MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LARECHERCHESCIENTIFIQUE

UNIVERSITE SAAD DAHLEB BLIDA

FACULTE DES SCIENCES DEPARTEMENT INFORMATIQUE

Mémoire Master

Spécialité Ingénierie des Logiciels

THEME

La classification des textes arabes en utilisant l'apprentissage profond

Présenté par

- ✓ **Moudjbeur Nour el Houda**
- ✓ **Maamri Asmaa**

Promoteur

M.Ferfera Soufiane

Devant le jury

- ✓ **Présidente : Mme H.Abed**
- ✓ **Examinatrice : Mme S.Daoud**

Promotion 2020-2021

Remerciements

Nos remerciements sincères vont en premier à ALLAH le tout Puissant qui nous a donné la force et le courage pour effectuer et mener à bien ce modeste travail de recherche.

Nos vifs remerciements vont directement à notre promoteur de recherche Monsieur **« Ferfera Soufiane ».**

Qui nous a fait part de son savoir, ainsi que l'écoute, son aide précieuse et pour ces conseils avisées tout au long de la réalisation de ce mémoire.

Nos remerciements vont aux membres de jury qui ont accepté de lire ce travail et de l'évaluer.

Nous remercions aussi tous les enseignants et les enseignantes du Département d'informatique

Nos remerciements s'adressent également aux membres de nos familles surtout nos parents et toutes les personnes qui nous ont soutenu de près et de loin à la réalisation et la finalisation de ce travail.

Moudjbeur Nour el Houda.

Maamri asmaa.

الملخص

شهد التصنيف التلقائي للنصوص مؤخرًااهتمام كبيرا. ويرجع ذلك أساسًا إلى التزايد المستمر للوثائق الرقمية والحاجة إلى تنظيمها بسرعة.

في هذا العمل نتناول مشكلة تصنيف النصوص العربية. لقد قمنا بتطوير ثلاثة نماذج على أساس التعلم العميق. الأول هو نموذج LSTM الذي اعتبرناه هو النموذج الأساسي ، الثاني هو نموذج BILSTM والأخير هو نموذج GRU تستخدم النماذج الثلاثةالسابق ذكرها عدة طرق لتمثيل النص مثل التمثيل في ngrams أو التمثيل في word2vec.

أظهرت الاختبارات أن نموذج BILSTM أعطى نتائج جيدة من حيث دقة التصنيف.

الكلمات المفتاحية: تصنيف النص العربي ، التعلم العميق ، مجموعة البيانات ، طريقة تمثيل نص ، نموذج التعلم العميق ، الشبكة العصبية المتكررة.

Résumé

La classification automatique de textes connaît ces derniers temps un fort regain d'intérêt. Cela est dû essentiellement à la forte croissance des documents numériques disponibles et à la nécessité de les organiser de façon rapide.

Dans ce travail nous abordons le problème de la classification des textes arabes. Nous avons développé trois modèles basés sur l'apprentissage profond. Le premier est un modèle LSTM que nous avons considéré comme modèle de base, le deuxième est un modèle BILSTM et le dernier est un modèle GRU. Les trois modèles utilisent plusieurs méthodes de représentations de textes comme la représentation en ngrams ou la représentation en word2vec.

Les tests ont montrés que le modèle BILSTM a donné de bons résultats en termes de précision de la classification.

Mots clés : Classification textes arabes, Deep learning , Dataset, méthode de représentation, modèle de l'apprentissage profond, réseau de neurone récurent.

Abstract

The automatic classification of texts has recently seen a strong resurgence of interest. This is mainly due to the strong growth of digital documents available and the need to organize them quickly.

In this work we approach the problem of the classification of Arabic texts. We have developed three models based on deep learning. The first is an LSTM model which we have considered as the base model, the second is a BILSTM model and the last is a GRU model. The three models use several methods of text representations such as the representation in ngrams or the representation in word2vec.The tests showed that the BILSTM model gave good results in terms of classification precision.

Keywords: Arabic text classification, Deep learning, Dataset, representation method, deep learning model, recurrent neural network

Table Des Matière

Listes des figures

Liste des tables

Introduction générale

La classification de texte est le processus de regroupement de documents textuels en classes ou catégories en fonction de leur contenu. Ce processus devient de plus en plus important en raison de la révolution de l'internet qui a fait exploser les informations textuelles, qui sont un patrimoine vivant des entreprises, des administrations et des particuliers, il est devenu indispensable aux utilisateurs du web de trouver les documents pertinents, pour cette raison il devient de plus en plus important de disposer de solutions efficaces pour conserver, chercher et classer ces informations, afin d'assister les utilisateurs à trouver leurs besoins et faciliter leur travail dans certaines tâches qui sont devenues impossible à traiter manuellement. La classification automatique de texte a connue donc un fort regain d'intérêt ces derniers temps.

Comme pour plusieurs langues, le nombre de documents textuels en langue arabe augmente considérablement chaque jour à mesure que de nouvelles pages Web ou d'articles de presse sont ajoutés. La langue arabe est la cinquième langue utilisée dans le monde. Elle est parlée par plus de 422 millions de personnes en tant que première langue et de 250 millions en tant que deuxième langue. Par conséquent, le classement des documents écrit en langue arabe dans des classes spécifiques revêt une grande importance afin de pouvoir exploiter leur contenu.

Les documents textuels sont le plus souvent représentés par les mots qui les composent. Cette représentation est peut-être la plus simple mais pas nécessairement la plus efficace. En effet, les méthodes de classification de documents se basent généralement sur les mots contenus dans les textes. Les documents qui contiennent le plus de mots en communs sont regroupés dans la même classe. Le mot ainsi prit de manière isolé peut ne pas refléter le sens qu'il porte que s'il est associé aux mots voisins. La notion de contexte peut capturer le sens du mot et ainsi mieux représenté un texte et donc améliorer la classification. En effet, pour classer correctement des documents textuels il faut avoir certes un bon modèle de classification mais aussi une bonne représentation des documents textuels. Des documents mal représentés peuvent diminuer dans la précision de la classification.

Le principal problème de la classification de textes est de savoir comment améliorer la précision de la classification. De nombreux algorithmes ont été proposés et mis en œuvre pour résoudre ce problème comme les algorithmes d'apprentissage profond. Cependant peu d'études ont été menées dans ce domaine pour catégoriser et classer les textes en langue arabe.

Notre objectif est de répondre à ces deux questions :
Quelle est la meilleure méthode de représentation des données textuelles ?
Quelle est la meilleure technique d'apprentissage pour la classification des textes arabes ?

Nous proposons donc d'explorer différentes techniques d'apprentissage profond en particulier les réseaux de neurones récurrents pour la classification des textes arabes associés à différentes méthodes de représentations de données textuelles. L'objectif est d'arriver à améliorer la précision de la classification.

Afin d'atteindre cet objectif, nous avons décomposé notre mémoire en trois chapitres.
Le premier chapitre vise à présenter l'état de l'art lié à notre travail. Il est divisé en trois parties. La première concerne la classification des textes, la deuxième consiste à définir les concepts fondamentaux de l'apprentissage profond et la dernière présente les différents travaux connexes.

Le deuxième chapitre porte sur la conception de notre système. Nous avons proposées trois modèles de l'apprentissage profond avec plusieurs méthodes de représentation de texte.

Le dernier chapitre contient les résultats obtenus et une évaluation selon des métriques définis pour pouvoir comparer entre les différents modèles de l'apprentissage profond et par rapport aux approches déjà développées dans des travaux précédents rentrants dans le même contexte .

Nous concluons en rappelant les principaux résultats obtenus et les perspectives à envisagées.

CHAPITRE 01 : ETAT DE L'ART

I. Introduction

Depuis de nombreuses années, l'Intelligence Artificielle est appliquée dans plusieurs domaines (Traitement d'image, Robotique, Raisonnement automatique et Acquisition de connaissances, Traitement du langage Naturel, etc.). Son utilité a été largement observée par la communauté scientifique, surtout avec l'émergence de l'apprentissage profond.

Actuellement, l'apprentissage profond a montré une grande capacité et efficacité à améliorer les approches du traitement automatique de la langue ,telle que la classification qui consiste à attribuer un ensemble de catégories prédéfinies à un texte donné.

Il existe plusieurs travaux sur la classification des textes en utilisant l'apprentissage profond mais peu de travaux sur la langue arabe.

Nous présentons dans ce chapitre trois parties. La première concerne la classification des textes, la deuxième consiste à définir les concepts fondamentaux de l'apprentissage profond et la dernière présente les différents travaux connexes de classification des textes arabes en utilisant l'apprentissage profond. Chaque partie est décrite en détails dans les sections suivantes.

II. Classification des textes

La Classification de textes [C.T] est aujourd'hui un domaine de recherche bien établi et très actif. Dans cette partie, nous présentons d'abord une définition de la [C.T], ainsi que le processus de C.T qui est constitué de plusieurs étapes : la représentation des textes, la pondération des termes, la réduction de la taille du vocabulaire, choix de classificateur, évaluation du processus de classification .Enfin nous citons les principales difficultés liées à la C.T.

II.1 Définition :

Plusieurs définitions de la classification des textes [C.T] ont vus le jour depuis son apparition, nous citons dans ce contexte les deux définitions suivantes

- **Définition1** : La C.T est une relation bijective qui consiste à "chercher une liaison fonctionnelle entre un ensemble de textes et un ensemble de catégories [étiquettes, classes]". (1)
- **Définition2 :** La CT est le processus qui consiste à associer une valeur booléenne à chaque paire [dj, ci] ∈ D × C, où D est l'ensemble des textes et C'est l'ensemble des catégories. La valeur V [Vrai] est alors associée au couple [dj, ci] si le texte dj appartient à la classe ci tandis que la valeur F [Faux] est associée dans le cas contraire. (2)

II.2 Processus de classification de textes

Comme mentionné dans le paragraphe précédent, le but de la classification automatique des textes est d'apprendre à une machine à classer un texte dans la bonne catégorie en se basant sur son contenu. Pour identifier la catégorie d'un texte, ou la classe à laquelle un texte est associé, un ensemble d'étapes sont habituellement suivi comme il est présenté dans le diagramme ci-dessous

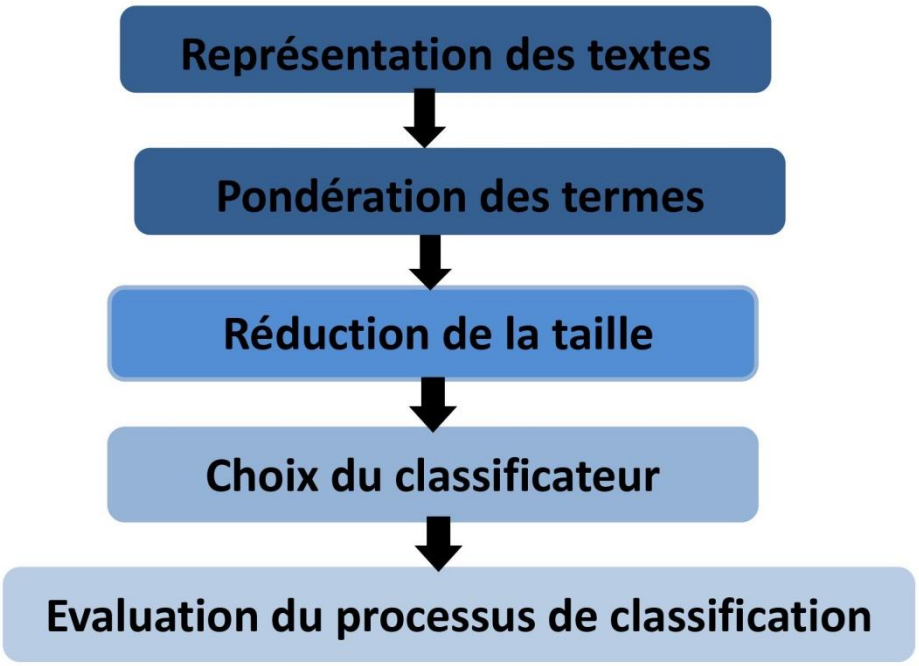

Figure 1 : processus de classification des textes

II.2.1 La représentation des textes

La représentation des textes est une étape très importante dans le processus de classification des textes pour cela il est nécessaire d'utiliser une technique de représentation efficace permettant de représenter les textes sous une forme exploitable par la machine. Le modèle le plus couramment utilisé est le modèle vectoriel dans lequel chaque texte est représenté par un vecteur de *n* termes pondérés. Les différentes représentations des textes sont :

