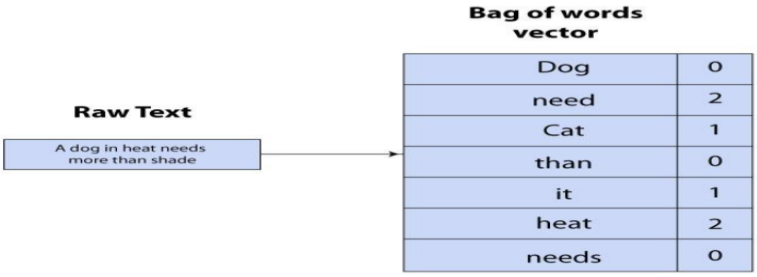


Figure 3: Exemple de sacs de mots (BOW)

6.2. Représentation avec les n-grammes

Cette méthode consiste à représenter le document par des n-grammes. Le n-gramme est une séquence de n caractères consécutifs. Elle consiste a découpé le texte en plusieurs séquence de n caractère en se déplaçant avec une fenêtre d'un caractère. **[6]**

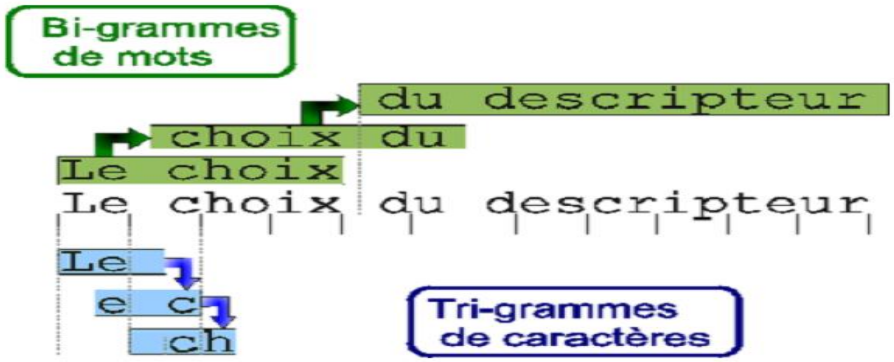


Figure 4: Exemple de n-grammes

Cette technique présente plusieurs avantages. Les n-grammes capturent automatiquement les racines des mots les plus fréquents sans passer par l'étape de recherche des racines lexicales, de plus c'est une méthode indépendante de la langue.

6.3. Représentation par phrases

Un certain nombre de chercheurs proposent d'utiliser les phrases comme unité de représentation au lieu des mots comme le cas dans la représentation « sac de mot », puisque les phrases sont plus informatives que les mots seuls, par exemple « recherche d'information », « world wide web », ont un degré plus petit d'ambiguïté que les mots constitutifs, et aussi que les phrases ont l'avantage de conserver l'information relative à la position du mot dans la phrase**.[7][8]**

7. Les études précédentes

Pendant l'année scolaire 2018/2017 à l'Université Ghardaïa, une étude a été réalisée sur la classification des textes en utilisant l'apprentissage profond. Ils ont appliqué les modèles CNN et LSTM sur un ensemble de données textuelles appelé IMDb (Internet Movie Database). Les résultats obtenus étaient les suivants **[23] :**

- **Algorithme de convolutional neural network**

Metrice	Précision	Rappel	Accuracy
Valeur de succès	0.87	0.47	0.903

Tableau 1: Résultat de CNN

- **Algorithme de long short term memory**

Metrice	Précision	Rappel	Accuracy
Valeur de succès	0.53	0.31	0.904

Tableau 2 : Résultat de LSTM

8. Conclusion

En conclusion, la classification de texte à l'aide de techniques d'intelligence artificielle et d'apprentissage automatique est un domaine vital dans l'informatique moderne. Elle permet d'analyser et de classer automatiquement de grandes quantités de données textuelles de manière plus efficace et rapide. Elle trouve des applications dans divers domaines tels que la classification des actualités, des commentaires et des e-mails, l'analyse des sentiments, les interactions sociales, ainsi que la classification de la musique, des photos et des vidéos. Cependant, l'utilisation de ces techniques nécessite de vastes ensembles de données et des algorithmes adaptés, ce qui peut être complexe. De plus, des erreurs peuvent se produire, et des améliorations constantes sont nécessaires pour obtenir de meilleurs résultats.

Dans l'ensemble, la classification de texte avec l'apprentissage automatique est un domaine fascinant qui continuera à se développer grâce aux avancées technologiques et à la disponibilité croissante des données. Ceci sera discuté plus en détail dans le deuxième chapitre.

Chapitre II: Architecture de deep learning et machine learning

1. Introduction

Un réseau de neurones artificiels, ou réseau neuronal artificiel, est un système dont la conception est à l'origine schématiquement inspirée du fonctionnement des neurones biologiques **(voir la figure 5),** et qui par la suite s'est rapproché des méthodes statistiques.

Les réseaux de neurones sont généralement optimisés par des méthodes d'apprentissage de type probabiliste, en particulier bayésien. Ils sont placés d'une part dans la famille des applications statistiques, qu'ils enrichissent avec un ensemble de paradigmes permettant de créer des classifications rapides (réseaux de Kohonen en particulier), et d'autre part dans la famille des méthodes de l'intelligence artificielle auxquelles ils fournissent un mécanisme perceptif indépendant des idées propres de l'implémenteur, et des informations d'entrée au raisonnement logique formel (Apprentissage profond).

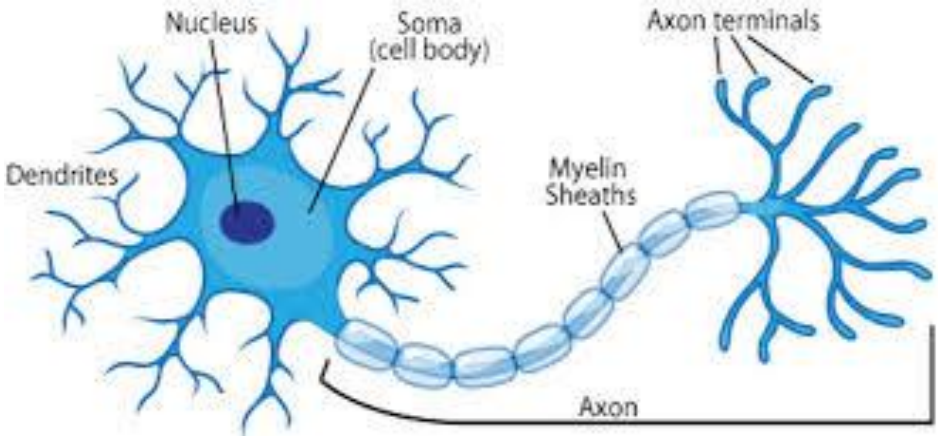


Figure 5: les réseaux de neurones

2. L'intelligence artificiel

L'intelligence artificielle (IA) est un processus d'imitation de l'intelligence humaine qui repose sur la création et l'application d'algorithmes exécutés dans un environnement informatique dynamique. Son but est de permettre à des ordinateurs de penser et d'agir comme des êtres humains.

- Pour y parvenir, trois composants sont nécessaires **:**
- Des systèmes informatiques
- Des données avec des systèmes de gestion
- Des algorithmes d'IA avancés (code)

Pour se rapprocher le plus possible du comportement humain, l'intelligence artificielle a besoin d'une quantité de données et d'une capacité de traitement élevées.

❖ L'intelligence artificielle est divisée en plusieurs types comme vous pouvez le voir **:**

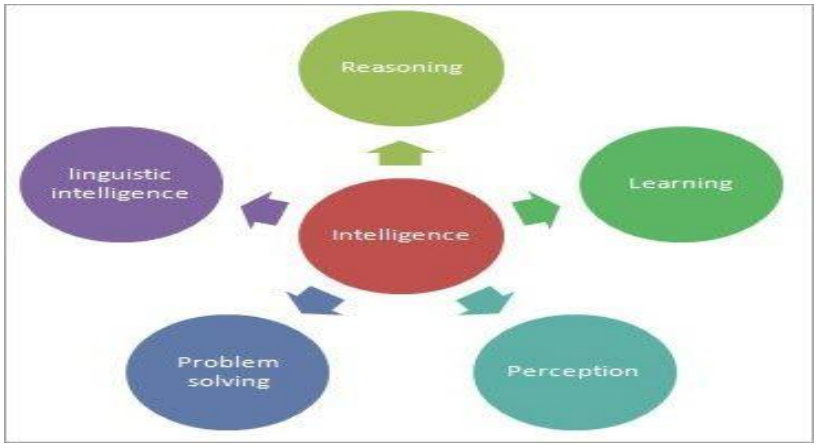


Figure 6 : les différents types de l'intelligence artificielle

L'intelligence artificielle, le deep learning, le machine learning et les données sont des éléments essentiels et interdépendants dans le domaine de l'informatique.

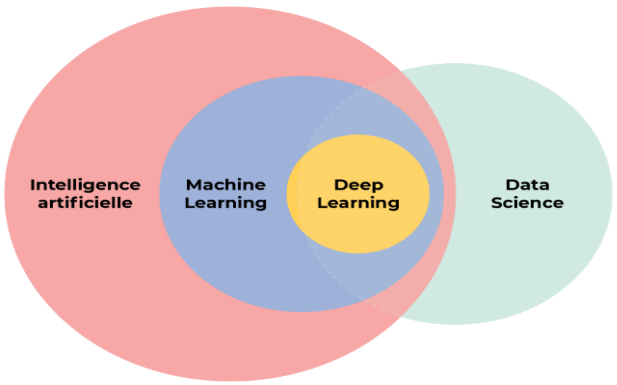


Figure 7 : Place de l'intelligence artificielle

3. Introduction à l'apprentissage automatique (en anglais : Machine learning)

3.1. L'historique de la machine learning [20]

Le paragraphe précédent résume les débuts du Machine Learning et de l'Intelligence Artificielle (IA). Le test de Turing en 1950 est souvent considéré comme le début de l'IA.

En 1959, le terme "machine Learning" a été utilisé pour la première fois par Arthur Samuel pour décrire son programme qui apprenait à jouer aux dames au fil des parties.

En 1957, le premier réseau de neurones appelé "perceptron" a été développé par Frank Rosenblatt. Cependant, les travaux de Rosenblatt n'ont pas abouti à des résultats concrets, ce qui a entraîné une période

de stagnation appelée "l'hiver de l'IA" dans laquelle la recherche en IA et en Machine Learning a été abandonnée. Dans les années 1990, l'intérêt pour l'IA et le Machine Learning a resurgi, principalement grâce à l'émergence d'Internet, qui a offert aux chercheurs de nouvelles possibilités d'interaction et un accès à de grandes quantités de données, élément essentiel pour de bonnes applications de Machine Learning

3.1. Les types d'apprentissage automatique (ML)

La classification automatique consiste à regrouper divers objets (les individus) en sous-ensembles d'objets (les classes), on distingue deux approche de classification, la classification supervisée les casses sont connues à priori, et la classification non-supervisée (en anglais clustering) les classes sont fondées sur la structure des objets.

3.1.1. Classification supervisée

Dans ce type de classification, les classes sont prédéfinies avec une description des documents. Lorsqu'un nouveau document arrive, on le compare avec la description de chaque classe et on le met dans celle qui lui ressemble le plus. Plusieurs techniques sont utilisées, on peut citer k voisins proches, arbre de décision, naïve bayes, machine à vecteur de support…

Dans ce qui suit notre travail va être concentré sur la catégorisation de textes (la classification supervisée)

3.1.2. Classification non-supervisée

Les objets sont groupés dans des classes homogènes disjointes. Pour faire ressortir les ensembles de documents, on doit maximiser l'homogénéité interne des classes et la dispersion entre elles. Les deux méthodes principales du clustering sont : les méthodes hiérarchiques et les méthodes non hiérarchiques. **[9]**

4. Les algorithmes d'apprentissage automatique (ML)

L'objectif de la phase d'apprentissage est d'induire une fonction de catégorisation à partir d'un jeu d'entraînement. On appelle jeu d'entraînement un ensemble de documents dont la catégorie est connue a priori. Pour apprendre, un système doit nécessairement intégrer de la connaissance à priori sur la nature de la solution.