II.2.1.1 Représentation en sac de mots [bag of words]

Les textes sont transformés simplement en vecteurs dont chaque composante représente un mot. (11) Pour clarifier la notion de mot, Y. Gilly dans son ouvrage « Texte et fréquence »(3) l'a considéré comme étant une séquence de caractères appartenant à un dictionnaire, ou formellement, comme étant une suite de caractères séparés par des espaces ou des caractères de ponctuations (Cette définition n'est pas valable pour toutes les langues). La méthode sac de mot est très simple à comprendre et à mettre en œuvre, mais elle souffre de néanmoins certaines lacunes :

- Le problème des mots composés comme : Arc-en-ciel, peut-être et le problème des sigles comme : APN, FAF, IBM.
- La présence des mots outils, qui constituent une grande part des mots d'un texte, mais qui portent peu d'informations utiles pour classer un texte.
- La distinction des mots d'une même famille en raison de leur variation morphologique [ex : écrire, écriture, écrit, écrivain,…] est en général un handicap, car chaque variation a une fréquence très faible alors que les regrouper permettrait d'avoir des fréquences importantes et d'amoindrir le phénomène d'imprécision des fréquences évoqué dans (7)
- Le chinois et le japonais ne séparent pas les mots par des espaces, ce qui peut mener à plusieurs segmentations.
- Les mots composés allemands peuvent être très complexes, par exemple : Le**vensversicherungsgesellschaftsangestellter**[employé d'une société d'assurance vie].

Enfin cette représentation est un regroupement de tous les mots du document « sac de mots » sans prendre en compte les combinaisons et l'ordre des mots dans la phrase entraine une perte dans la sémantique du texte.

Pour y remédier, un prétraitement linguistique amené par l'application des procédures de lemmatisation et de stemming, avant la représentation des documents, est indispensable. (24)

II.2.1.2 Représentation avec les lemmes

La lemmatisation consiste à utiliser l'analyse grammaticale afin de remplacer les verbes par leur forme infinitive et les noms par leur forme au singulier. En effet, Un mot donné peut avoir différentes formes dans un texte, mais leur sens reste le même. Par exemple, les mots jouons, joueurs, jouet seront remplacés par leurs lemmes : « jouer », « joueur » et « jouet » selon le contexte. Cette représentation est simple mais elle peut causer une perte d'informations donnée par le contexte.[5]

II.2.1.3 Représentation avec les racines lexicales (stemming)

Cette méthode consiste à remplacer les mots du document par leurs racines lexicales, et à regrouper les mots de la même racine dans une seule composante. Ainsi, plusieurs mots du document seront remplacés par la même racine, tel que les mots jour, journalier, journée ont la même racine « jour » mais se rendent à trois notions différentes, cette représentation dépend aussi de la langue utilisée. (4)

La représentation des textes par ces stems peut apporter des résultats supérieurs à ceux obtenus par les lemmes , démontré et approuvé par [Loupy](8),et bien meilleur que le codage de type « sac de mots » ou chaque variation d'un mot est considéré comme une nouvelle composante du vecteur. Alors, on peut facilement imaginer combien on va gagner en question de dimensionnalité en optant pour les stems comme descripteurs. (8)

II.2.1.4 Représentation avec les n-gramme

Cette méthode consiste à représenter le document par des n-grammes. Le n- gramme est une séquence de n caractères consécutifs, bi-grammes pour n=2, trigrammes pour n=3, quadri-grammes pour n=4, etc… On n'a plus besoin de chercher les délimiteurs [les espaces ou les caractères de ponctuations] comme c'était le cas pour les mots.

Elle consiste a découpé le texte en plusieurs séquence de n caractère en se déplaçant avec une fenêtre d'un caractère. Nous présentons ci-dessous Figure 2les deux types de n-grammes, caractères et mots (6)

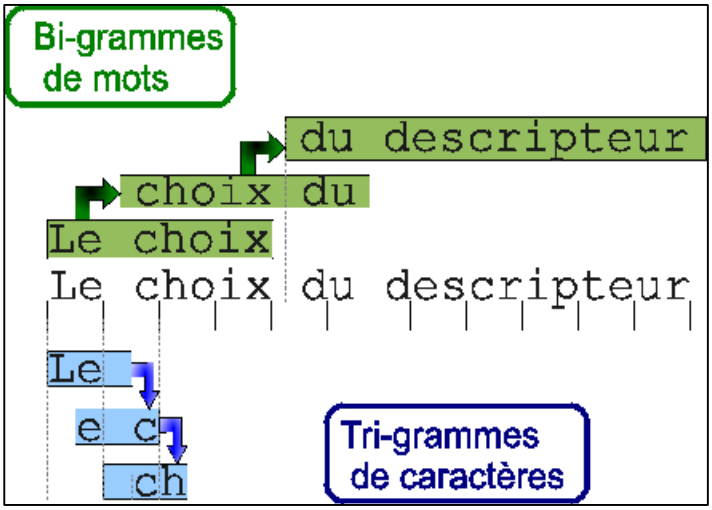

Figure 2 : Exemple de N-grammes de mots et de caractères

.

Notons que la technique des n-grammes est toujours très utilisée, pour représenter les textes, en raison de ses avantages, confirmés par plusieurs auteurs, dont les principaux sont les suivants :

- La représentation en n-grammes s'attaque aux documents presque dans leur état brut, contrairement aux autres représentations qui nécessitent des traitements purement linguistiques, comme les traitements d'élimination des mots vides, de stemming et de lemmatisation [Jalam](1) . Ces traitements améliorent la performance des systèmes à base sur les mots mais en contrepartie la mise en œuvre informatique de ces procédures est relativement lourde.
- Un autre point à souligner en faveur des n-grammes : quelques algorithmes de lemmatisation ne semble pas être en mesure de regrouper des termes comme automatisation**,** automatiser et automatique dans la même classe, par contre, le découpage en n-grammes est suffisant pour classer les trois termes dans la même classe, les tri-grams : aut, uto, tom, oma, mat, ati, permettent par une mesure de similarité d'affirmer que c'est l'automatique dont il est question.
- Le codage en n-grammes n'a pas besoin de segmenter le texte avant d'extraire les termes, ce qui offre à cette technique une caractéristique très intéressante, capable de représenter et traiter les langues pour lesquelles les frontières entre termes ne sont pas bien marquées comme l'allemand, le chinois, ou la langue arabe, dans laquelle les pronoms, sujets et compléments sont liés dé fois aux verbes, une seule chaîne de caractères unie représentant une phrase comme, par exemple, « kalamtouhou » ["je lui ai parlé"](1)

II.2.1.5 Représentation avec wordembedding

Le wordembedding désigne un ensemble de techniques de machine Learning qui visent à représenter les mots ou les phrases d'un texte par des vecteurs de nombres réels, décrits dans un modèle vectoriel (ou VectorSpace Model).

Ces nouvelles représentations de données textuelles ont permis d'améliorer les performances des méthodes de traitement automatique des langues NLP (Natural LanguageProcessing), comme le TopicModeling ou l'analyse des sentiments.

Le wordembedding repose sur la théorie linguistique fondée par ZellingHarriset connue sous le nom de « DistributionalSemantics ». Cette théorie considère qu'un mot est caractérisé par son contexte, c'est à dire par les mots qui l'entourent. Ainsi, des mots qui partagent des contextes similaires partagent également des significations similaires. Les algorithmes de wordembedding sont le plus souvent employés pour décrire des mots à travers de vecteurs numériques, mais ils peuvent également être utilisés pour construire des représentations vectorielles de phrases entières, de données biologiques comme les séquences d'ADN, ou encore des réseaux représentés comme des graphes.

Il existe plusieurs approches de wordembedding, les premières remontent aux années 1960 et reposent sur des méthodes de réduction de dimensionnalité. Plus récemment, de nouvelles techniques basées sur des modèles probabilistes et des réseaux de neurones, comme Word2Vec, ont permis d'obtenir de meilleures performances.

Dans l'exemple suivant, on schématise plusieurs termes ainsi que leurs représentations vectorielles bidimensionnelles. Les mots représentés, s'organisent en deux groupes, l'un relève de l'économie (trade, product, economy, euros*)* et l'autre appartient au champ lexical politique (rights, people, freedom, left*)*.

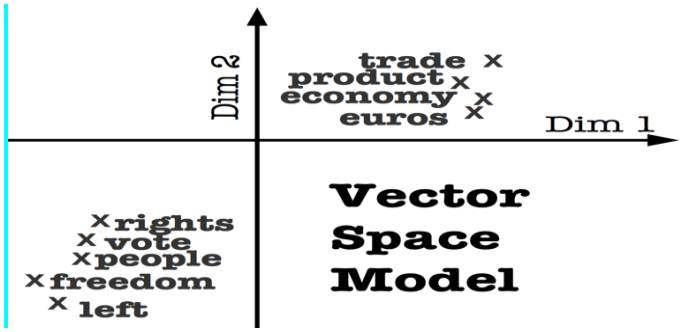

Figure 3 : l'espace entre les mots

- **Word2Vec :**

Word2vec est un algorithme de wordembedding parmi les plus connus. Il a été développé par une équipe de recherche de Google sous la direction de Tomas Mikolov. Il repose sur des réseaux de neurones à deux couches et cherche à apprendre les représentations vectorielles des mots composant un texte, de telle sorte que les mots qui partagent des contextes similaires soient représentés par des vecteurs numériques proches.

Cet algorithme possède deux architectures neuronales, appelées CBOW et Skip-Gram . CBOW reçoit en entrée le contexte d'un mot, c'est à dire les termes qui l'entourent dans une phrase, et essaye de prédire le mot en question. Skip-Gram fait exactement le contraire : elle prend en entrée un mot et essaye de prédire son contexte. Dans les deux cas, l'entraînement du réseau se fait en parcourant le texte fourni et en modifiant les poids neuronaux afin de réduire l'erreur de prédiction de l'algorithme.

Word2Vec possède différents paramètres, dont les plus importants sont :

- La dimensionnalité de l'espace vectoriel à construire, c'est à dire le nombre de descripteurs numériques utilisés pour décrire les mots (entre 100 et 1000 en général).
- La taille du contexte d'un mot, c'est à dire le nombre de termes entourant le mot en question (les auteurs suggèrent d'utiliser des contextes de taille 10 avec l'architecture Skip-Gram et 5 avec l'architecture CBOW).

Étant donné que Word2Vec repose sur des réseaux de neurone composé que de deux couches, cet algorithme est rapide à entraîner et à exécuter, ce qui se révèle être un avantage important par rapport à d'autres méthodes de wordembedding.