4.1. Les arbres de décision

4.1.1. Définition

Les arbres de décision sont plus populaires des méthodes d'apprentissage. Les Algorithmes connus sont ID3 (Quinlan 1986) et C4.5 (Quinlan 1993). Ils sont également populaires pour la classification de document. Comme toute méthode d'apprentissage supervisée, les arbres de décision utilisent des exemples. Si l'on doit classer des documents dans des catégories, il faut construire un arbre de décision par catégorie. Pour déterminer à quelle(s) catégorie(s) appartient un nouveau document, on utilise l'arbre de décision de chaque catégorie auquel on soumet le document à classer. Chaque arbre répond Oui ou Non (il prend une décision). Concrètement, chaque nœud d'un arbre de décision contient un test (un IF...THEN) et les feuilles

ont les valeurs Oui ou Non. Chaque test regarde la valeur d'un attribut de chaque exemple. En effet, on suppose qu'un exemple est un ensemble d'attributs/valeurs. Pour des documents, chaque attribut peut être un mot et la valeur sera par exemple 0 ou 1 selon que ce mot appartient ou non au document**.[10]**

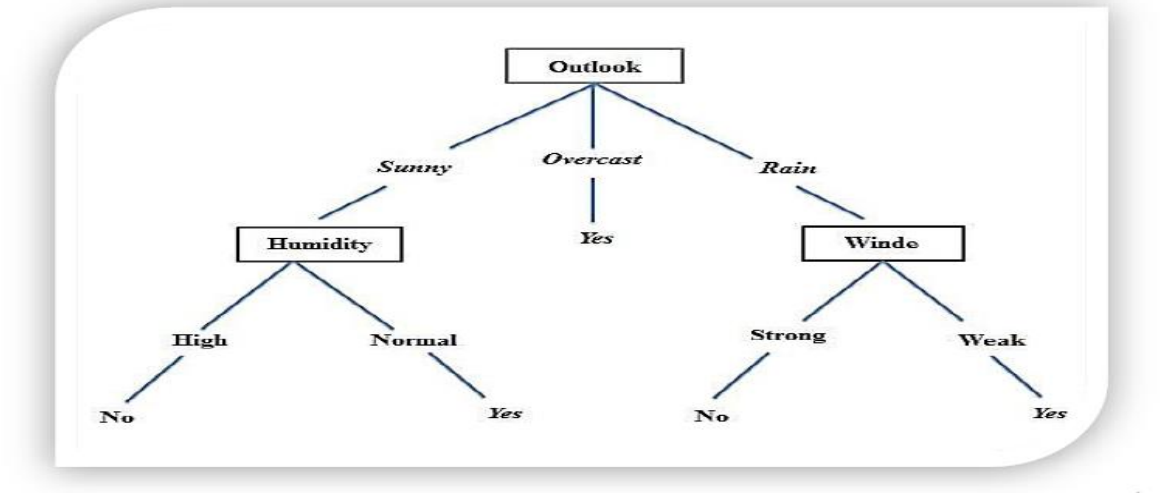


Figure 6 : l'arbre de décision

- Pour construire l'arbre de décision, il faut trouver quel attribut tester à chaque nœud. C'est un processus récursif. Pour déterminer quel attribut tester à chaque étape, on utilise un calcul statistique qui détermine dans quelle mesure cet attribut sépare bien les exemples Oui/Non. On crée alors un nœud contenant ce test, et on créer autant de descendants que de valeurs possibles pour ce test.

  Exemple : si on teste la présence d'un mot, les valeurs possibles sont Présent/Absent. A chaque fois, on aura donc deux descendants pour chaque nœud..

4.1.2. Algorithme

En général, l'algorithme d'arbre de décision se présente de la façon suivante **:**

```
Arbre ← arbre vide ; nœud_courantracine
Répéter
    Décider si le nœud courant est terminal
        Si le nœud terminal alors lui affecter une classe
        Sinon sélectionner un test et créer autant de nœud fils qu'il y a de réponse au test
        Passer au nœud suivant (s'il existe)
Jusqu'à obtenir un arbre de décision
```

4.1.3. Les domaines d'application

Cette méthode peut être utilisée dans plusieurs domaines tels que **:**

Les études (pour comprendre les critères prépondérants dans l'achat d'un produit, l'impact des dépenses publicitaires), les ventes (pour analyser les performances par région, par enseigne, par vendeur), l'analyse de

risques (pour détecter les facteurs prédictifs d'un comportement de non-paiement), Le domaine médical (pour étudier les rapports existant entre certaines maladies et des particularités physiologiques ou sociologiques)**[11]**

4.2. Algorithme des k-voisins les plus proches KNN

4.2.1. Définition [19]

L'algorithme des k-voisins les plus proches («k-nearest neighbors» ou kNN) est une méthode d'apprentissage à base d'instances.

La méthode ne nécessite pas de phase d'apprentissage ; c'est l'échantillon d'apprentissage, associé à une fonction de distance et à une fonction de choix de la classe en fonction des classes des voisins les plus proches, qui constitue le modèle.

Lorsqu'un nouveau document à classer arrive, il est comparé aux documents d'entraînement à l'aide d'une mesure de similarité. Ses *k* plus proches voisins sont alors considérés : on observe leur catégorie et celle qui revient le plus parmi les voisins est assignée au document à classer. C'est là une version de base de l'algorithme que l'on peut raffiner. Souvent, on pondère les voisins par la distance qui les sépare du nouveau texte.

4.2.2. Principe de fonctionnement

L'algorithme de KNN comparé avec ceux déjà classés en cherchant ses ***K*** plus proches voisins. Une fois ces derniers déterminés, le nouveau document est classé dans la catégorie qui inclut le maximum de voisins parmi les ***K*** trouvés**.[12]**

Deux paramètres sont utilisés : le nombre ***K*** et la fonction de similarité pour comparer le nouveau document à ceux déjà classés telle que la distance euclidienne par exemple qui est donnée par l'équation suivante **:**

$$d\,(x, y)=\sqrt{\sum_{j=1}^{m}(x_j - y_j)_2}$$

La fonctionnement de l'algorithme KNN

Paramètre : le nombre K de voisins

Contexte : un échantillon de L textes classés en C = C_1 ,C_2,…, C_n classes

Début

Pour chaque texte T faire

Transformer le texte T en vecteur T = (x_1,$x_{2, \ldots..}$, x_m),

Déterminer les K plus proches textes du texte T selon une métrique de distance,

Combiner les classes de ces K exemples en une classe C

Fin pour

Fin

Sortie : le texte T associé à la classe C

La distance entre un texte et ses voisins se fait via une métrique de distance. Cette métrique peut être comme suit :

- **Mesure Cosinus** qui consiste à calculer le produit scalaire entre deux vecteurs ***a*** et ***b***, que nous divisons par le produit de la norme de ces deux vecteurs. La formule de la mesure Cosinus est :

$$\textit{Cosinus (a, b)} = \frac{\sum(\mathbf{a}*\mathbf{b})}{\sqrt{\sum \boldsymbol{a}^2 * \sum \boldsymbol{b}^2}}$$

- **Mesure de *Distance euclidienne*** La formule de la mesure de Distance est comme suivante :

$$\textit{D (a, b)} = \sqrt{\sum |\mathbf{a} - \mathbf{b}|^2}$$

- **Mesure de Jaccard** La formule de la mesure de Jaccard est **:** $\boldsymbol{\Sigma(a*b)\Sigma a2+\Sigma b2-\Sigma ab}$

$$\textit{J(a , b)} = \frac{\sum(\mathbf{a}*\mathbf{b})}{\sum \mathbf{a}^2 + \sum \mathbf{b}^2 - \sum \mathbf{ab}}$$

4.2.3. Les domaines d'application

La méthode peut s'appliquer dès qu'il est possible de définir une distance sur les champs. Or, il est possible de définir des distances sur des champs complexes tels que des informations géographiques, des textes, des images, et du son. C'est parfois un critère de choix de la méthode *K-PPV* car les autres méthodes traitent difficilement les données complexes. On peut noter, également, que la méthode est robuste au bruit

4.3. Machines à support de vecteurs (ou SVM)

4.3.1. Définition [11]

Les machines à support de vecteurs (SVM) sont à l'origine de nouvelles méthodes de catégorisations, bien que les premières publications sur le sujet datent des années 60. Avant d'aborder le principe de fonctionnement général des SVM voici quelques notions de base **:**

- **Hyperplan** : est un séparateur d'objets des classes. De cette notion, nous pouvons dire qu'il est évident de trouver une mainte d'hyperplans mais la propriété délicate des SVM est d'avoir l'hyperplan dont la distance minimale aux exemples d'apprentissage est maximale, cet hyperplan est appelé L'hyperplan optimal, et la distance appelée marge.

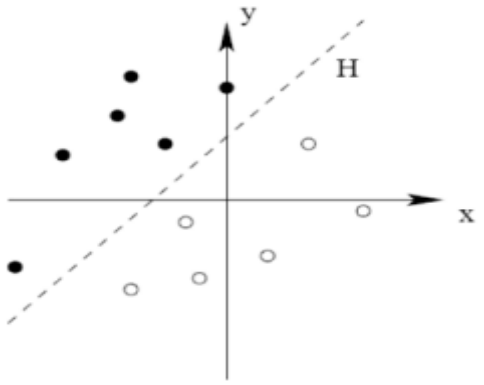


Figure 7 : la séparation du l'hyper plan par les SVM

- **Vecteurs Support** : ce sont les points qui déterminent l'hyperplan tels qu'ils soient les plus proches de ce dernier

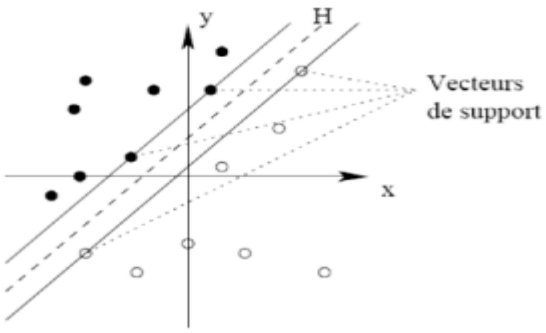


Figure 8 : Le vecteur de support

4.3.2. Principe de fonctionnement

Le principe des SVM consiste en une stratégie de minimisation structurelle du risque mais le problème revient à trouver une frontière de décision qui sépare l'espace en deux régions, à trouver

L'hyperplan qui classifie correctement les données et qui se trouve le plus loin possible de tous les exemples. On dit qu'on veut maximiser la marge qui veut dire la distance du point le plus proche de l'hyperplan.

Dans le cas de la catégorisation des textes, les entrées sont des documents et les sorties sont des catégories. En considérant un classificateur binaire, on voudra lui faire apprendre l'hyperplan qui sépare les documents appartenant à la catégorie et ceux qui n'en font pas partie **[13].**

5. L'apprentissage profond (en anglais : deep learning)

5.1. L'historique de l'apprentissage profond [20]

L'évolution des réseaux de neurones artificiels depuis leur première proposition en 1943. Il mentionne l'apparition du premier modèle de "neurone formel" binaire, suivi du développement du "perceptron" en 1957, marquant le début du réseau de neurones artificiels. Il évoque ensuite l'"hiver

De l'IA" et les avancées ultérieures, telles que le "perceptron multicouche" dans les années 1980. L'introduction du terme "deep Learning" est attribuée aux travaux de Yann LeCun sur les réseaux de neurones convolutifs. Cependant, leur utilisation était limitée en raison des exigences élevées en termes de puissance de calcul. En 2012, le deep Learning a connu un regain d'intérêt grâce à un programme gagnant lors d'un défi de reconnaissance visuelle. Depuis lors, la majorité des entreprises utilisant le machine Learning se tournent vers le deep Learning dans divers domaines.

5.2. Définition

L'apprentissage profond (« deep Learning ») est un ensemble de techniques d'apprentissage automatique qui a permis des avancées importantes en intelligence artificielle dans les dernières Années. Dans l'apprentissage automatique, un programme analyse un ensemble de données afin de tirer des règles qui permettront de tirer des conclusions sur de nouvelles données. L'apprentissage profond est basé sur ce qui a été appelé, par analogie, des « réseaux de neurones artificiels », composés de milliers d'unités (les « neurones ») qui effectuent chacune de petites opérations simples. Les résultats d'une première couche de « neurones » servent d'entrée aux calculs d'une deuxième couche et ainsi de suite. Par exemple, pour la reconnaissance visuelle, des premières couches d'unités identifient des lignes, des courbes, des angles… des couches supérieures identifient des formes, des combinaisons de formes, des objets, des contextes… Les progrès de l'apprentissage profond ont été possibles notamment grâce à l'augmentation de la puissance des ordinateurs et au développement de grandes bases de données (« big data »).**[14]**

5.3. Domaines d'application de l'apprentissage profonde

Ces techniques se développent dans le domaine de l'informatique appliquée aux NTIC (reconnaissance visuelle — par exemple d'un panneau de signalisation par un robot ou une voiture autonome — et vocale notamment) à la robotique, à la bio-informatique, la reconnaissance ou comparaison de formes, la sécurité, la santé, etc…, la pédagogie assistée par l'informatique, et plus généralement à l'intelligence artificielle. L'apprentissage profond peut par exemple permettre à un ordinateur de mieux reconnaitre des objets hautement déformables et/ou analyser par exemple les émotions révélées par un visage photographié ou filmé, ou analyser les mouvements et position des doigts d'une main, ce qui peut être utile pour traduire le langage des signes, améliorer le positionnement automatique d'une caméra, etc… Elles sont utilisées pour certaines formes d'aide au diagnostic médical (ex. : reconnaissance automatique d'un cancer en imagerie médicale), ou de prospective ou de prédiction (ex. : prédiction des propriétés d'un sol filmé par un robot).