Le schéma suivant illustre l'architecture CBOW, appliquée au dernier vers du poème de Guillaume Apollinaire « Nuit rhénane ». Dans cet exemple, les mots considérés comme trop communs, appelés aussi « mots outils » ou «stop words » sont éliminés des données fournies au réseau de neurones. La première couche du réseau projette chaque mot du contexte (en gris) vers sa représentation vectorielle (en bleu). Puis la couche cachée (en vert) analyse ces représentations vectorielles afin de tenter de prédire le mot central (encadré en rouge). (23)

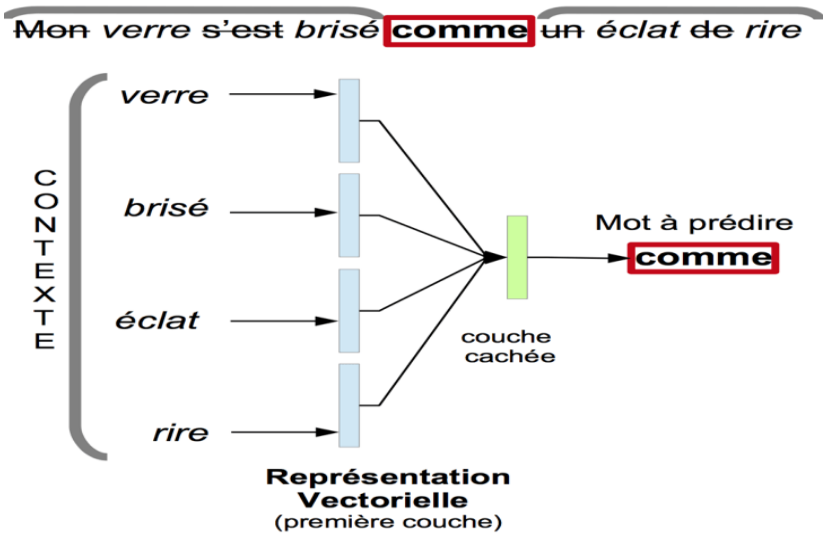

Figure 4 : architecture cbow

II.2.2La pondération des termes

La pondération des termes permet de mesurer l'importance d'un terme dans un document. Cette importance est souvent calculée à partir de considérations et interprétations statistiques. L'objectif est de trouver les termes qui représentent le mieux le contenu d'un document. Pour calculer la pondération on distingue les méthodes suivantesMesure

II.2.2.1 Mesure TF [la fréquence du terme]

Cette mesure prend en compte le nombre d'occurrences du terme dans le document. [Pondération locale]. Elle peut être utilisée telle quelle ou selon plusieurs déclinaisons [log [TF], présence/absence . . .].

II.2.2.2 Mesure IDF [l'Inverse de la Fréquence en Document]

Cette mesure prend en compte le nombre d'occurrence du terme dans le corpus.

IDF = log [N / Df]

Où:

Df: Le nombre de documents contenant le terme.

N : Le nombre total de documents de la base documentaire TFIDF [TermFrequency Inverse Document Frequency]

II.2.2.3 TFIDF

corrige la fréquence du terme [TermFrequency] en fonction de sa fréquence dans le corpus [Inverse Document Frequency]. La correction se fait en multipliant du rapport des n textes du corpus sur le nombre de documents contenant le terme car un terme qui apparait souvent dans la base documentaire ne doit pas avoir le même impact qu'un terme moins fréquent. La mesure TFIDF est calculée comme suit :

TFIDF = TermFrequency * Inverse Document Frequency

C'est à dire :

TFIDF[T, D] = TF[T, D]log [N/Df[T]]

Où:

TF[T, D] : est la fréquence du terme dans le document.

II.2.3 La réduction de la taille du vocabulaire

Le problème qu'on doit inévitablement résoudre est celui de la taille du vocabulaire car si on utilise tous les mots présents dans les documents de l'espace d'apprentissage, on se retrouve face à un espace vectoriel ayant une dimension très large. Le traitement d'un tel espace nécessitera beaucoup de mémoire et de temps de calcul et pourra nous empêcher d'utiliser des algorithmes d'apprentissage plus puissants. Pour résoudre ce problème on utilise les techniques de réduction de la taille du vocabulaire qui ont pour objectif de sélectionner ou d'extraire un sous-ensemble optimal de caractéristiques pertinentes pour un critère fixé. Ces techniques de réduction sont classées comme suit :

II.2.3.1 Sélection d'attributs

La réduction de la taille de l'espace d'apprentissage par la méthode de sélection d'attributs vise à diminuer la taille de l'espace d'apprentissage de$|T|$ à une taille $|T'| \ll |T|$ en sélectionnant uniquement un

Figure 5: Sélection des attributs

sous-ensemble des attributs existants

Pour simplifier les choses, considérons la **Figure 5**sélection des attributs où on dispose d'un ensemble d'attributs VT = {$t1$, $t2$, $t3$, $t4$, $t5$} réduit en VT' par le biais d'une sélection. On constate que les attributs n'ont subi aucune transformation. V^{T} est réduite en en $V^{T'}$ sélectionnant uniquement un sous-ensemble des attributs de V^{T}.

II.2.3.2 Extraction d'attribut

Contrairement aux techniques de sélection d'attributs qui visent à proposer par sélection un sous ensemble des attributs existants, l'extraction des attributs a pour objectif de proposer, via une synthétisation, un sous ensemble |T'| ≪ |T| composé de nouveaux attributs à partir des attributs existants. Ce processus consiste à créer à partir des attributs originaux un sous ensemble d'attributs synthétiques qui maximise l'efficacité de la classification et qui élimine les problèmes liés aux synonymies, homonymies, et polysémie. (1)

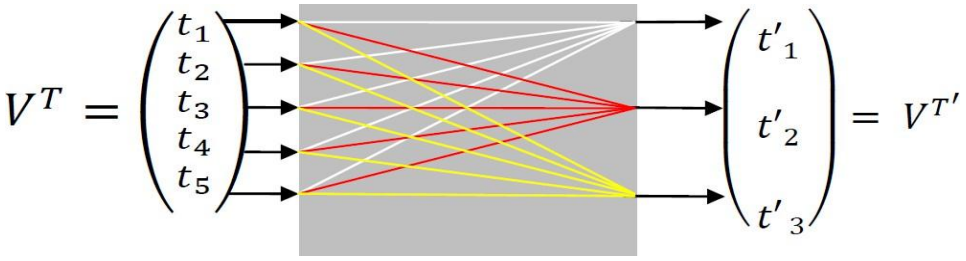

Figure 6: Extraction d'attributs

II.2.4 Choix de classificateur

La classification des textes contient le choix de technique d'apprentissage [ou classificateur] disponibles. Parmi les méthodes d'apprentissage les plus souvent utilisées figurent :

II.2.4.1 La régression logistique

La régression logistique est un algorithme d'apprentissage automatique supervisé développé pour l'apprentissage des problèmes de classification. Un problème d'apprentissage de la classification se produit lorsque la variable cible est catégorique. Le but de la régression logistique est de mapper une fonction des caractéristiques de l'ensemble de données aux cibles pour prédire la probabilité qu'un nouvel exemple appartienne à l'une des classes cibles.

II.2.4.2 les réseaux de neurones

S'appuyant sur l'inspiration du neurone biologique, le réseau de neurones artificiels [ANN] est une société d'agents connexionnistes qui apprennent et transfèrent des informations d'un neurone artificiel à l'autre. Au fur et à mesure des transferts de données entre neurones, une hiérarchie de représentations ou une hiérarchie de caractéristiques est apprise, d'où le nom d'apprentissage profond des représentations ou d'apprentissage profond.

II.2.4.3 Les plus proches voisins

Est l'un des algorithmes d'apprentissage automatique les plus simples, un algorithme non paramétrique, il suppose la similitude entre la nouvel donnée et les données disponibles et place la nouvel donnée dans la catégorie la plus similaire aux catégories disponibles, puis stocke toutes les données disponibles et classe un nouveau point de données en fonction de la similitude. Cela signifie que lorsque de nouvelles données apparaissent, elles peuvent être facilement classées dans une catégorie de suite de puits en utilisant l'algorithme K-NN. Il peut être utilisé pour la régression ainsi que pour la classification, mais il est principalement utilisé pour les problèmes de classification.

II.2.4.4 Les machines à vecteurs supports

La machine à vecteurs de support [SVM] est un algorithme d'apprentissage automatique pour l'apprentissage des modèles de classification et de régression.

Pour construire l'intuition, nous allons considérer le cas de l'apprentissage d'un modèle de classification avec SVM. Étant donné un ensemble de données avec deux classes cibles qui sont linéairement séparables, il s'avère qu'il existe un nombre infini de lignes qui peuvent discriminer entre les deux classes.

Le but du SVM est de trouver la meilleure ligne qui sépare les deux classes. Dans les dimensions supérieures, cette ligne est appelée un hyperplan [Un hyperplan est une ligne ou plus techniquement appelée discriminant qui sépare deux classes dans un espace à n dimensions. Lorsqu'un hyperplan est dessiné dans un espace 2D, il s'appelle une ligne. Dans l'espace 3-D, il est appelé un plan, et dans les dimensions supérieures à 3, le discriminant est appelé un hyperplan, Pour tout monde à n dimensions, nous avons n-1 hyperplans.

II.2.5 Evaluation du processus de classification

Certains principes d'évaluation sont utilisés de manière courante dans le domaine de catégorisation de textes. Les performances en termes de classification sont généralement mesurées à partir de deux

indicateurs traditionnellement utilisés à savoir les mesures de rappel et précision, ainsi .Initialement elles ont été conçues pour les systèmes de recherche d'information, mais par la suite la communauté de classification de textes les a adoptées. Formellement, pour chaque classe C_i, on calcule deux probabilités qui peuvent être estimées à partir de la matrice de contingence correspondante Tableau 1. (14)

II.2.5.1 Matrices de contingence

Les documents correctement classés sont des VP (vrai positif).

Les documents correctement non classés sont des VN (vrai négatif).

Les documents incorrectement classés sont des FP (faux positif).

Les documents incorrectement non classés sont des FN (faux négatif).

ClasseCi		**Jugement Expert**	
		Oui	Non
Jugement Classifieur	Oui	**VP_i**	**FP_i**
	Non	**FN_i**	**VN_i**

Tableau 1: matrice de contingence

Ces deux mesures sont définies de la manière suivante:

II.2.5.2 Le rappel : est la proportion des solutions pertinentes qui sont trouvées et toutes les solutions pertinentes**.**

II.2.5.3 La précision : est la Proportion de solutions trouvées qui sont pertinentes et les solutions non- pertinentes (refusé par le système).

D'autres mesures peuvent être calculées comme l'exactitude (accuracy) et F-mesure :

II.2.5.4 Accuracy (exactitude) : est la Proportion de solutions trouvées qui sont pertinentes et toutes les solutions les solutions.

II.2.5.5 **La mesure F score :** également appelé score F1, elle correspond à la moyenne harmonique du taux de précision et du rappel (15). Le score F1 est utilisé pour évaluer les problèmes de classification avec plus de deux classes [classification multi -classe] (16)

II.3 Problèmes de la classification de textes

Plusieurs difficultés peuvent s'opposer au processus de classification de textes, les principales sont les suivantes :

II.3.1 La redondance [la synonymie]

La redondance sémantique est le sens unique que peut y avoir plusieurs expressions distinctes, ou les différentes manières avec lesquelles on peut exprimer un sentiment, un fait, ou toute autre chose avec le langage naturel.