6. Principes de fonctionnement

Les réseaux d'apprentissage en profondeur sont formés sur la base des structures de données complexes qu'ils rencontrent. Ils construisent des modèles informatiques composés de plusieurs couches **(couche d'entrée, couche cachée, couche de sortie)** de traitement pour créer plusieurs niveaux d'abstraction pour représenter les données**.**

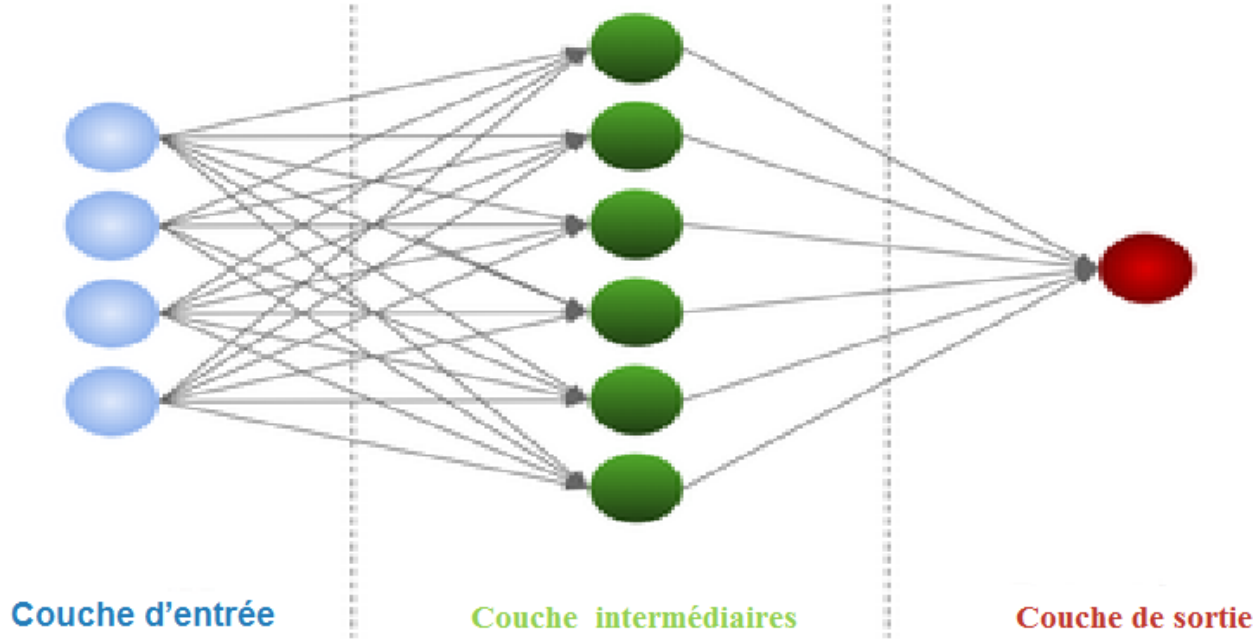


Figure 9 : les couches des réseaux profonds

6.1. Type des couches dans les réseaux profonds

Un réseau de neurones artificiels profond est constitué **:**

- **Couche d'entrée**

Composée d'un ensemble de neurones qui favoriseront la propagation des informations dans le réseau neuronal. Elle comprend un nombre de neurones généralement égal au nombre de caractéristiques constituant l'enregistrement en entrée (p. ex. une image, une transaction, etc.). Ainsi, pour un réseau de neurones qui traitera des images de 32 pixels sur 32 pixels, le nombre de neurones de la couche d'entrée serait 32 X 32 X 3, soit une valeur de 1024 pixels X 3 (représentant les nuances RGB, soit du rouge, du vert et du bleu). Le nombre de neurones à la couche d'entrée équivaudra donc à 3072 neurones.

- **Couche de sortie**

Formée d'un ensemble de neurones représentant les différentes classes de résultat. Dans une problématique de classification, les classes sont les différentes possibilités que le résultat peut offrir. Par exemple, dans un réseau de neurones profond qui sert à classer une collection
De photos en une classe « photos de chats » et une classe « photos de chiens », la couche de sortie comportera deux neurones.

- **Couches intermédiaires**

Servant à traiter l'information propagée dans le réseau de neurones pour capturer les caractéristiques de son apprentissage. Par exemple, pour un réseau de neurones dont l'objectif

est d'identifier les chats sur des images, les patrons regrouperont les caractéristiques communes qui permettront d'identifier un chat sur une image.

7. Les poids

Les poids des flèches sont utilisés pour accorder de l'importance à certaines fonctionnalités par rapport à d'autres, afin d'obtenir les résultats souhaités. La somme pondérée de tous les poids des flèches est calculée pour chaque neurone d'une couche c2achée, et chacun de ces neurones exécute une fonction d'activation qui lui est propre.

8. La fonction d'activation

Une fonction d'activation, qui associe à chaque valeur agrégée une unique valeur de sortie dépendant du seuil.

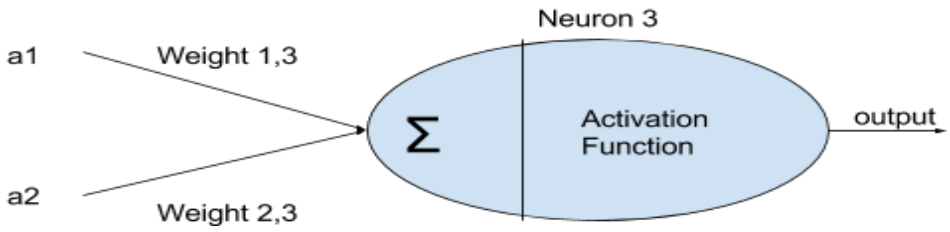


Figure 10 : Fonction d'activation

8.1. La fonction d'identité

La fonction basique d'un neurone artificiel est d'effectuer une somme de toutes les données d'entrées afin de produire une fonction de sortie**.[16]**

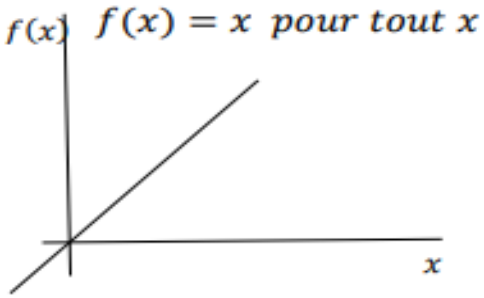


Figure 11 : Fonction d'identité

8.2. Fonction d'un seuil θ

Une autre fonction d'activation est la fonction d'un seuil. La valeur de sortie est 1 lorsque la somme pondérée des valeurs d'entrée est

Supérieure ou égale à θ, sinon la valeur est 0. **[16]**

$$f(x) = \begin{cases} 1 \ si \ x \geq \theta \\ 0 \ si \ x < \theta \end{cases}$$

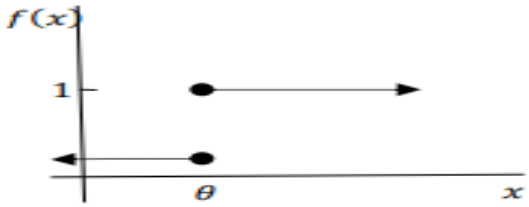


Figure 12 : Fonction d'un seuil θ

8.3. La fonction sigmoïde

De 0 à 1 est souvent utilisée comme fonction d'activation pour les réseaux de neurones dans lesquels les valeurs de sortie désirées sont soit binaires soit dans un intervalle compris entre 0 et 1. **[16]**

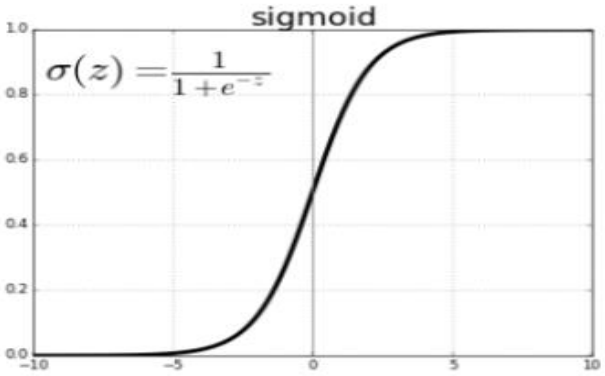


Figure 13 : fonction sigmoïde

9. Propagation vers l'avant

La propagation vers l'avant est le moyen de passer de la couche d'entrée (à gauche) à la couche de sortie (à droite) dans le réseau de neurones.

10. Propagation des mots en retour

La propagation vers l'arrière est la méthode préférable pour ajuster ou corriger les poids afin d'atteindre la fonction de perte minimisée.

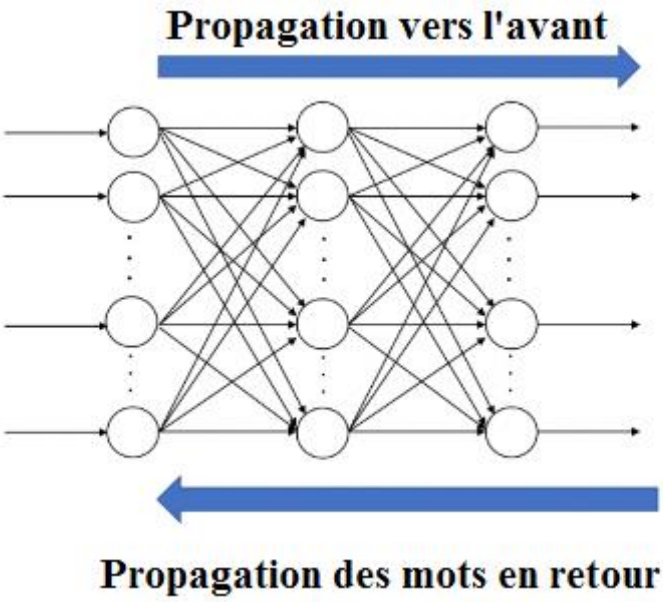


Figure 14 : le concept de back propagation

11. Les différent type de model deep learning

Plusieurs types de réseaux de neurones sont utilisés en apprentissage profond. Chaque réseau de neurones possède ses propres spécificités et applications.

Les différents types de réseaux de neurones disponibles à ont des modèles très utilisé et très célèbre dans le domaine Deep Learning, tels que les réseaux de neurones Convolutifs (CNN), les réseaux de neurones récurrents (RNN) et (LSTM « Long Short Term Memory »).

11.1. Le réseau de neurone convolutif (CNN) [21]

Les CNN, également appelés ConvNets, sont constitués de plusieurs couches et sont principalement utilisés pour le traitement d'images et la détection d'objets. Les CNN sont largement utilisés pour identifier des images satellites, traiter des images médicales, prévoir des séries chronologiques et détecter des anomalies.

Les CNN ont plusieurs couches qui traitent et extraient les caractéristiques des données **:**

- **Couche de convolution :** Le CNN possède une couche de convolution qui comporte plusieurs filtres pour effectuer l'opération de convolution.
- **Unité linéaire rectifiée (ReLU) :** Les CNN ont une couche ReLU pour effectuer des opérations sur les éléments. La sortie est une carte de caractéristiques rectifiée.
- **Couche de pooling :** Ce type de couche est souvent placé entre deux couches de convolution : elle reçoit en entrée plusieurs feature maps, et applique à chacune d'entre elles l'opération de pooling. Une couche de pooling, agit comme une couche de réduction. Elle divise l'image en blocs et ne garde que le maximum de chaque bloc. Cela permet de réduire la dimension de l'image tout en conservant

Les caractéristiques les plus importantes. On obtient en sortie le même nombre de feature maps qu'en entrée, mais celles-ci sont bien plus petites.

- **Couche fully connected :** constitue toujours **la dernière couche** d'un réseau de neurones, convolutif ou non, elle n'est donc pas caractéristique d'un CNN. Ce type de couche reçoit un vecteur en entrée et produit un nouveau vecteur en sortie. Pour cela, elle applique une combinaison linéaire puis éventuellement une fonction d'activation aux valeurs reçues en entrée.

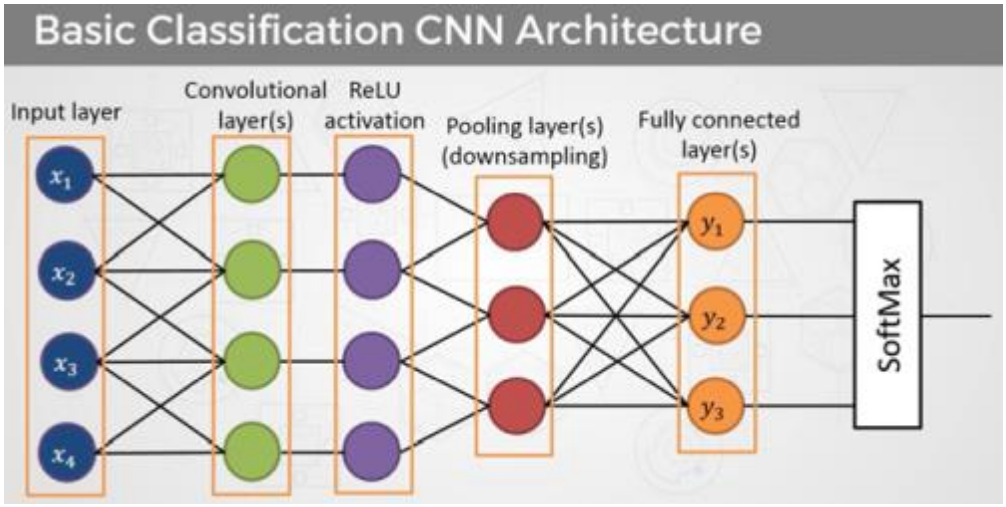


Figure 15 : Basic classification CNN architecture

11.2. Réseau de neurones récurrents (RNN) [22]

Les réseaux de neurones récurrents (RNR ou RNN : Recurrent Neural Network, en anglais) font partie des algorithmes de deep learning (Pour rappel des principaux concepts en deep learning, vous reportez à notre premier article sur ce sujet : Mieux comprendre le Deep Learning appliqué à la reconnaissance d'images).