Cette subtilité du langage naturel est illustrée dans l'exemple suivant: "les États Unis d'Amérique", "le pays de l'oncle Sam" et "le gendarme du monde", ces expressions désignent toutes la même chose, mais lors de la représentation vectorielle ces séquences de mots seront représentées différemment. C'est pourquoi dans certains cas, l'utilisation du mot comme descripteur peut s'avérer superflu. (17)

Il est alors important de rassembler ces termes en un groupe sémantique commun. Pour y remédier, il est alors intéressant de concevoir une ontologie afin de cerner les sens des termes, naturellement, cela engendre des coûts supplémentaires pour sa réalisation et sa maintenance. (11)

II.3.2 L'ambiguïté [Polysémie]

A la différence des données numériques, les données textuelles sont sémantiquement riches, du fait qu'elles sont conçues et raisonnées par la pensée humaine. À cause de l'ambigüité, les mots sont parfois de mauvais descripteurs ; par exemple le mot avocat peut désigner le fruit, le juriste, ou même au sens figuré, la personne qui défend une cause.

II.3.3 La graphie

Un terme peut comporter des fautes d'orthographe ou de frappe comme il peut s'écrire de plusieurs manières ou s'écrire avec une majuscule. Ce qui va peser sur la qualité des résultats. Parce que si un terme est orthographié de deux manières dans le même document [par exemple : Ghelizane, Relizane], la simple recherche de ce terme avec une seule forme graphique néglige la présence du même terme sous d'autres graphies.

II.3.4 Complexité de l'algorithme d'apprentissage

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.

II.3.5 Présence-Absence de termes

La présence d'un mot dans le texte indique un propos que l'auteur a voulu exprimer, on a donc une relation d'implication entre le mot et le concept associé, quoiqu'on sache très bien qu'il y a plusieurs façons d'exprimer les mêmes choses, dès lors l'absence d'un mot n'implique pas obligatoirement que le concept qui lui est associé est absent du document. Cette réflexion pointue nous amène à être attentifs quant à l'utilisation des techniques d'apprentissage se basant sur l'exclusion d'un mot particulier.

II.3.6 Les mots composés

La non prise en charge des mots composés comme : Arc-en-ciel, peut-être, etc... Dont le nombre est très important dans toutes les langues, et traiter le mot Arc-en-ciel par exemple en étant 3 termes séparés réduit considérablement la performance d'un système de classification néanmoins l'utilisation de la technique des n-grammes pour le codage des textes atténue considérablement ce problème des mots compos

III. Concepts fondamentaux de l'apprentissage profond (deeplearning)

III.1 Réseaux de neurones artificiels [ANN]

Les ANN sont des réseaux à base des réseaux de neurones formels. Ce sont une société d'agents connexionnistes qui apprennent et transfèrent des informations d'un neurone artificiel à l'autre. Au fur et à mesure des transferts de données entre neurones, une hiérarchie de représentations ou une hiérarchie de caractéristiques sont apprise, et si le system est complexe alors la hiérarchie va être profonde donc le réseau neuronal artificiel va être réseau neuronal profond, d'où le nom d'apprentissage profond est apparu.

Un réseau de neurone artificiel constitué de différentes couches : une d'entrée, une de sortie, et entre les deux, éventuellement une ou plusieurs couches cachée[s]. À l'intérieur d'une même couche, les neurones ne sont pas reliés entre eux. Mais ils sont, en général, connectés à tous les neurones des couches suivantes. (21)

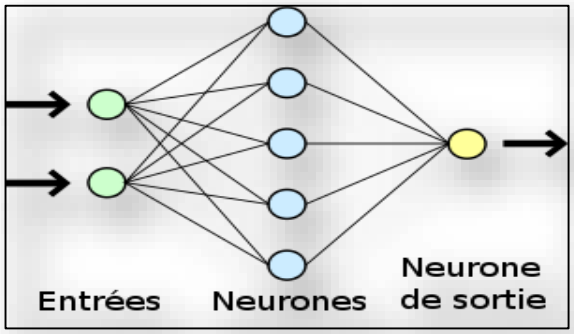

Figure 7 : représentation simplifier d'un réseau de neurone

III.2 Les principales composantes du réseau de neurones

III.2.1 Neurones: [ensemble de fonctions]

Ils prennent une donnée d'entrée et produisent une donnée de sortie. Un certain nombre de neurones sont groupés en couches. Tous les neurones du même groupe remplissent un type de fonction similaire.

III.2.2 Couches: [groupement de neurones]

Les couches contiennent des neurones et aident à faire circuler l'information. Il existe principalement deux couches dans un réseau de neurones la couche d'entrée [input layer] et la couche de sortie [output layer].

III.2.3 Poids et biais [valeurs numériques]

Les poids et biais sont des variables du modèle qui sont mises à jour pour améliorer les performances du réseau. Un poids est appliqué à l'entrée de chacun des neurones pour calculer une donnée de sortie. Les réseaux de neurones mettent à jour ces poids de manière continue.

Les biais sont également des valeurs numériques qui sont ajoutées une fois que les poids sont appliqués aux valeurs d'entrée.

Les poids et les biais sont donc en quelque sorte des valeurs d'auto-apprentissage pour les réseaux de neurones.

III.2.4 Fonction d'activation: [algorithmes mathématiques appliqués aux valeurs de sortie]

La fonction d'activation est une formule mathématique activée dans certaines circonstances. Elles lissent ou normalisent la donnée de sortie qui est comprise entre $-\infty$et$+\infty$ avant qu'elle ne soit transmise aux neurones suivants. Cette valeur est calculée par l'équation :

$$Y=\sum[weight*input]+bias$$

Ces fonctions aident les réseaux de neurones à apprendre et à s'améliorer.Il existe un grand nombre de fonctions d'activation, parmi:

III.2.4.1 Fonction d'activation binaire

L'activation de pas binaire peut être utilisée pour mapper vers des sorties binaires au moment de la prédiction, mais elle n'est pas différenciable à 0, Donc elle empêche son utilisation pour créer la fonction de perte au moment de l'apprentissage [pas de mise à jour des poids et biais] .la figure suivante représente le graphe de la fonction d'activation.

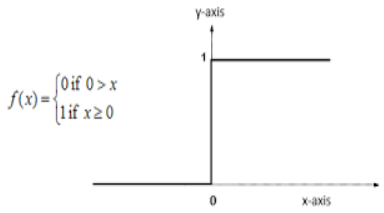

Figure 8 : graphe de la fonction binaire

III.2.4.1 Fonction d'activation linéaire

C'est la fonction la plus basic parmi les fonctions d'activation, elle représente une ligne droite et ne fournit aucune non-linéarité. Elle est souvent utilisée dans le nœud de sortie, lorsque la cible est une valeur réelle ou la sortie produite est proportionnelle à l'entrée, mais elle ne peut pas gérer la complexité des données.

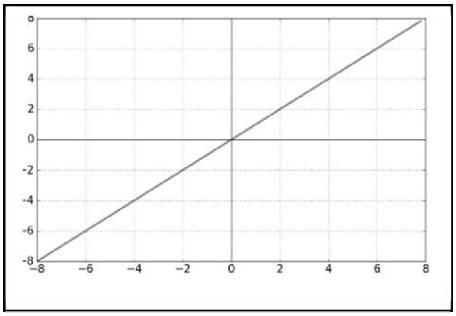

Figure 9: graphe des fonctions linéaire

f[z] = z

Fonctions d'activion non linéaire

- **Sigmoïde**

C'est la fonction la plus utiliser, elle génère une valeur entre [0, 1], ce qui est intéressant pour effectuer des calculs qui doivent être interprétés comme des probabilités. En outre, il normalise les données et construire des fonctions de perte dérivées de modèles de maximum de vraisemblance.

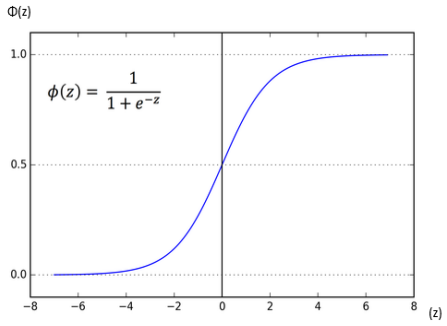

Figure 10 : graph de la fonction sigmoide

- **Unité linéaire rectifiée [ReLu]**

C'est une fonction qui y avait une popularité dans le domaine de deeplearning, cette fonction converge plus rapidement, et elle peut activer tous les neurones au même temps.

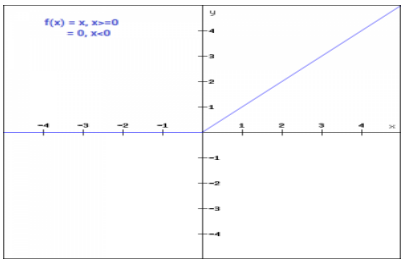

Figure 11 : graph de la fonction relu

- **Fonctions de tangente hyperbolique [tanh]**

Tanh est similaire à celle de sigmoide , la seul déférence est quel génère une valeur entre [-1, 1]. Cependant, elle rend moins bien compte des relations et elle est plus lente à converger.

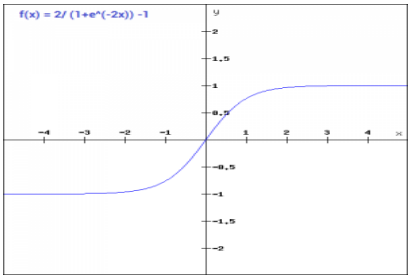

Figure 11:graph de la fonction tanh

- **Unité linéaire rectifiée [ReLu]**

C'est une fonction qui y avait une popularité dans le domaine de deepl earning, cette fonction converge plus rapidement, et elle peut activer tous les neurones au même temps.

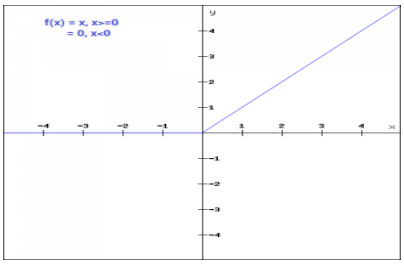

Figure 12 : graphe de la fonction relu

- **Softmax**

Softmax est définie généralement comme une combinaison de plusieurs sigmiod, elle est utilisée dans la plupart des cas dans les problèmes de classification multi class dans la

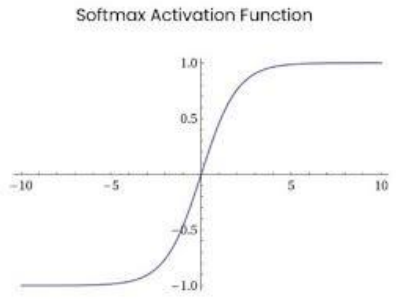

Figure 13 : graphe de la fonction softamax

III.3 Apprentissage profond [Deeplearning]

L'apprentissage profond est un domaine étendu de l'apprentissage automatique et l'une de ces principales technologies(23), il s'appuie sur des réseaux de neurones artificiels [avec plusieurs couches], où la machine est capable d'apprendre par elle-même.

Le terme «profond» dans l'apprentissage profond fait référence à la profondeur de l'architecture de réseau de neurones artificiels, et «apprentissage» signifie l'apprentissage via le réseau artificiel lui-même (20).