Les réseaux de neurones récurrents ont une architecture d'apprentissage profond capable de prédire des séries temporelles ou des séquences.

Cette architecture est largement utilisée dans des applications récentes et dans de nombreux domaines. Nous citons par exemples, la prévision de la météo, l'analyse des prix des actions en bourse, le traitement automatique du langage naturel (NLP) pour faire de la traduction automatique, de la reconnaissance automatique de la parole, de l'analyse de sentiments à partir d'un texte ou d'un fichier audio.

Les réseaux de neurones récurrents sont capables de travailler sur des séries de longueurs quelconques plutôt que sur des entrées de taille figée comme c'était le cas avant l'émergence des RNNs. Pour faire une prédiction performante, comme pour tout algorithme d'intelligence artificielle, les modèles doivent être entraînés sur des données historiques robustes et généralisables

11.2.1. Principe de fonctionnement réseau des neurones récurrents

Un RNN est très similaire à un réseau de neurones non bouclé classique, dans lequel le flux des activations se dirige dans un sens unique, depuis la couche d'entrée vers la

couche de sortie. Son architecture est composée d'une couche d'entrée, de couches cachées et d'une couche de sortie.

En revanche, le RNN se distingue par la capacité de certaines connexions à revenir en arrière dans le réseau.

Expliquons le principe de fonctionnement du RNN à partir d'un RNN constitué par un seul neurone.

Ce neurone reçoit des entrées et produit une sortie. Dans le détail, à chaque étape temporelle t, ce neurone récurrent reçoit le vecteur d'entrée x(t) ainsi que sa propre sortie produite à l'étape temporelle précédente y(t-1). Nous pouvons représenter ce réseau le long d'un axe du temps. On dit alors qu'on a « déplié le réseau dans le temps ».

A chaque étape temporelle t, chaque neurone récurrent reçoit à la fois le vecteur d'entrée x(t) et le vecteur de sortie de l'étape temporelle précédente y(t-1).

Chaque neurone récurrent possède 2 types de poids **:**

- des poids (notés W sur le schéma ci-dessous) reliant les entrées à la sortie (comme pour un réseau de neurones classique)
- des poids (notés R) entre la sortie et l'entrée de la couche, qui sont les connexions récurrentes.

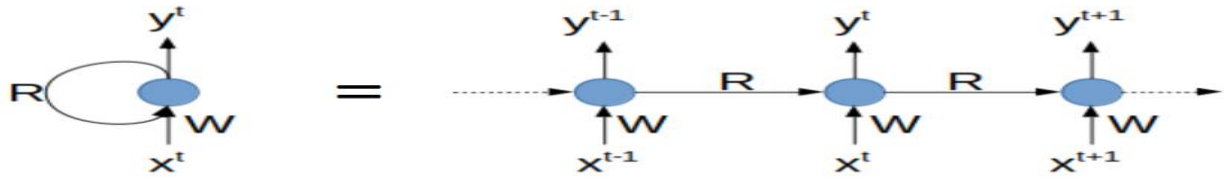


Figure 16 : RNN et sa version dépliée dans le temps

Y(t) est une fonction de **X(t)** et de **Y(t-1),** qui est une fonction de **X(t-1)** et de **Y(t-1),** qui elle-même est une fonction de **X(t-2)** et de **Y(t-3),** … Par conséquent, Y(t) est une fonction de toutes les entrées depuis l'instant **t=0**. Lors de la première étape temporelle, à t=0, les sorties précédentes n'existent pas (en général, sont supposées nulles).

en fait, la sortie d'un neurone récurrent étant une fonction de toutes les entrées des étapes précédentes, on considère que ce neurone possède une forme de mémoire.

On appelle cellule de mémoire, la partie du réseau de neurones récurrent qui conserve un état entre plusieurs étapes temporelles. L'état d'une cellule à l'étape t, noté **h(t)** (h pour « hidden » layer qui signifie couche

cachée), est une fonction de certaines entrées à cette étape temporelle et de son état à l'étape temporelle précédente **: h(t) = f(h(t-1), x(t)).**

La sortie est donc fonction de l'état précédent et des entrées courantes.

11.3. Les réseaux LSTM

Les réseaux de mémoire à long terme à court terme généralement appelés simplement (LSTM : Long Short Term Memory) sont un type spécial de RNN. Ils ont été introduits par Hochreiter Schmidhuber en 1997. Les Réseaux neuronaux récurrents présentés dans la section précédente sont capables d'apprendre des règles de mise à jour de séquence arbitraire en théorie. Dans la pratique, cependant, ces modèles oublient rapidement le passé . C'est ce qu'on appelle le problème de la disparition de gradient et c'est pourquoi ils ont inventé le LSTM . La cellule LSTM est une adaptation de la couche récurrente qui permet aux signaux plus anciens des couches profondes de se déplacer vers la cellule du présent .La figure suivante représente une chaîne de trois cellules LSTM**[16] :**

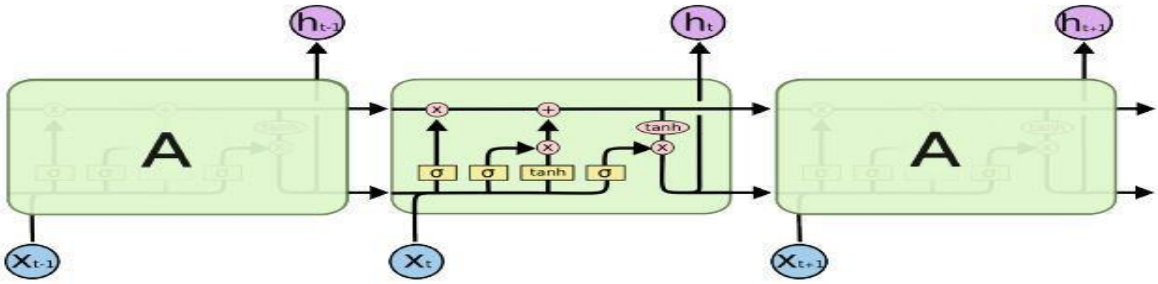


Figure 17 : Chaîne de cellules LSTM

Les calculs se déroule comme suit [17] :

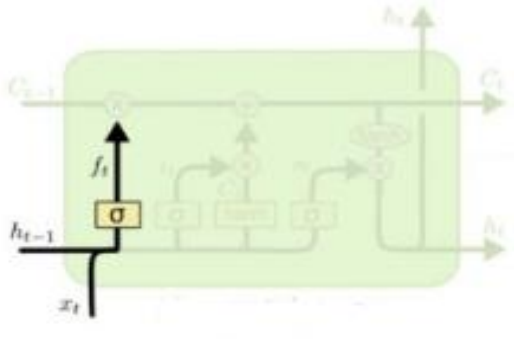


Figure 18 : opérateur d'oubli d'informations

$ft=\sigma\ (Wf.\ [ht-1\ ,\ xt]+bf)$

Tel que :
$ht-1$ **:** Sortie à l'instant $t-1$
xt **:** Entrée courant à l'instant t
b **:** Bais
w **:** Poids
σ **:** Fonction sigmoïde

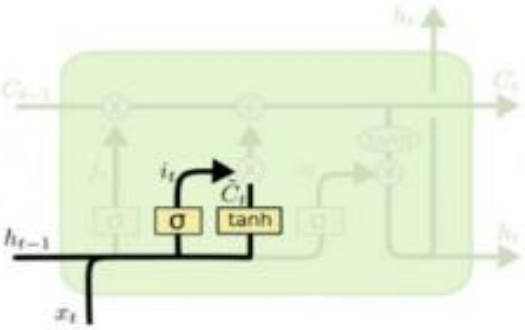


Figure 19: Opérateur d'ajout d'informations

$it=\ \sigma\ (\ .[ht-1\ ,\mathrm{x}t]+bi)$

$\tilde{Ct}=$ tanh $(WC.[ht-1\ ,xt]+bC)$

Tel que :

tanh : Fonction d'activation tangente hyperbolique

$\tilde{Ct}$: Valeur candidate

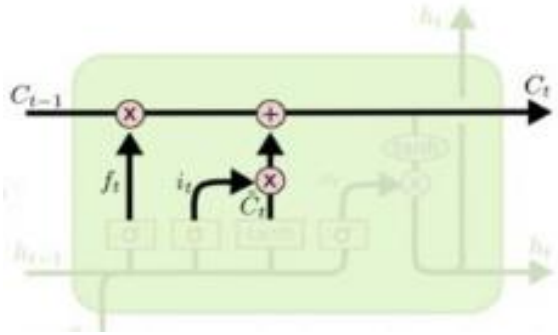


Figure 20: Mise à jour de la mémoire $\mathbf{c_t}$

$\boldsymbol{Ct}= ft*Ct-1+it*C\tilde{t}$

Tel que :

$\mathbf{C_t}$**:** État interne

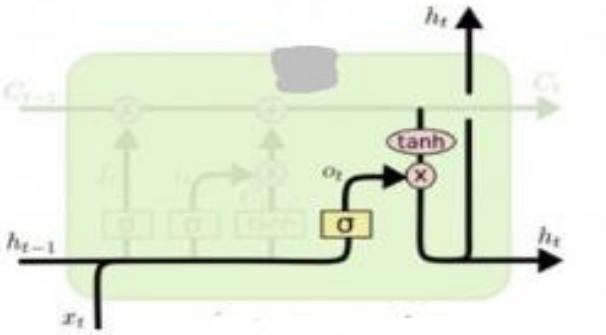


Figure 21: Sortie de la couche cachée

$\boldsymbol{ot}= \sigma\ (W0[ht-1\ ,xt]+b0)$

$\boldsymbol{ht}=ot*\ \tanh(Ct)$

Tel que :

***h*:** Sortie

Les réseaux neuronaux mentionnés, ne sont pas tous les réseaux qui existent

11.4. Réseaux de neurones Artificiel (ANN)

11.4.1.1. Définition

C'est une structure constituée de suite successive de couches de
Nœuds et qui permet de définir une fonction de transformation non linéaire des vecteurs D'entrées (composés dans le cas de classification des mots pondérés de leur poids) en vecteur de catégories. La disposition des neurones dans le réseau ainsi que le nombre de couches Utilisées ont une influence sur le résultat de classification.
Comparés aux autres méthodes de classification par apprentissage supervisé,

Les réseaux de neurones artificiels sont habituellement utilisés pour des tâches de Classification. Par analogie avec la biologie, ces unités sont appelées neurones formels.
Un neurone formel est caractérisé par **:**

- Le type des entrées et des sorties.
- Une fonction d'entrée.
- Une fonction de sortie.

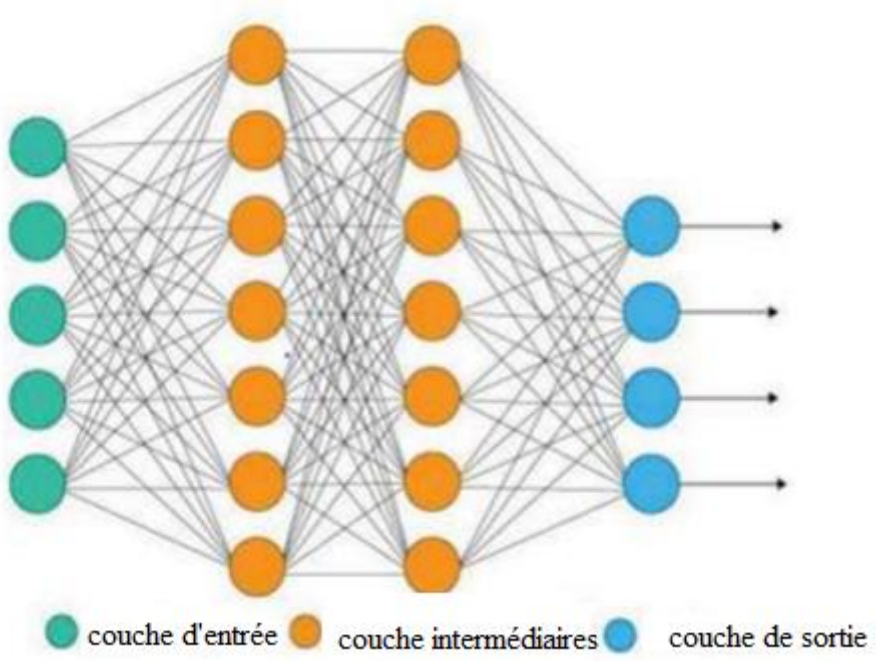


Figure 22 : Artificial neural network architecture

12. Machine learning vs deep learning

Concrètement, le deep Learning n'est qu'un sous-ensemble du machine Learning. En fait, l'apprentissage en profondeur est un apprentissage automatique et fonctionne de manière similaire (d'où la raison pour laquelle les termes sont parfois vaguement échangés). Cependant, ses capacités sont différentes.