La machine dans l'apprentissage automatique se comporte en fonction des informations données par l'humain , contrairement à l'apprentissage profond qui 'est un système avancé basé sur le cerveau humain et qui comporte des vaste réseaux de neurones artificiel, ces neurones sont basé sur des approches mathématique qui interconnecte entre eux pour traiter , mémoriser des informations , comparer des problèmes où des situations quel quant avec des situations similaires passés et analyser les solutions. (27)

l'apprentissage profond permet aux algorithmes de réseau de neurone artificiel de bien fonctionner dans la construction de modèles de prédiction autour de problèmes complexes tels que la vision par ordinateur,

l'auto-conduite, la reconnaissance vocale(20) , Et peut être aussi avéré très utile dans le domaine du texte, de l'image .

Il est connu aussi comme un modèle de réseau avec des neurones ayant plusieurs paramètres et avec des couches entre l'entrée et la sortie, donc il suit l'approche de l'architecture de réseau de neurone artificiel.

L'avantage de L'apprentissage profond par rapport aux méthodes traditionnelles d'apprentissage automatique, réside dans ses fonctionnalités et ses représentations qui donnent un vue hiérarchique, et rend les algorithmes très robustes.

L'apprentissage profond est connu par deux principaux réseaux, les réseaux de neurones convolutifs et les réseaux de neurones récurrents, ce dernier est l'objet de notre travail.

III.3.1 Réseaux de neurones convolutifs

Les réseaux de neurones convolutifs[CNN] sont un type spécifique de systèmes de réseaux de neurones qui sont particulièrement adaptés aux problèmes de vision par ordinateur tels que la reconnaissance d'image, la reconnaissance de l'écriture manuscrite et la détection de visage.

Leur structure est basée sur l'échantillonnage d'une fenêtre ou d'une partie d'une image, la détection de ses caractéristiques, puis l'utilisation des caractéristiques pour créer une représentation. Cela conduit à l'utilisation de plusieurs couches, ces modèles ont donc été les premiers modèles d'apprentissage profond.

Les CNN ont tendance à être très complexes en termes de taille de paramètres et de nombre de calculs. Ainsi, la mise en œuvre des CNN nécessite de grandes exigences de calcul et de stockage.

Un réseau CNN est généralement composé de plusieurs couches telles que :

1) **La couche de convolution :** est composé de filtres et de cartes de caractéristiques.

- **Filtres** : l'un des principaux composants des CNN, ce sont des neurones dans la couche convolutionnelle, qui sont des matrices carrées et qui ont des dimensions **nK × nK**, où **nK** est un entier qui est généralement un petit nombre comme 3 ou 5. Parfois, les filtres sont également appelés noyaux. L'utilisation de noyaux provient des techniques classiques de traitement d'image. ils sont passés sur les pixels de l'image d'entrée pour capturer un ensemble spécifique de caractéristiques dans un processus appelé convolution. La sortie d'un filtre est appelée une carte des caractéristiques.

La première étape pour comprendre les CNN est de comprendre la convolution.

- **convolution** : la convolution est le processus par lequel une fonction est appliquée à une matrice pour extraire des informations spécifiques de la matrice. La fonction est implémentée comme une

fenêtre glissante à travers la matrice, et elle est plus communément appelée un filtre convolutif ou un noyau. Les deux termes sont utilisés de manière interchangeable dans la littérature.

- Les cartes d'entités sont les sorties d'un filtre dans une couche convolutionnelle. Il met en évidence certains modèles de l'image d'entrée tels que les lignes horizontales, les lignes verticales, les bords, etc. Ces cartes de caractéristiques des différents neurones empilés ensemble forment une couche neuronale convolutionnelle et permettent à la couche d'apprendre des modèles et des caractéristiques complexes d'une image. En se déplaçant plus profondément dans le réseau de neurones convolutifs, les entrées d'une couche convolutionnelle plus profonde sont les cartes de caractéristiques de la couche précédente.

2) **Couche de mise en commun [pooling]**

La mise en commun est la deuxième opération fondamentale dans les CNN. Cette opération est beaucoup plus facile à comprendre que la convolution. Les couches de pooling suivent généralement une ou plusieurs couches convolutives.

L'objectif de cette couche est de réduire ou de sous-échantillonner la carte d'entités de la couche convolutive.

Il résume les caractéristiques d'image apprises dans les couches réseau précédentes. Cela permet également d'éviter que le réseau ne se suralimente. De plus, la réduction de la taille d'entrée est également de bon augure pour les coûts de traitement et de mémoire lors de la formation du réseau. La couche de regroupement peut être considérée comme une fonction d'agrégation qui consolide les caractéristiques apprises et extrait les caractéristiques essentielles des couches précédentes. Il n'effectue aucune transformation multiplicative sur les cartes d'entités en entrée comme on le voit dans la couche convolutionnelle. Les fonctions d'agrégation exécutées par la couche pooling incluent max, sum et average. La fonction d'agrégation la plus fréquemment utilisée dans la pratique le max et qui est communément appelée « MaxPool ».

Les fonctions d'agrégation de la couche de pooling servent à filtrer les couches.

L'avantage essentiel de la couche de pooling est sa capacité à injecter l'invariance de localisation dans le réseau. L'invariance d'emplacement signifie que les entités peuvent être détectées par le réseau, où qu'elles se trouvent sur l'image. La couche de pooling applique sa fonction d'agrégation à tous les canaux de l'entrée.

3) **Blocs de construction d'un CNN**

La couche de réseau entièrement connecté [FCN] est un réseau neuronal à feedforward régulier ou perceptron multicouche. Ces couches ont généralement une fonction d'activation non linéaire. Dans tous les cas, le FCN est la dernière couche du réseau de neurones convolutifs. Dans ce cas, une activation softmax est utilisée pour sortir les probabilités qu'une entrée appartient à une classe particulière.

Avant de passer une entrée dans le FCN, la matrice d'image devra être aplatieles opérations de convolution et de pooling pour créer les couches utilisées dans les CNN.

III.3.2 Réseaux de neurones récurrents

Les réseaux de neurones récurrents sont considérer comme un type des réseaux de neurones pour le traitement des données séquentielles [texte, audio, vidéo…], Il s'agit des neurones interconnectées interagissant non-linéairement et pour lequel il existe au moins un cycle dans la structure. (28)

Les réseaux de neurones récurrents comportent des cycles au sein du graphe de neurones (29) comme est illustrer dans la figure. L'objectif principale derrière ce type d'architecture est de pouvoir manipuler des séquences de vecteurs d'entrée, représentant chacun un événement qui a une signification temporel (30)

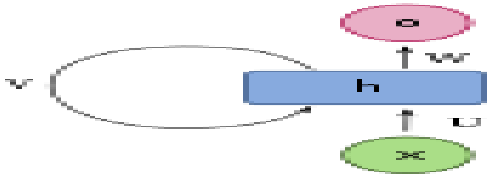

Figure 14 : représentation basic d'un RNN

Les RNN comportent différent types d'architecture, dont :

- **Un à un [one to one]**

  Architecture « un à un »veut dire que une entrée et une sortie, mathématiquement $T_X = T_Y = 1$, ce type à une architecture traditionnelle qui est la plus basic et la plus simple parmi les types des RNN, un exemple qui peut illustrer ce type la classification d'images, où l'entrée est une image et la sortie est la classe auquel elle appartient.

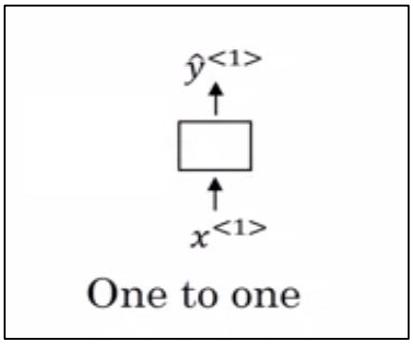

Figure 15 : architecture un à un

- **Un à Plusieurs [one to many]**

C'est un type d'architecture RNN appliquée dans des situations où il existe une seule entrée et plusieurs sorties [T_x=1, T_y>1], montré dans la Figure 16.

Un exemple de son application serait la génération de musique, l'entrée est une seule note et la sortie serait la génération d'un morceau de musique [sortie multiple].

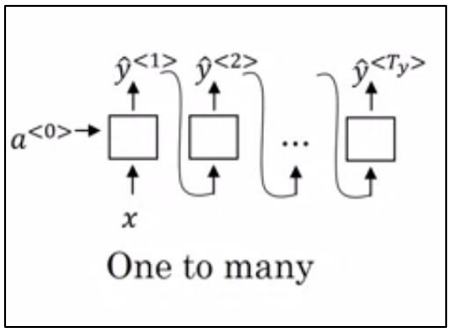

Figure 16 : architecture un à plusieurs

- **Plusieurs à un [many to one]**

Ce prend plusieurs entrées et donne une seule sortie [Tx>1,Ty=1], l'exemple typique pour cette architecture est l'analyse des sentiments.

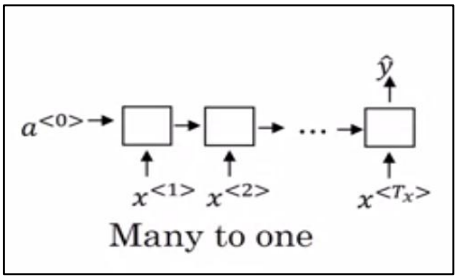

Figure 17 : architecture plusieurs à un

- **Plusieurs à Plusieurs [many to many]**

Ce types RNN [T_x>1,T_y>1] prend plusieurs entrées et donne plusieurs sorties, ce dernier peut avoir deux modèles comme il est représenté dans les Figures18et Figures19.

- Tx= Ty, Cela veut dire que les couches d'entrée et de sortie ont la même taille, en d'autres termes chaque entrée ayant une sortie, un exemple pour ce modèle est la reconnaissance d'entité nommée.

- Tx != Ty, implique que les couches d'entrée et de sortie sont de tailles différentes, comme un exemple la traduction automatique.

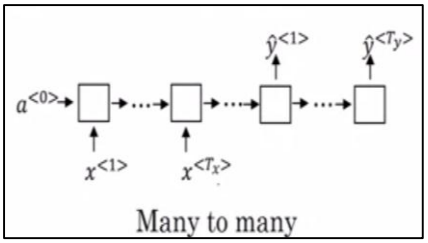

Figure 18: architecture plusieurs à un plusieurs 1

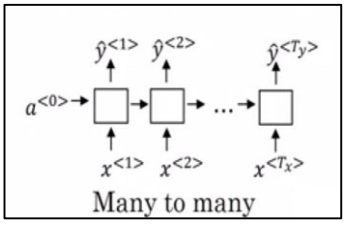

Figure 19 : architecture plusieurs à un plusieurs 2

Les réseaux de neurones récurrents sont adaptés pour es données d'entrée de taille variable. Ce sont des modèles capables de prendre en compte un contexte dans leur fonction de décision. Ils sont pour cela adaptés à plusieurs tâches de Traitement Automatique des Langues [TAL], notamment celles qui consistent à prédire une information séquentielle.

L'avantage des RNN réside dans leur capacité à prendre en compte le contexte passé lors du traitement de l'information courante .Suivant cette modélisation.