Bien que les modèles d'apprentissage automatique de base améliorent progressivement leurs fonctions spécifiques à mesure qu'ils absorbent de nouvelles données, ils nécessitent toujours une intervention humaine. Si un algorithme d'IA renvoie une prédiction inexacte, un ingénieur doit intervenir et faire des ajustements. Avec un modèle d'apprentissage en profondeur, un algorithme peut déterminer si une prédiction est exacte ou non via son propre réseau de neurones - aucune aide humaine n'est requise.

Un modèle d'apprentissage en profondeur est capable d'apprendre grâce à sa propre méthode de calcul, une technique qui donne l'impression qu'il a son propre cerveau.

Pour récapituler, les principales différences entre l'apprentissage automatique et l'apprentissage en profondeur sont :

- L'apprentissage automatique utilise des algorithmes pour analyser les données, apprendre de ces données et prendre des décisions éclairées en fonction de ce qu'il a appris.
- L'apprentissage en profondeur structure les algorithmes en couches pour créer un « réseau de neurones artificiels » capable d'apprendre et de prendre des décisions intelligentes par lui-même.
- L'apprentissage en profondeur est un sous-ensemble de l'apprentissage automatique. Bien que les deux relèvent de la vaste catégorie de l'intelligence artificielle, l'apprentissage en profondeur est ce qui alimente l'IA la plus humaine.**[15]**

13. Conclusion

En conclusion, il est indéniable que l'intelligence artificielle et le deep learning ont révolutionné de nombreux domaines de la technologie et continuent de le faire. Avec des avancées en matière de matériel et de logiciel, les capacités de l'IA et du deep learning continuent de s'étendre, ouvrant ainsi de nouvelles possibilités dans la résolution de problèmes complexes. De l'analyse de données à la reconnaissance d'images et de la parole, en passant par les applications de la robotique et de l'automatisation, l'IA et le deep learning ont le potentiel d'améliorer considérablement notre qualité de vie et de transformer nos économies et nos sociétés. Cependant, il est important de prendre en compte les questions éthiques et de sécurité qui se posent avec l'utilisation de ces technologies. En fin de compte, il est essentiel de trouver un équilibre entre l'innovation technologique et la responsabilité sociale pour s'assurer que l'IA et le deep learning sont utilisés de manière responsable et bénéfique pour l'humanité.

Chapitre 03 : Méthodologie de travail

1. Introduction

Au cours des deux chapitres précédents, nous avons exposé plusieurs algorithmes de classification de texte en apprentissage automatique et en apprentissage profond. Dans ce chapitre, nous aborderons les mécanismes de traitement des données spécifiques ainsi que les différentes méthodes de représentation des textes. Nous avons expérimenté l'utilisation des algorithmes CNN, LSTM et ANN pour la classification, en ajustant de nombreux paramètres et en enregistrant les résultats pour évaluer et déterminer le meilleur algorithme de classification.

2. Notre objet

L'objectif de cette étude est de souligner l'importance de la classification de texte dans la construction de modèles de prédiction. Nous entreprenons une analyse comparative des divers algorithmes de classification automatique et profonde. Pour ce faire, nous avons utilisé un ensemble de données comprenant plus de 6000 messages, sur lesquels nous avons appliqué ces tests.

3. Prétraitement

Le prétraitement des données (ou nettoyage des données) est une étape essentielle dans l'analyse de données. Cela implique la transformation des données brutes en données exploitables, en éliminant les erreurs, les valeurs manquantes, les doublons et en transformant les données en un format standardisé.

Voici les étapes du prétraitement des données qui nous avons faire **:**

- **Importer les données**

  Pour importer des données dans Python, vous pouvez utiliser la bibliothèque pandas qui fournit des outils pour lire différents types de fichiers de données. Pour importer des données à partir d'un **fichier CSV**, vous pouvez utiliser la fonction **read_csv()** de pandas.

- **Sélectionner les colonnes nécessaires**

  La sélection des colonnes dont nous avons besoin est une étape importante du prétraitement des données car cela permet de réduire la taille de l'ensemble de données et de se concentrer uniquement sur les variables pertinentes pour notre travail**.**

  Après avoir sélectionné les colonnes principales, nous les renommons avec des noms associés , on utilise la fonction **rename().**

- **Suppression des documents contenant des valeurs nulles**

  La tâche consistait à supprimer les documents qui contiennent des valeurs nulles dans un **DataFrame** en Python. Pour cela, nous avons utilisé la méthode **dropna()** de pandas, qui permet de supprimer les lignes ou les colonnes contenant des valeurs nulles.

- **Supprimer les caractères spéciaux**

  Pour supprimer les caractères spéciaux d'une chaîne de caractères en Python, vous pouvez utiliser les expressions régulières **(regex**) et la méthode **sub()** du module **re .** La méthode **sub()** permet de remplacer des parties d'une chaîne de caractères par une autre chaîne vide.

- **Conversion d'étiquettes en variables binaires**

  Maintenant que nous avons une compréhension de base de ce à quoi ressemble notre ensemble de données, convertissons nos étiquettes en variables binaires (nous faisons une classification binaire), en utilise la fonction **label.map()**

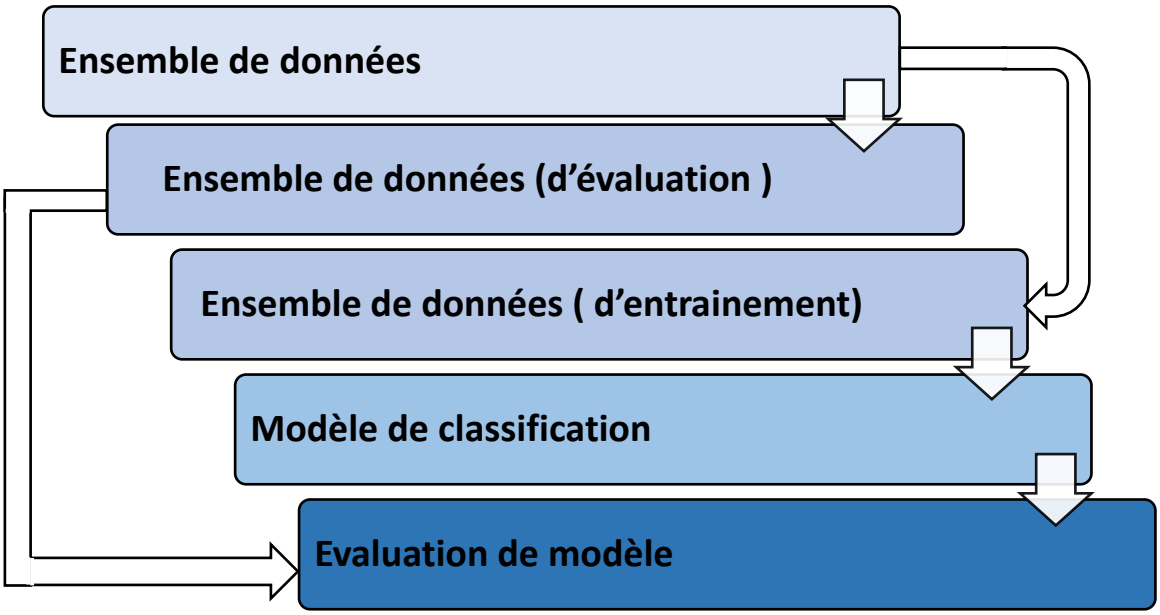
4. La représentation des textes

4.1. La représentation en sacs de mots << back of words >>

Bag of Words **(BoW),** ou sac de mots en français, est une méthode de traitement de texte qui consiste à représenter un document sous forme d'un vecteur de fréquence de mots. Le principe est de transformer chaque document en un vecteur de taille fixe, où chaque dimension représente un mot unique dans le corpus de texte, et où la valeur de chaque dimension est la fréquence du mot correspondant dans le document.

La méthode **BoW** est souvent utilisée pour des tâches de classification de texte telles que la classification de courriels en spam et non-spam, la classification de commentaires en positif et négatif, ou encore la classification de documents en différentes catégories.

5. Processus général de notre système classification



6. Méthodologie de travail

Dans le cadre de notre étude, nous avons choisi d'évaluer les performances de plusieurs algorithmes d'apprentissage profond : **CNN** (Convolutional Neural Network), **LSTM** (Long Short-Term Memory), et **ANN** (Artificiel Neural Network) sur une base de données de **6742 messages**.

La base est connu sous le nom SMS-SPAM . Notre objectif est de déterminer le modèle qui présente les meilleurs résultats de performance en termes de précision, rappel et Accuracy dans la phase de prétraitement nous avons utilisé différentes techniques : pour le codage de texte on a choisi la représentation sac de mots. Egalement notre processus contient une phase de tokenisation , stop words et stemming. Les corpus dont nettoyés selon cette méthode avant d'être utilisées pour entraînement et test. Les résultats obtenus ont été analysés et comparés pour déterminer le modèle le plus performant.

1	**Charger les données**
	✓ Ensembles des données contient 6742 Doc

2	**Prétraitement**
	✓ Sélectionner les colonnes nécessaires ✓ Suppression des documents contenant des valeurs nulles ✓ Supprimer les caractères spéciaux ✓ Conversion d'étiquettes en variables binaires

3	**La représentation des textes**
	✓ La représentation en sacs de mots << back of words >>

4	**Les modèles de classification Deep Learning**
	✓ **CNN** (Convolutional Neural Network) ✓ **LSTM** (Long Short-Term Memory) ✓ **ANN** (Artificiel Neural Network)

4	**Les modèles de classification Machine Lerning**
	✓ **SVM (**Support Vector Machine) ✓ **NB (**Naïve Bayes) ✓ **DT (**Decision Tree)

5	**Résultat et comparaison des algorithmes de classification**

Tableau 3: Les étapes de notre systèmes classificattion

7. Evaluation des modèle classification

Trois mesures de performances ont été choisi dans notre étude : notamment précision, rappel et Accuracy .

7.1 Précision

La précision (P) est le rapport du nombre de documents correctement attribués à la collection de document au nombre total de documents.

$$p = \frac{VP}{VP+FP}$$

7.2 Rappel (recall en anglais)

Le Rappel (R) présente le rapport du nombre de documents correctement attribués à la collection au nombre total de documents.

$$R = \frac{VP}{VP+FN}$$

7.3 L'exactitude (accuracy en anglais)

L'exactitude est calculée en divisant le nombre total d'exemples correctement classés par le nombre total d'exemples dans l'ensemble de données.

$$E = \frac{VP+VN}{VP+FP+VN+FN}$$

Tel que

VP : vrais Positifs

VN : vrais négatifs

FP : faux positifs

FN : faux négatifs

8. Optimisation des modèle classification

Quatre mesures d'optimisation ont été choisi dans notre étude : notamment Adam, Nadam SGD et AdaDelta.

8.1 ADAM (Adaptive Moment Estimation)

Ce sont les étapes principales de l'algorithme Adam pour mettre à jour les poids pendant le processu d'entraînement. L'algorithme Adam combine la prise en compte des moments 1 et 2 ainsi que le taux d'apprentissage pour mettre à jour les poids, ce qui lui permet d'améliorer la stabilité et la vitesse du processus d'entrainement dans de nombreux cas.

Initialisation : Les poids du modèle sont initialisés de manière aléatoire.

Calcul des gradients : Les gradients des poids du modèle par rapport à la fonction de coût sont calculés à l'aide de la technique de la rétropropagation du gradient.

Mise à jour des moments 1 et 2 : Les moments 1 et 2 sont mis à jour en utilisant **les formules suivantes :**

$$m\_t = beta1 * m\_t\text{-}1 + (1 - beta1) * gradient\_t$$

$$v\_t = beta2 * v\_t\text{-}1 + (1 - beta2) * (gradient\_t\text{^}2)$$

(t est le temps actuel et beta1 et beta2 sont des paramètres qui contrôlent l'importance des moments.)

Correction des biais 1 et 2 : Les biais 1 et 2 sont corrigés pour compenser les biais initiaux **:**

$$\text{m\_t\_corr} = \text{m\_t} / (1 - \text{beta1\^{}t})$$
$$\text{v\_t\_corr} = \text{v\_t} / (1 - \text{beta2\^{}t})$$

Mise à jour des poids : Les poids sont mis à jour en utilisant la formule suivante :

$$\text{W\_t} + 1 = \text{W\_t} - (\text{learning\_rate} / (sqrt(v\_t\_corr) + epsilon)) * m\_t\_corr$$

(learning_rate est le taux d'apprentissage et epsilon est une petite valeur pour éviter la division par zéro.)

Répétition : Les étapes 2 à 5 sont répétées jusqu'à ce qu'une condition d'arrêt spécifiée soit remplie.

8.2. Nadam (Nesterov-accelerated Adaptive Moment Estimation)

Ce sont les principales étapes de l'algorithme Nadam pour mettre à jour les poids pendant le processus d'entraînement. L'algorithme Nadam combine les techniques de la rétro propagation du gradient et de la correction des biais 1 et 2, ce qui lui permet d'améliorer la stabilité du processus d'entraînement et l'efficacité de la mise à jour des poids dans de nombreux cas.