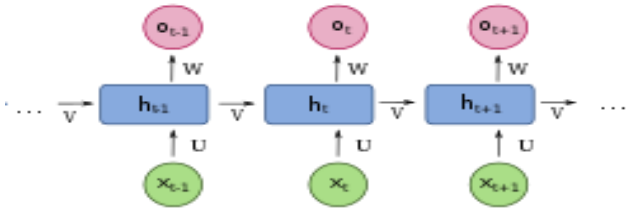

Figure 20 : un réseau RNN détaillé

La figure ci-dessus montre l'architecture RNN, qui prend en entrée une séquence d'événements **x = [x1, x2, . . . ,xT]** et définit la séquence d'états cachés **h = [h1, h2, . . . , hT]** pour produire la séquence de vecteurs de sortie **y = [y1, y2, . . . , yT]** en itérant de **t = 1** à **T .**

Ci-dessous montre l'équation et la figure représente une cellule RNN pour la cellule de la sortie :

$$h_t = \tanh[W_{xh}x_t + W_{hh}h_{t-1}]$$

$$y_t = W_{hy}h_t + b_y$$

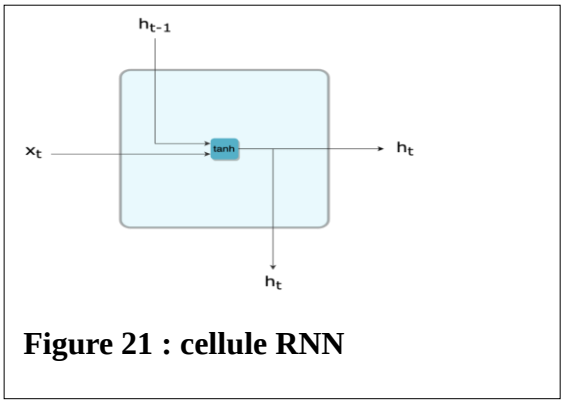

Figure 21 : cellule RNN

Où :

T est le nombre total de vecteurs d'entrée.

$\mathbf{W}_{\alpha\beta}$est la matrice de poids entre les couches α et β,

$\mathbf{b}_{\beta}$est le vecteur de biais de la couche β.

fonctiontanh utilisée dans le cas des RNN la fonction d'activation tangente hyperbolique.

III.3.3 Réseau de neurone récurrent bidirectionnel [BRNN]

Un RNN bidirectionnel [BRNN] est un modèle proposé pour pallier le problème des informations future à un état donné que les RNN ne pouvaient pas les considérer. Les BRNN utilise un algorithme de retro-propagation pour le régler.

Cette méthode divise les états réguliers de neurone de RNN en avant et en arrière .En d'autres termes, il y aura deux réseaux récurrents différents. Ces deux réseaux se connectent à la même couche de sortie pour générer des informations de sortie. Avec cette structure, les situations passées et futures des intrants séquentiels dans un délai donné sont évaluées sans délai(31).L'architecture du BRNN est illustrée dans la figure suivante :

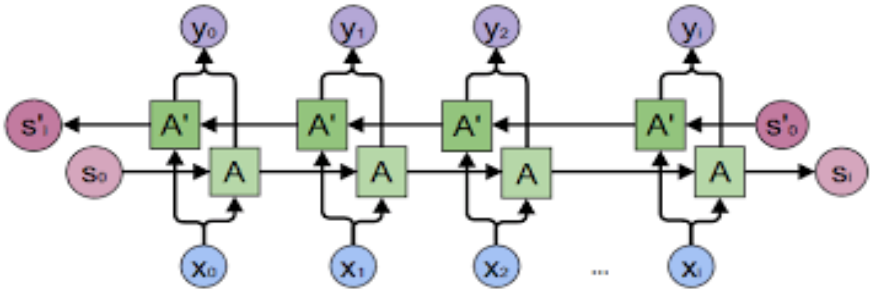

Figure 22 : réseau BRNN

III.3.3.1 Problème des réseaux RNN et BRNN

Les RNN classiques ont des difficultés à traiter les séquences relativement longues notamment celles contenant plus de 10 événements (32) Ces difficultés entrainent des erreurs, les erreurs résulte le décroit de la rétro propagation du gradient qui est nommé la « dissipation du gradient » [vanishing gradient en anglais].

L'augmentation de ce dernier d'une manière exponentielle par rapport à l'échelle du temps est appelé « l'explosion du gradient ».

III.3.3.2 LSTM (long short term memory)

Un réseau« Long Short-Term Memory » ou LSTM est un réseau neuronal, qui a une structure considérée comme une version étendue de la structure RNN.

Il a été introduit la première fois par Hochreiter et Schmidhuberen[1997](33) , pour éviter les problèmes de dépendance à longue terme présente dans les RNN.

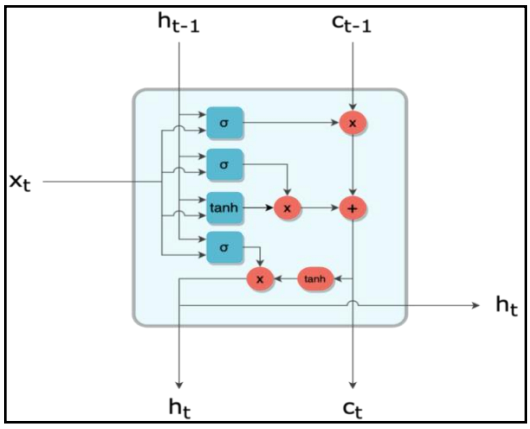

Figure 23:Représentation simplifiée d'une cellule LSTM

Un LSTM a un état caché [ht] qui est similaire à celui des RNN, mais sa particularité réside dans la manière selon laquelle l'état caché est géré. Dans le cas des RNN simples, le traitement de la récurrence est symbolisé par la fonction d'activation qui est généralement tanh. En ce qui concerne les LSTM, ce traitement est remplacé par une « cellule mémoire », la cellule est représentées dans la Figure 23, cette dernière est caractérisée par un nœud central, contenant l'état cellulaire [ct][ou mémoire interne de la cellule],qui a comme rôle la mise à jour des paramètres de manière additive au lieu de la manière multiplicative habituelle qui est la source du problème de RNN [dissipation du gradient et explosion du gradient].(34)

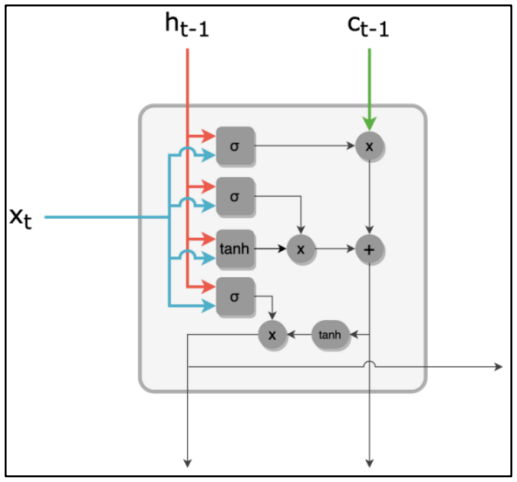

Figure 24 : les entrées du cellule LSTM

Une cellule LSTM prend comme entrée trois paramètres, l'état de la sortie de la cellule précédente [h_{t-1}], l'état cellulaire de la cellule précédente [c_{t-1}] et une entrée [x_t].

Le flux d'information en entrée et en sortie de la cellule LSTM est contrôlé par un mécanisme, ce mécanisme permet de supprimer ou d'ajouter des informations à l'état de la cellule en trois étapes différentes appelées le mécanisme des portes, dont on trouve :

III.3.3.2.1 la porte d'oubli [Forget Gate[ft]]

Une porte d'oubli est chargée de décider s'il faut supprimer les informations qui ne sont plus nécessaires au LSTM de l'état cellulaire ou les garder.

Cette porte reçoit deux entrées $[h_{t-1}]$ l'état caché de la cellule précédente [la sortie de la cellule précédente]et l'entrée $[x_t]$ de ce pas de temps particulier. Cette étape peut être faite en deux étapes

- Les entrées $[h_{t-1,}][x_t]$seront multipliées par une matrice de poids $[W_f]$,ensuite l'ajout d'un biais$[b_f]$.à la fin, l'application de la fonction sigmoïde au valeur calculer , en appliquant l'équation suivante :

$$f_t=[W_f[h_{t-1},x_t]+ b_f]$$

Pour mieux comprendre cette étape, la Figure25explique cette étape :

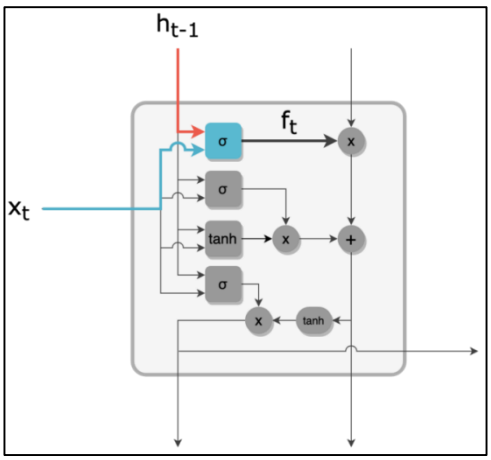

Figure 25:porte d'oublie

La sortie [f $_t$]montrer dans la figure donne un vecteur, avec des valeurs allant de 0 à 1, la fonction sigmoïde est chargée de décider quelles valeurs conserver et lesquelles supprimer. Si un « 0 » est émis pour une valeur particulière dans l'état de la cellule, cela signifie que la porte d'oubli veut que l'état de la cellule oublie complètement cette information. De même, un « 1 » signifie que la porte d'oubli veut se souvenir de toute cette information.

- Ce vecteur de sortie de la fonction sigmoïde est multiplié ensuite par l'état de la cellule précédente montré dans la Figure 26.cette étape est calculer avec l'équation suivante

$$C^{'}=f_t*c_{t-1}$$

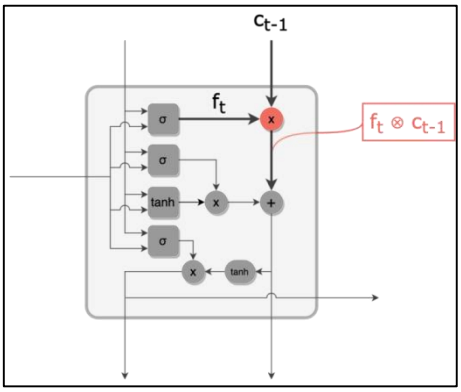

Figure 26:étape 1 de l'état de la cellule

Ainsi lors de cette multiplication, les valeurs proches de 0 dans C' ne seront pas autorisées à passer, à l'inverse, si les valeurs sont proches de 1 les informations seront préservées et autorisées à passer.

III.3.3.2.2 La porte d'entrée [Input Gate [it]]

La porte d'entrée est responsable à l'ajout des informations dans l'état de la cellule en se basant sur les informations provenant de l'entrée [x_t] à l'instant t.

Cet ajout des informations est essentiellement un processus en trois étapes.