Initialisation : Les poids du modèle sont initialisés de manière aléatoire.

Calcul des gradients : Les gradients des poids du modèle par rapport à la fonction de coût sont calculés en utilisant la technique de la rétro propagation du gradient.

Mise à jour des moments 1 et 2 : Les moments 1 et 2 sont mis à jour en utilisant les formules suivantes **:**

$$\text{m\_t} = \text{beta1} * \text{m\_t-1} + (1 - \text{beta1}) * \text{gradient\_t}$$
$$\text{v\_t} = \text{beta2} * \text{v\_t-1} + (1 - \text{beta2}) * (\text{gradient\_t\^{}2})$$

(t est le temps actuel et beta1 et beta2 sont des paramètres qui contrôlent l'importance des moments.)

Correction des biais 1 et 2 : Les biais 1 et 2 sont corrigés pour compenser les biais initiaux **:**

$$\text{m\_t\_corr} = \text{m\_t} / (1 - \text{beta1\^{}t})$$
$$\text{v\_t\_corr} = \text{v\_t} / (1 - \text{beta2\^{}t})$$

Mise à jour des poids : Les poids sont mis à jour en utilisant la formule suivante **:**

$$W\_t + 1 = W\_t - (learning\_rate / (sqrt(v\_t\_corr) + epsilon)) * (beta1 * m\_t\_corr + (1 - beta1) * gradient\_t / (1 - beta1\^t))$$

(learning_rate est le taux d'apprentissage et epsilon est une petite valeur pour éviter la division par zéro.)
Répétition : Les étapes 2 à 5 sont répétées jusqu'à ce qu'une condition d'arrêt spécifiée soit remplie.

8.3 Adadelta

Adadelta est un algorithme d'optimisation adaptatif utilisé pour entraîner des modèles de réseaux neuronaux profonds. Il est conçu pour ajuster automatiquement le taux d'apprentissage pendant l'entraînement sans nécessiter de paramétrage manuel
Voici les étapes de l'algorithme Adadelta en général **:**

- **Initialisation** : Le poids initial du modèle est initialisé aléatoirement.
- **Variables d'initialisation** : Les variables suivantes sont initialisées avec une valeur initiale nulle :
  Accumulated gradient square: accum_g=0
  Accumulated update square: accum_u = 0
- **Calcul du dégradé** : Les gradients sont calculés pour le poids du modèle par rapport à la fonction de coût en utilisant la technique du gradient de rétro propagation.
- **Variables de mise à jour** : Les variables sont mises à jour à l'aide des formules Suivantes **:**

  accum_g = rho * accum_g + (1 - rho) * gradient^2

  Calculer la mise à jour **:**

  $$\text{update} = -\text{sqrt}(\text{accum\_u} + \text{epsilon}) / sqrt\left(accum_g + epsilon\right) * gardient$$

  Carré de mise à jour cumulé **:**

  accum_u = rho * accum_u + (1 - rho) * update^2
- **Mise à jour du poids** : Le poids est mis à jour en appliquant la formule suivante : W_t+1 = W_t + update
- **Répétition** : Les étapes 2 à 5 sont répétées jusqu'à ce que la condition d'arrêt spécifiée soit remplie.

8.4 SGD (Stochastic Gradient Descent)

Ce sont les principales étapes de l'algorithme SGD pour mettre à jour les poids pendant le processus d'entraînement. Les poids sont mis à jour en fonction du gradient calculé et du taux d'apprentissage. Les étapes sont appliquées à chaque échantillon ou lot de données dans le cas de l'entraînement par mini-lots (mini-batch training) ou à chaque donnée dans le cas de l'entraînement par lots complets (batch training). Le processus est répété jusqu'à ce que les conditions spécifiées, telles que le nombre de cycles ou l'atteinte de la précision souhaitée, soient remplies

Initialisation

Les poids du modèle sont initialisés de manière aléatoire**.**

Calcul des gradients

Les gradients des poids du modèle par rapport à la fonction de coût sont calculés en utilisant la technique de la rétro propagation du gradient**.**

Mise à jour des poids

Les poids sont mis à jour en utilisant la formule suivante **:**

$$W_{t+1} = W_t - learning\_rate * gradient_t$$

(où W_t représente les poids actuels, W_{t+1} est le poids mis à jour, et learning_rate est le taux d'apprentissage.)

Répétition :

Les étapes 2 et 3 sont répétées pour chaque donnée d'entraînement ou pour un nombre spécifié de cycles (epochs) jusqu'à ce qu'un critère d'arrêt spécifié soit atteint

9. Pertes des modèle classification

La fonction de coût Binary Crossentropy, ou entropie croisée binaire en français, est une fonction couramment utilisée dans les problèmes de classification binaire. Elle mesure la différence entre les prédictions du modèle et les étiquettes réelles associées aux données

0……. La formule mathématique de la fonction de coût Binary Crossentropy est la suivante **:**

$$BCE(y, t) = -[t * \log(y) + (1 - t) * \log(1 - y)]$$

BCE : représente la Binary Crossentropy.

y : est la sortie (prédiction) produite par le modèle. t est la valeur réelle cible (étiquette), où t est soit 0 soit 1.

log représente le logarithme naturel.

10. Conclusion

Dans ce chapitre nous avons éclaircir la stratégie générale de notre travail. Nous avons présenté l'objectif visé par ce travail et expliqué la méthodologie suivie pour l'étude comparative que nous avons mené entre trois algorithmes de classification des textes combinés avec des méthodes de représentation du document et codage de termes. La mise en œuvre de cette étude comparative sur un vrai corpus de textes résultats obtenus ainsi que leur analyse et discussion feront l'objet du prochain chapitre.

Chapitre 04 : Implémentation et résultats

1. Introduction

Ce chapitre est essentiellement consacré aux expérimentations réalisées afin d'atteindre l'objectif de notre thème d'étude. Dans ce but, nous décrivons dans cette partie les outils exploités pour réaliser les objectifs tracés, à savoir le choix du langage de programmation, l'environnement de développement et le matériel utilisé. Le processus de classification en utilisant différentes architectures de l'apprentissage profond est présenté tout en donnant les résultats des expérimentations et leurs explications. Nous enchainons ensuite avec une discussion des résultats obtenus. A la fin de ce chapitre, nous terminons par une conclusion.

2. Outils matériels et logiciels

2.1. Configuration matérielle

Ce travail a été implémenté sur un PC, caractérisé comme suit :

- **Un Processeur :** Intel core i3-6006u cpu @ 2.00ghz 1.99ghz
- **Une RAM :** 8 GB
- **Une système d'exploitation :** Windows 10 Pro 64 bits

2.2. Environnement logiciel

Pour implémenter notre application, on a choisi comme outils de développement, Python.

Python est le langage de programmation open source le plus largement utilisé par les informaticiens. Il a pris la tête de la gestion de l'infrastructure, de l'analyse de données et du développement de logiciels. Python offre de nombreuses qualités qui permettent aux développeurs de se concentrer sur leur travail plutôt que sur les détails techniques. Il a libéré les développeurs des contraintes de syntaxe imposées par les langages plus anciens, ce qui rend le développement de code plus rapide et plus efficace avec Python par rapport à d'autres langages.

Anaconda est une distribution de logiciels open source très populaire en langage Python. Il fournit un environnement complet pour le développement et l'exécution de programmes Python, en incluant un gestionnaire de paquets, un environnement de développement intégré **(IDE)** et de nombreux packages scientifiques préinstallés. Anaconda est couramment utilisé dans les domaines
de la science des données, de l'apprentissage automatique et de l'analyse statistique, offrant aux utilisateurs une plateforme puissante et conviviale pour leurs projets de programmation.

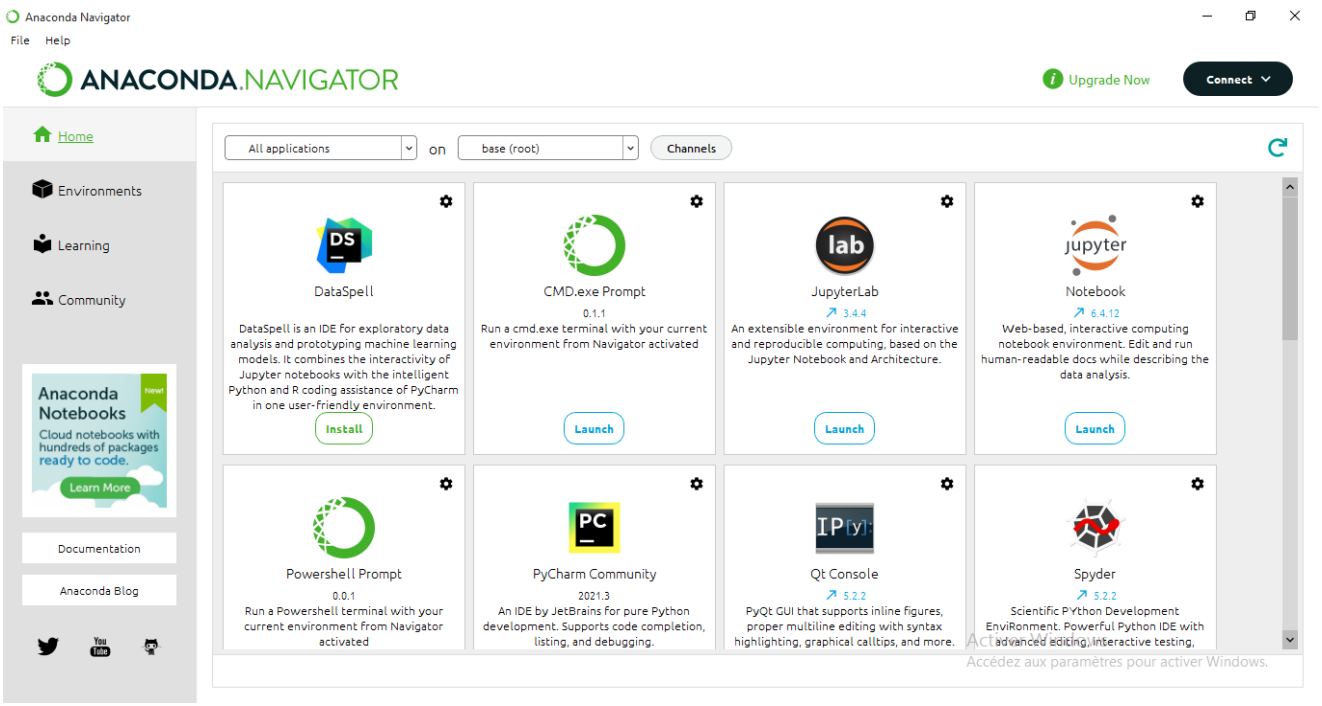


Figure 23: Interface d'Anaconda Navigator

Jupyter notebook

Est une plateforme open source permettant d'exécuter et de partager du code informatique. Il est principalement utilisé pour le développement et l'exécution de code Python, bien qu'il prenne également en charge d'autres langages de programmation. Jupyter permet aux utilisateurs de créer des notebooks interactifs, dans lesquels ils peuvent combiner du code, des visualisations, du texte explicatif et des résultats en temps réel. Ces notebooks peuvent être partagés et collaborés avec d'autres utilisateurs, ce qui en fait un outil très utile pour l'exploration de données, l'apprentissage automatique, l'analyse statistique et bien d'autres domaines**.**

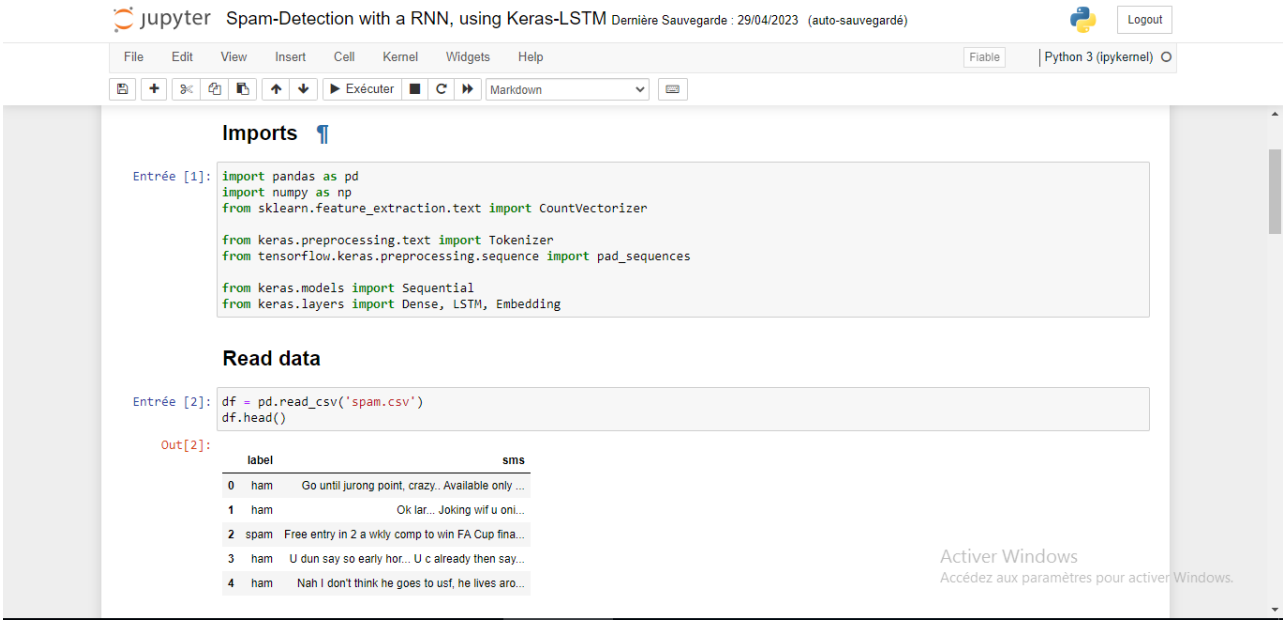


Figure 24 : interface de Jupyter Notebook

3. Bibliothèques utilisées

Dans cet environnement on a installé des packages qui nous ont facilité la programmation qui sont **:**

3.1. NumPy

Est une bibliothèque pour langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux

3.2. Bandas

Est une librairie python qui permet de manipuler facilement des données à analyser: manipuler des tableaux de données avec des étiquettes de variables (colonnes) et d'individus (lignes).