- La première étape consiste à Observer et contrôler quelles valeurs doivent être ajoutées à l'état de la cellule en impliquant une fonction sigmoïde, cette étape agit comme un filtre pour toutes les informations de [h_{t-1}] et [x_t].le fonctionnement est très similaire à la porte d'oubli, Ce qui maintien l'équation suivante :

$$i_t = [W_i[h_{t-1}, x_t] + b_i]$$

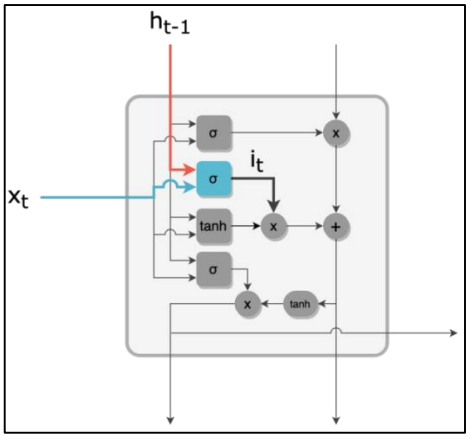

Figure 27 : étape 1 de la porte d'entrée

- Création d'un vecteur contenant toutes les valeurs possibles pouvant être ajoutées [telles que perçues à partir de h_t-1 et x_t] à l'état de la cellule. Cela se fait à l'aide de la fonction tanh, qui génère des valeurs de -1 à +1.

$$K_t=\tanh[W_c[h_{t-1},x_t]+ b_c]$$

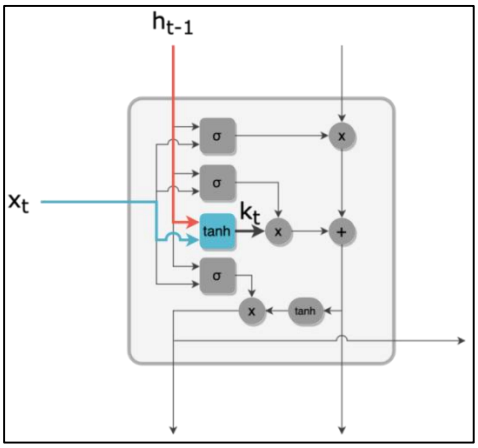

Figure 28 : étape 2 de la porte d'entrée

- Multiplier la valeur du filtre régulateur [fonction sigmoïde] au vecteur créé [la fonction tanh].

$$C"_t=i_t*K_t$$

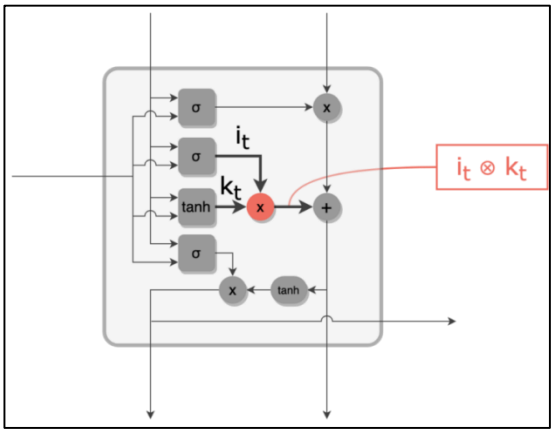

Figure 29 : étape 3 de la porte d'entrée

L'état de la cellule filtré [obtenu à la dernière étape de la porte d'oubli] est mis à jour grâce au vecteur candidat filtré [obtenu à la dernière étape de la porte d'entrée]. La mise à jour est une simple addition terme à terme de ces deux vecteurs. On obtient alors l'état cellulaire c_t de cette cellule.

$$C_t = C'_t + C''_t$$

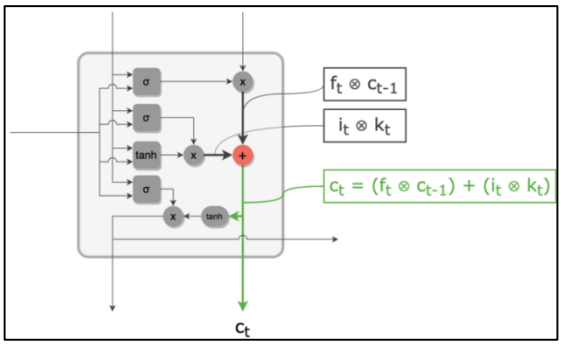

Figure 30:étape 2 de l'état de la cellule

Une fois ce processus en trois étapes terminé, nous nous assurons que seules les informations importantes et non redondantes sont ajoutées à l'état de la cellule.

III.3.3.2.3 la porte de sortie [o_t][Output Gate]

La dernière étape de ce mécanisme, qui permet de sélectionner les informations utiles à partir de l'état actuel de la cellule et de leur afficher en tant que sortie.

Le fonctionnement d'une porte de sortie peut encore être décomposé en trois étapes :

- Réalisation d'un filtrer en utilisant les valeurs [h_{t-1}] et [x_t], multipliant ces valeurs vont être multiplié avec la matrice des poids [W] ensuit ajouter un biais [b]à la fin une fonction sigmoïde va être appliqué sur le vecteur de sorte qu'il puisse réguler les valeurs qui doivent être sorties du vecteur.

  Mathématiquement :

$$O_t = [W_o[h_{t-1}, x_t] + b_o]$$

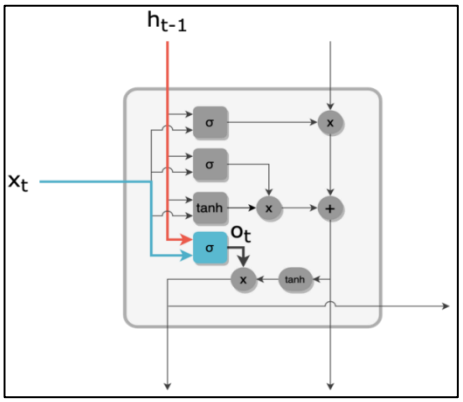

Figure 31:étape 1 de la porte de sortie

- Les valeurs du nouveau l'état cellulaire **[c_t]** sont ramenées à l'intervalle]-1, 1[par une activation tanh« **tanh[C_t] ».**

- Multiplier la valeur du filtre régulateur par le vecteur créé à l'étape 2, et l'envoyer en sortie [h_t] de la cellule en question à l'instant[t], ainsi qu'à l'état caché de la cellule suivante

$$h_t = O_t * \tanh[C_t]$$

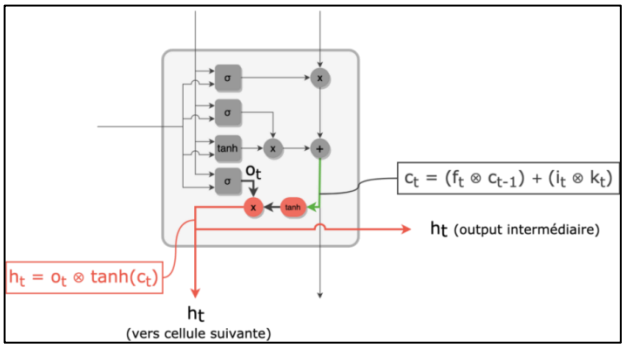

Figure 32 : étape 2 de la porte de sortie

Le but des cellules ainsi que les états cachés dans LSTM est que les cellules sont responsables sur les informations du passé qui vont être utiles dans les prédictions futur, les états cachés sont responsable à leur tour sur les informations du passé qui vont être utiles dans les prédictions courantes. Alors que dans RNN il existe seulement les états cachés.(35)

III.3.3.3 Gated recurrent units [GRU]

Les réseaux de neurones récurrents à portes[GRU]sont des variantes des réseaux LSTM, ils ont été introduits par cho et al en 2014 pour la réinitialisation et la mise à jour du contexte mémoire et pour fixer les problèmes de la structure LSTM, parmi, le grand nombre des paramètres dans cette dernière peut causer un débordement, en particulier pour les taches à faibles ressources, l'entrainement nécessite plusieurs mécanismes complexe qui peuvent rendre difficile l'ajustement des paramètres.

La structure des GRU est similaire à celle du LSTM sauf qu'elle comporte deux portes, l'utilité de cette structure est d'exposer son contenue de mémoire à chaque étape et équilibre la sortie entre l'état de mémoire précédente et le nouvel état de mémoire candidat.

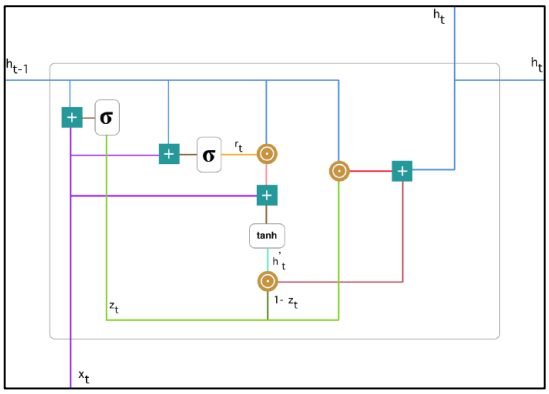

Figure 33 : une cellule GRU

Figure 34 : les différents opérations dans une cellule GRU

III.3.3.3.1 Porte de mise à jour [Update gate[z_t]]

Cette porte agit exactement de la même manière que les portes d'oubli et d'entrée du LSTM (34), elle décide des informations à conserver et celles à oublier à partir des états passés. L'équation ci-dessus montre les calcule.

$$z_t=[W_z.x_t+U_z.h_{t-1}]$$

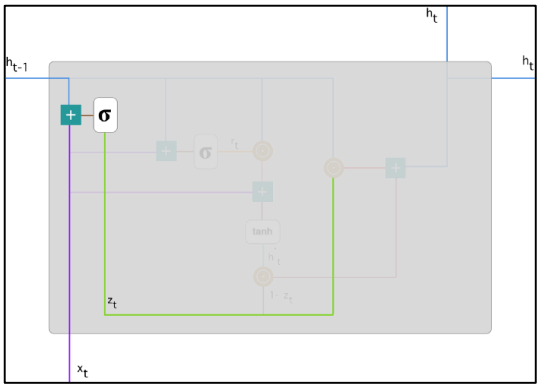

Figure 35:Porte de mise à jour

Les données d'entrées [xt] et l'ancien état caché [ht–1]sont concaténés après les multiplier avec les poids [W_z],ensuite ils passent par une fonction sigmoïde dont le rôle est de déterminer quelles sont les composantes importantes.

III.3.3.3.2 Porte de réinitialisation [reset gate[r_t]]

Cette porte sert à contrôler combien d'information passée que le réseau doit oublier. Celle-ci est calculée par la formule suivante:

$$r_t=[W_r.x_t+ U_r.h_{t-1}]$$

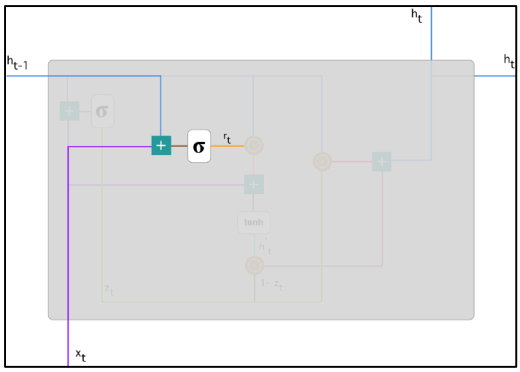

Figure 36:Porte de réinitialisation

L'état caché précédent [h_{t-1}], concaténé avec les données d'entrée [x_t], passe par une sigmoïde [pour conserver que les coordonnées pertinentes] puis est multiplié par l'ancien état caché, on n'en conserve pas donc que les coordonnées importantes [telles qu'elles] de l'état précédent [on a donc perdu une partie de l'état précédent dans cette porte].

III.3.3.3.3 Contenu actuel de la mémoire

Dans cette étape et avec l'utilisation de la porte de réinitialisation, Un contenu actuel de la mémoire [h'$_t$] est introduisez pour stocker les informations pertinentes du passé. Ce dernier est calculé suivant cette équation:

$$h'_t = \tanh[W.x_t + r_t.\Theta.U.\ h_{t-1}]$$

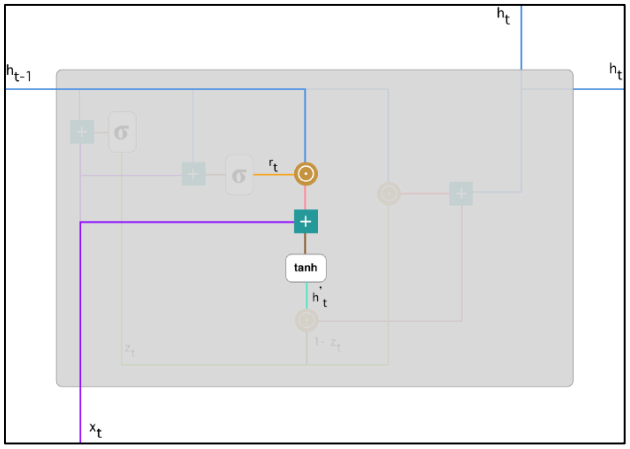

Figure 37:Contenu actuel de la mémoire

Où **Θ** est le produit de Hadamard [élément par élément] entre les deux vecteurs.