3.3. Keras

Est une bibliothèque open source en langage Python dédiée à l'apprentissage profond (Deep Learning). Elle permet de créer, entraîner et évaluer facilement des modèles de réseaux de neurones artificiels. Keras offre une interface conviviale et abstraite, facilitant le développement de modèles d'apprentissage profond.

3.4. TensorFlow

Est une bibliothèque open source très populaire pour l'apprentissage automatique (machine learning) et l'intelligence artificielle. Développée par Google, TensorFlow permet de créer, entraîner et déployer des modèles d'apprentissage automatique, en mettant l'accent sur les réseaux de neurones et le calcul numérique à grande échelle.

4. Base de données utilisée

Contient un ensemble de spam SMS est un ensemble de messages SMS étiquetés qui ont été collectés à des fins de recherche sur le spam SMS. Ces SMS ont été recueillies pour la recherche au Département d'informatique de l'Université nationale de Singapour**.** Elle contient un ensemble de messages SMS en anglais de plus de 6000 messages, étiquetés comme étant légitimes (ham) ou du spam. Les fichiers contiennent un message par ligne. Chaque ligne est composée de deux colonnes : la colonne 1 contient l'étiquette (ham ou spam) et la colonne 2 contient le texte brut. La collection comprend 13,41% de messages SMS de spam et 86,95% de messages SMS légitimes (ham) .

Vous pouvez télécharger l'ensemble de données à partir d'ici

Scanner pour télécharger l'ensemble de données

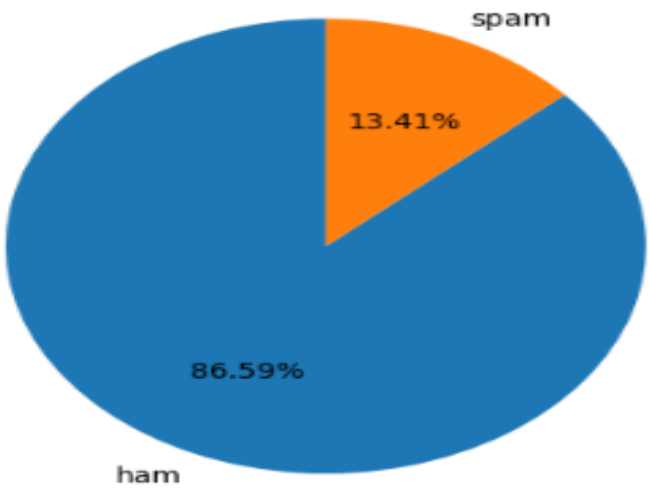


Figure 25 : Nombre Totale de Ham et spam dans l'ensemble de donnée

5. Expérimentation

Nous avons réalisé une série d'expérimentations sur l'ensemble de données de spam. Plus précisément, nous avons effectué des tests et comparé les performances des modèles suivants : CNN, LSTM et ANN. Nous avons alloué 70% des données à l'entraînement, tandis que les 30% restants ont été réservés à l'évaluation des résultats. Nous avons consacré 10 epoch à l'apprentissage.

« Epoch » : représente une exécution complète du modèle sur l'ensemble de données d'apprentissage à partir duquel apprendre.

6. Les résultats des algorithmes

Dans cette partie, nous allons afficher les résultats obtenus dans chaque algorithme, avec un résultat final comme les meilleurs paramètres pour cet algorithme

6.1. Agorithme de artificiel neural network

Les paramètres		Les résultats	
Optimiz	Metrice	Loss value	Succes value
ADAM	ACCURACY	0.0686	0.9838
SGD		0.1687	0.9474
ADADELTA		0.1673	0.9474
NADAM		0.0693	0.9844
ADAM	RAPPEL	0.1246	0.8716
SGD		0.1247	0.8716
ADADELTA		0.1247	0.8716
NADAM		0.1937	0.8670
ADAM	Précision	0.2400	0.9895
SGD		0.2400	0.9895
ADADELTA		0.2400	0.9895
NADAM		0.2690	0.9894

Tableau 4: Résultat d'algorithme artificiel neural network

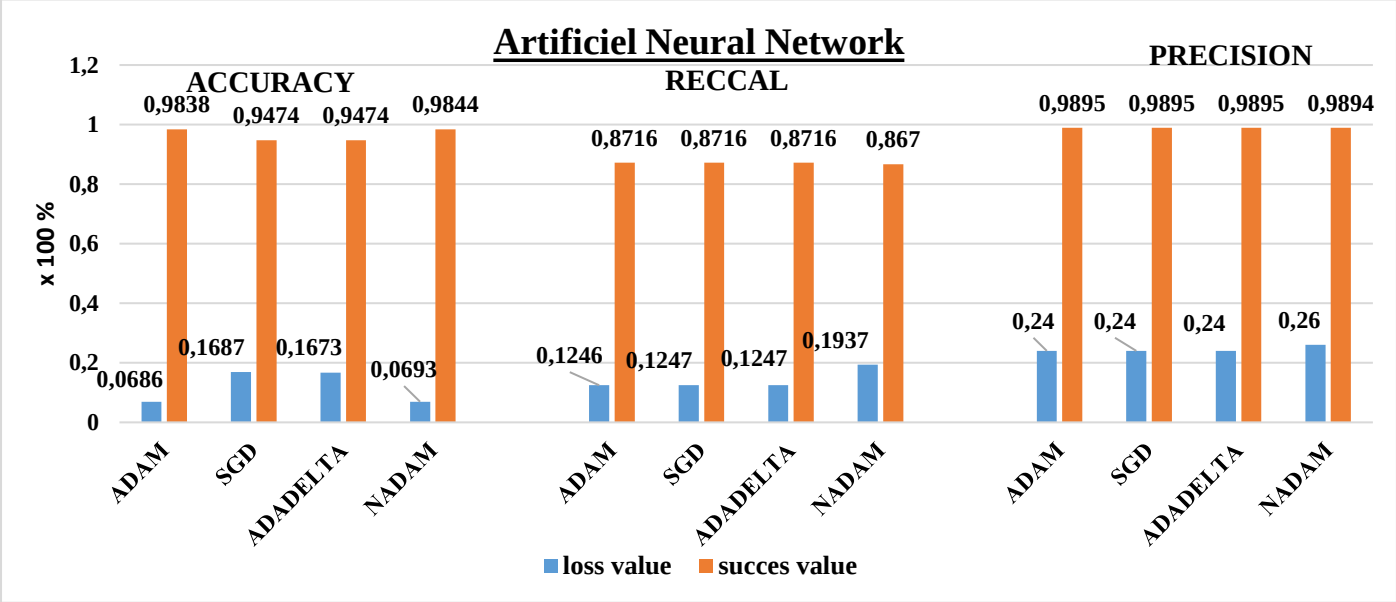


Figure 26 : Représentation graphique des résultats obtenus

D'après les résultats obtenus dans le tableau précédent, il est évident que l'optimiseur Adam et NADAM, utilisé avec la Metrice (accuracy), ont obtenu les meilleurs résultats**.**

6.2. Algorithme de convolutional neural network

Les paramètres		Les résultats	
Optimiz	Metrice	Loss value	Succes value
ADAM	ACCURACY	0.026	0.9996
SGD		0.405	0.864
ADADELTA		0.669	0.864
NADAM		0.023	0.996
ADAM	Recall	0.028	0.982
SGD		0.399	0
ADADELTA		0.665	0
NADAM		0.026	0.978
ADAM	Précision	0.023	0.978
SGD		0.404	0
ADADELTA		0.661	0
NADAM		0.021	0.970

Tableau 5: Résultat d'algorithme CNN

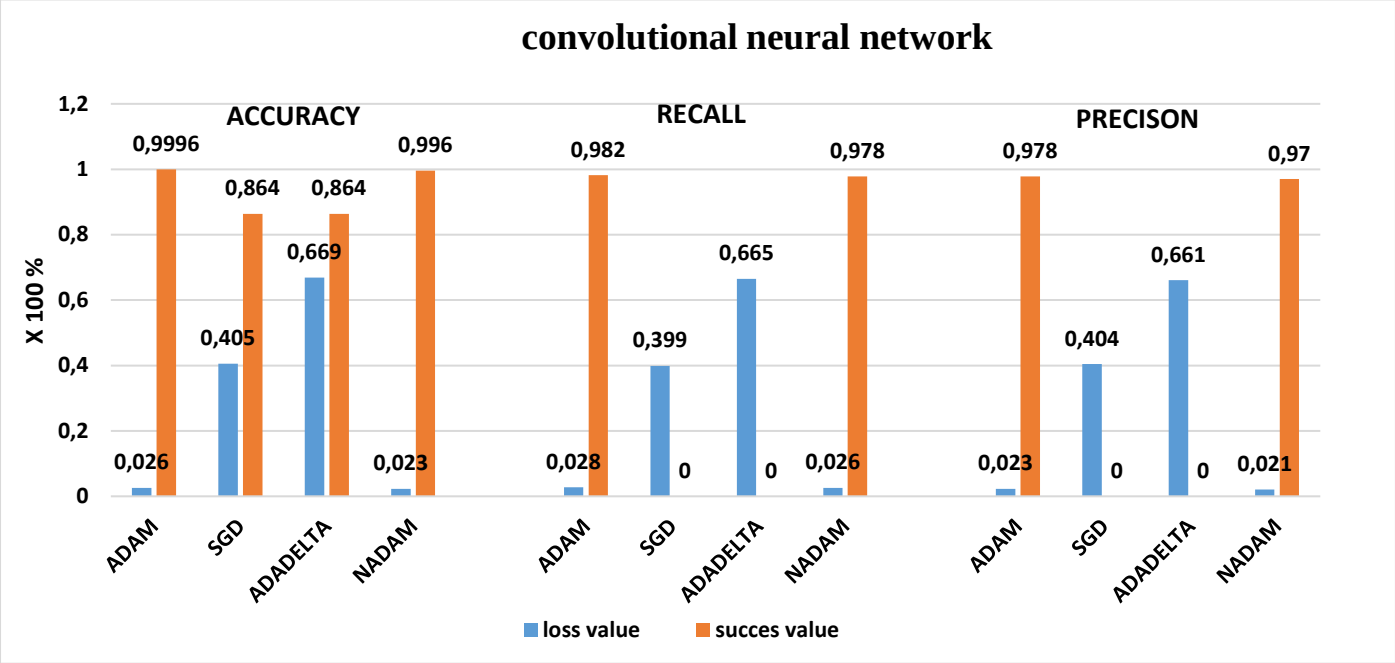


Figure 27 : Représentation graphique des résultats obtenus

D'après les résultats obtenus dans le tableau précédent, il est évident que l'optimiseur Adam, utilisé avec la Metric (accuracy), ont obtenu les meilleurs résultats**.**

6.3. Algorithme de long short term memory

Les paramètres		Les résultats	
Optimiz	Metrice	Loss value	Succes value
ADAM	ACCURACY	0.1758	0.9779
SGD		0.3299	0.8606
ADADELTA		0.6616	0.8606
NADAM		0.1740	0.9761
ADAM	Recall	0.1338	0.9099
SGD		0.3330	0
ADADELTA		0.6684	0
NADAM		0.1835	0.887
ADAM	Précision	0.1858	0.9535
SGD		0.3358	0
ADADELTA		0.6589	0
NADAM		0.2000	0.9550

Tableau 6: Résultat d'algorithme LSTM

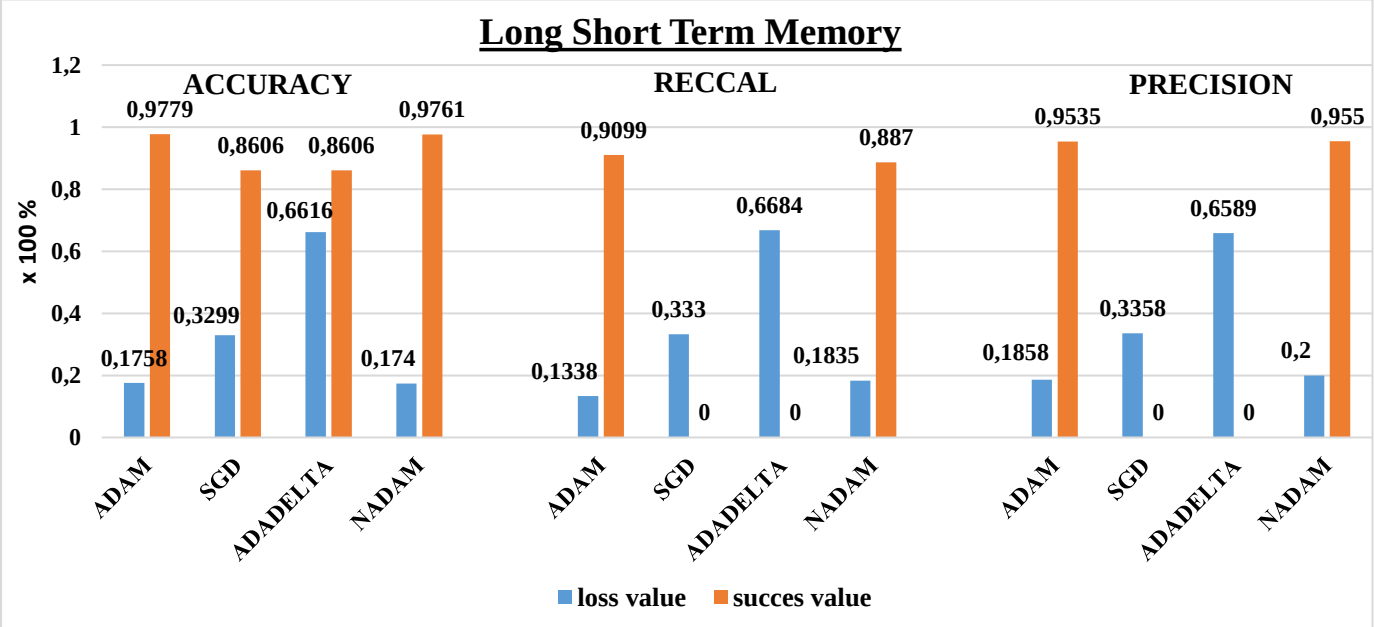


Figure 28 : Représentation graphique des résultats obtenus

D'après les résultats obtenus dans le tableau précédent, il est évident que l'optimiseur Adam et NADAM, utilisé avec la Metric (accuracy), ont obtenu les meilleurs résultats**.**

7. Comparaison de résultat d'algorithme deep learning et machine learning

Nous comparerons les meilleurs résultats des algorithmes d'apprentissage en profondeur obtenus ci-dessus avec les algorithmes d'apprentissage automatique et proposerons le meilleur algorithme en fonction de la VALEUR du succès de la classification.

ALGORITHME	**SUCCESS VALUE**
CNN (convolutional neural network)	**0.9996**
LSTM (Long Short Term Memory)	**0.9779**
ANN (Artificiel Neural Network)	**0.9838**
SVM (Support Vector machine)	**0.9826**
DT (Decision Tree)	**0.9677**
NB (Naive Bayes)	**0.9850**

Tableau 7: Résultat des algorithmes

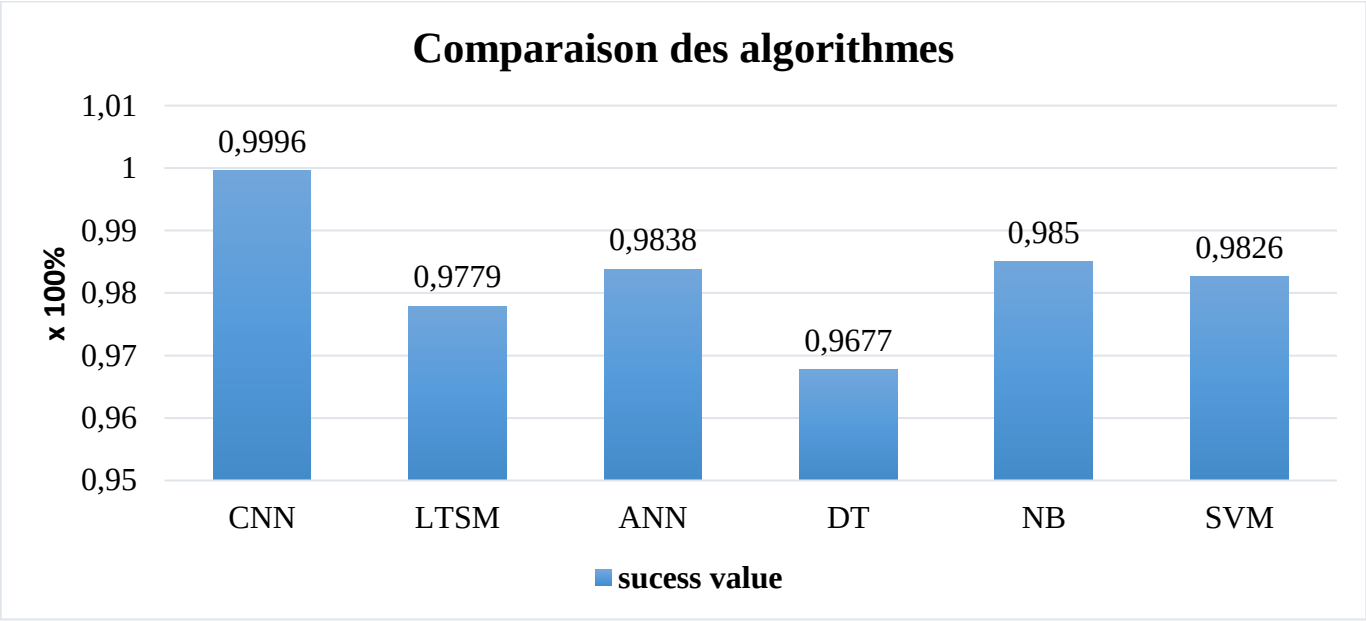


Figure 29: Résultat de comparaison d'algorithme DL ML

D'après ces résultats, nous pouvons observer que l'algorithme CNN a obtenu le meilleur score de succès avec une valeur de 0.9996. Cependant, il convient de noter que les algorithmes ANN, SVM et NB ont également obtenu des scores élevés, dépassant 0.98. Par conséquent, pour choisir le meilleur algorithme en fonction de la précision de la classification, il est recommandé d'envisager l'algorithme CNN comme première option, suivi de près par les algorithmes ANN, SVM et NB.

8. Conclusion

Dans ce chapitre nous a permis de réaliser la partie expérimentale de notre projet. Cela nous a donné l'opportunité d'expérimenter différentes configurations d'algorithmes d'apprentissage profond afin de déterminer lequel est le meilleur pour la classification de texte. Ce chapitre nous a également permis de comparer les algorithmes d'apprentissage automatique, et nous en sommes arrivés à la conclusion que l'algorithme CNN est le meilleur pour la classification de texte, avec un taux de Succès de **99%** .

Conclusion générale

Au cours des dernières décennies, on vit une explosion gigantesque des données textuelle sur l'internet et en particulier les réseaux sociaux ont vu un grand usage d'utilisation du courrier électronique. Entraînant l'envoi de milliers, voire de millions de messages. Cependant, cette croissance a également causé des problèmes tels que l'apparition des messages malveillants, indésirables (spam).

Le recours aux méthodes de l'intelligence artificiel données se propose comme une alternative indispensable pour résoudre ce genre de problème. Parmi ces méthodes, on trouve les techniques du Deep Learning qui ont fait ces preuves dans plusieurs domaines d'application notamment ….classification des maladies des images des audio …. etc

Notre mission dans ce mémoire est de résoudre un problème qui s'articule sur la classification des messages textuels en utilisant les techniques d'apprentissage en profondeur. On a appliqué différentes architecture CNN LSTM et ANN.

Ainsi d'autres classifieurs d'apprentissage automatique SVM DT et NB. On a choisi comme base de test les données SMS-Spam.

Pour réaliser notre objectif on a suivi les étapes suivantes. La première étape est consacrée à la représentation textuelle : sac de mots. Nous avons divisé l'ensemble de données textuelles en deux parties, dont 70 % sont destinés pour l'entraînement. Et 30% pour la vérification. Dans un second temps. Dans la partie expérimentation nous avons appliqué les algorithmes d'apprentissage profond, le réseau de neurones Convolutional (CNN), le réseau de mémoire à long terme (LSTM), et le réseau de neurones artificiel (ANN) avec le choix d'algorithmes d'amélioration des performances : Adam et NADAM, ainsi que ADADELTA et SGD. L'évaluation des performances on a utilisé les mesure de la valeur de succès Précision, Recall et Accuracy et Nous avons choisi la mesure Binary Crossentropy pour mesurer la valeur de perte. Pour mettre en œuvre notre projet. Nous avons choisi l'environnement Jupyter et le langage Python en se basant sur un ensemble de bibliothèques informatiques. Les tests expérimentaux sont appliqués sur la base de test spam sms dans laquelle il existe plus que 6000 sms répartis en plus que 4000 d'entrainement plus que 2000 de test. Ce travail nous a permis de comparer les résultats obtenus dans la classification des algorithmes d'apprentissage profond et de les comparer avec l'algorithme d'apprentissage automatique SVM... Les résultats de notre expérience montrent que l'algorithme de réseau Convolutional CNN est le meilleur en termes de précision, recall et accuracy dans la base textuel.

Finalement, nous proposons à l'avenir d'utiliser une base de données de grande taille ainsi que d'expérimenter une base de données différente contenant des messages en français ou en arabe, ainsi que de tester ces algorithmes sur des appareils performants afin d'obtenir des résultats rapidement.

Bibliographe :

[1] S.DELLALI, « Exploitation des Ontologies pour la Classification des Textes Arabes», Mémoire de Master, Université Mohamed BOUDIAF– Msila, Algérie, **2016-2017.**

[2] FABRIZIO SEBASTIANI, « Machine Learning in Automated Text Categorization », Conseil recherche National, Italie, **Mars 2002**.

[3] MATALLAH Hocine «Classification Automatique de Textes Approche Orientée Agent», UNIVERSITE ABOUBEKR BELKAID-TLEMCEN, Février **2011.**

[4] OUALI Choayb, « Classification automatique de textes»,universite de m'sila ,2014.

[5] J.Clech, D.A.Zighed. « Une technique de réétiquetage dans un contexte de catégorisation detextes ».

[6] Jalam, R. and Chauchat, J.-H. « Pourquoi les n-grammes permettent de classer des textes? Recherche de mots-clefs pertinents à l'aide des n-grammes caractéristiques ».

[7] Caropreso Maria Fernanda, Stan Matwin, Fabrizio Sebastiani, « Statistical Phrases in Automated Text Categorization », **2000.**

[8] Furnkranz Johannes, Tom Mitchell, Ellen Rilo « A Case Study in Using Linguistic Phrases for Text Categorization on the WWW » School of Computer Science Carnegie Mellon University URL :www.cs.utah.edu/~riloff/pdfs/final-webslog-paper.pdf, **1998**.

[9] Radwan JALAM, «Apprentissage automatique et catégorisation de textes multilingues», thèse de doctorat, université Lumière Lyon2, Année **2003.**

[10] R. Lefébure, G.Venturi, « Le Data Mining » Edition EYROLLES, deuxième tirage **1998**.

[11]. Raheel, « L'Apprentissage Artificiel pour la Fouille de Données Multilingues : Application à la Classification Automatique des Documents Arabes », Thèse de doctorat en Sciences de l'Information et de la Communication, Université Lumière Lyon 2, **2010**.

[12] Simon RÉHEL, « Catégorisation automatique de textes et Cooccurrence de mots provenant de documents non étiquetés », Mémoire, Université Laval Québec, Canada, Janvier **2005.**

[13] T.DERDRA Amel, F.BENSFIA, « La Représentation Conceptuelle pour la Catégorisation des Textes Multilingue », Mémoire de Master, Université Abou Bakr Belkaid– Tlemcen, **2011-2012.**

[14] Boughaba Mohammed et Boukhris Brahim « L`apprentissage profond (Deep Learning) pour la classification et la recherche d'images par le contenu », Mémoire Master Professionnel,UNIVERSITE KASDI MERBAH OUARGLA

[15] Gurney, k., « an introduction to neural networks », **1997.**

[16] Reza Bosagh Zadeh, Bharath Ramsundar «TensorFlow for Deep Learning», **2017.**

[17] Https **://**colah.github.io/posts/2015-08-Understanding-LSTMs/

[18] By Patrick Grieve, «Contributing Writer Last updated »,**March 8, 2022**

[19] Salton and McGill, 1983

[20] Laure Audubon, « Le Machine learning: définition, histoire et cas d'usage» ,juin 29, 2023

[21] mobiskill « équels sont les algorithmes de deep learning ? », 26 mai 2021

[22] devoteam.com « Aller plus loin en deep learning avec les réseaux de neurones récurrents (RNNs) »

[23] Hind fihakhir, «Classification du texte avec deep learning », Université de Ghardaia, , 2017/2018