III.3.3.3.4 Mémoire finale au pas de temps actuel

Comme dernière étape, il faut calculer [h_t] qui contient les informations de la cellule actuelle et les transmettre au réseau, pour ce faire, la porte de mise à jour est nécessaire. Il détermine ce qu'il faut collecter du contenu actuel de la mémoire [h'$_t$] et ce qu'il faut des étapes précédentes [h_{t-1}]. Cela se fait comme suit:

$$h_t = z_t\ \Theta\ h_{t-1} + [1 - z_t]\Theta\ h'_t$$

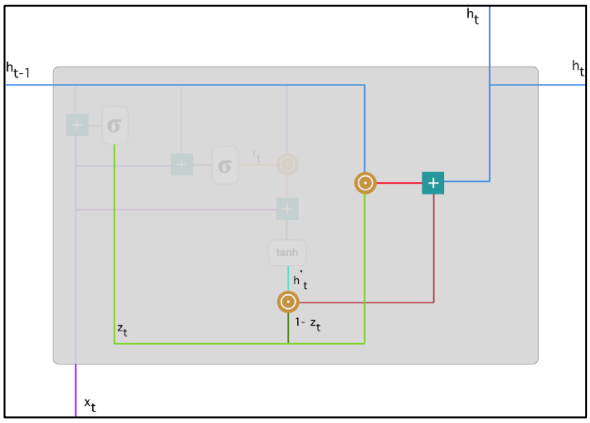

Figure 38 : Mémoire finale

1. Application de la multiplication élément par élément aux portes de mise à jour $[z_t]$ et $[h_{t-1}]$.
2. Application de la multiplication élément par élément à $[1- z_t]$ et $[h'_t]$.
3. Additionnez les résultats de l'étape 1 et 2.

III.3.3.4 Les réseaux de neurones LSTM bi directionnels [BILSTM]

La version LSTM de la structure BRNN s'appelle Bidirectionnel LSTM [BILSTM] qui peut être obtenue en remplaçant les deux couches récurrentes du BRNN par des cellules LSTM.

Cette version peut améliorer le rendement du modèle LSTM dans les processus de classification, il permet de porter un contexte long et tire profit de la structure en deux directions. (29)(31)

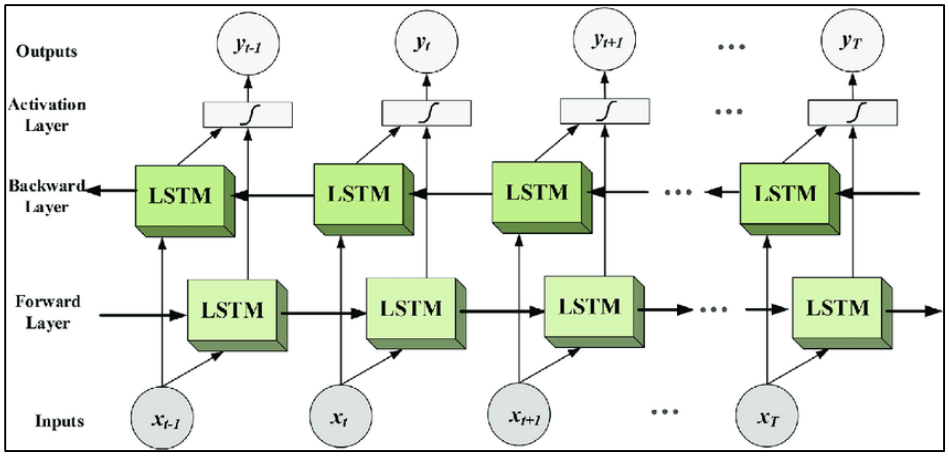

Figure 39 : architecture BILSTM

IV. Travaux connexes

Il existe plusieurs recherches portant sur la classification de texte [TC] en utilisant les modèles de l'apprentissage profond dans nombreuses langues, cependant pour la langue arabe, les recherches sont limitées. Nous citons dans ce qui suit quelques travaux récents :

➢ Classification des textes arabes en utilisant les réseaux de neurones convolutionnel (36)

Biniz et al. (36)Ont présenté l'algorithme Convolutional Neural Network [CNN], l'un des plus célèbres algorithmes d'apprentissage en profondeur. Le but de ce travail est de bénéficier des avantages de cet algorithme, qui ont été prouvés dans d'autres domaines, dans la classification de texte arabe.

Le dataset utilisé est une collection de textes, qui couvre la langue arabe moderne utilisée dans les articles de journaux. L'ensemble de données se compose de 319 254 124 mots et de 111 728 documents structurés en fichiers texte et collectés auprès de 3 journaux arabes en ligne : Assabah[www.assabah.ma], Hespress[www.hespress. com] et Akhbarona[www.akhbarona.com] en utilisant un processus d'exploration Web semi-automatique.

L'auteur a commencé d'abord par l'élimination des chiffres, des symboles et des mots vides ensuite, il a appliqué l'algorithme de stemming et la pondération TF-IDF. Enfin, la classification est réalisée à l'aide de l'algorithme CNN.

Pour tester l'influence de la taille des ensembles de données, différentes tailles d'ensembles de données sont entrainées [data_27k, data_55k, data_83k, data_111k]. Les résultats obtenus ont montré que :

- La précision est reliée à l'augmentation de l'ensemble de données c'est-à-dire plus la taille de l'ensemble de données utilisée est grande plus la précision obtenue est meilleure.
- Le modèle CNN peut efficacement composer la représentation sémantique des textes et capturer plus d'informations contextuelles sur les caractéristiques par rapport aux méthodes traditionnelles basées sur le modèle du sac de mots.
- Les données de formation et de test sont tirées de diverses distributions et sources, la précision supérieure à 92 % est estimée comme plus que satisfaisante par l'auteur.

➢ Classification des textes arabes en utilisant l'apprentissage profond (38)

Cette recherche se concentre sur la classification du texte arabe à l'aide d'un réseau de neurones à convolution [CNN], car il a obtenu d'excellents résultats dans différents types de projets de traitement du langage naturel [NLP].Les auteurs ont également introduit un nouvel algorithme qui est nommé Gstem pour regrouper les mots arabes similaires.

L'approche proposé est appliqué sur le Dataset SANAD qui possède un nombre total de 194 797 articles, les ensembles de données AlKhaleej et Akhbarona-Alanba ont sept catégories qui sont : Culture, Finance, Médical, Politique, Religion, Sports et Technologie. Quant à l'ensemble de données AlArabiya, il comporte six catégories : Culture, Finance, Médical, Politique, Sports et Technologie. Dans le travail suivant 80 % de cet ensemble de données est utilisé pour l'entrainement et 20 % pour le test.

Dans la phase de prétraitement le système commence d'abord par la suppression des chiffres, des signes de ponctuation, des espaces en doubles ensuite il passe à la normalisation et la suppression des mots qui ont moins de trois lettres. Les auteurs se sont basés sur Word2Vec pour la conversion des textes en espace vectoriel.

L'algorithme GStem est utilisé en tant que couche de prétraitement du CNN pour dériver statistiquement les mots arabes, en regroupant les mots similaires en fonction des lettres arabes supplémentaires par exemple, Regrouper les mots " سيقول " et " قال" dans la même racine " قال" qui signifie " dit" .

Ce travail a mené aux résultats suivants :

- L'utilisation de GStem comme étape de prétraitement augmente la précision du modèle CNN parce que le nombre de mots distincts a été réduit.
- Lors de l'utilisation de GStem , il ne faut que 4 époques pour atteindre la valeur maximale de la mesure de précision ; cependant , il faut 18 époques pour atteindre la valeur de précision maximale du modèle CNN-Normal.
- Le résultat montre que l'effet sur la précision lors de l'utilisation de CNN-GStem surpasse CNN-Normal.

➢ La classification des textes basée sur GRU combiné avec SVM (42)

Le Gated recurrent units[GRU] et la machine à vecteurs de support [SVM] ont été utilisées avec succès pour les systèmes de traitement du langage naturel [NLP] avec des résultats comparatifs remarquables.

Muhammad Zulqaranain et al. (42) Ont utilisé dans ce travail le GRU, un type de RNN créé pour pallier des problèmes dans l'architecture LSTM avec des portes de mise à jour et de réinitialisation,

comme est mentionné dans le chapitre 1. De plus, ils ont implémenté SVM qui a été initialement conçu comme classificateurs binaires,Hamlg.Pour combiner GRU et SVM , ils ont remplacé :

- Le softamax traditionnel de la couche de sortie par SVM.
- La fonction d'activation [tanh] par [LRelu], Il a été démontré que les unités LRelu offrent de meilleures performances que les non-linéarités.
- La fonction d'entropie croisée par une fonction basée sur la marge.

Concernant les ensembles de données expérimentales, ils contiennent des corpus chinois collectés par le Dr Li Rongla de l'Université de Fudan, le corpus se compose de 10 catégories sélectionné au hasard dont 1885 documents pour l'entrainement et 940 documents pour le test.

Les auteurs de ce travail ont conclu que le GRU-SVM a une meilleur capacité de classification de texte en terme de rappel, de précision et de F1 par rapport aux approches existantes des réseaux d' auto-encodeur profond [DABN] (44) , et de la mémoire bidirectionnelle à long terme avec couche convolutive[BILSTM-C] (42). Aussi la méthode proposée atteint de meilleures performances en particulier lorsque la taille du stockage est limitée.

- **Apprentissage profond combinant CNN et Bi-LSTM avec un classificateur SVM pour la classification des sentiments arabes (45)**

Ce travail présente une combinaison CNN avec BILSTM et un classificateur SVM linéaire pour la classification des sentiments arabes, le modèle proposé ne suit pas les CNN et Bi-LSTM classiques. Au lieu de cela, les couches entièrement connectées sont remplacées par un algorithme SVM linéaire pour effectuer la tâche de classification.

Pour cette approche, Omar Elharbi (45)a utilisé trois ensembles de données accessibles au public, à savoir LABR, OMCCA et l'ensemble de données présenté dans (46).Ces ensembles de données varient en taille et en forme d'écriture, où MSA(Arabe Standard Moderne) et différents dialectes arabes ont été détectés

Par exemple LABR contient plus de 63 000 critiques écrites principalement en MSA avec la présence de phrases dialectales. D'autre part, OMCAA se compose de plus de 28 000 critiques écrites principalement en dialectes jordanien et arabe saoudien. Le dernier ensemble de données présentait 2400 critiques écrites en dialecte jordanien avec la présence de MSA. La répartition entraînement / validation / test est fixée à 70%, 15%, 15% , respectivement.

Ce système commence par éliminer les lettres répétées, les signes diacritiques, les ponctuations, les chiffres, ensuite la normalisation est appliquée. L'auteur a introduit le modèle d'incorporation de

mots pré-entraîné qui s'est entraîné sur d'autres grands textes arabes (47) car la taille de l'ensemble de données adopté est relativement petite, ce qui peut conduire à des intégrations de mots de mauvaise qualité. Le modèle utilisé est une combinaison CNN et Bi-LSTM avec SVM, où CNN et Bi-LSTM sont utilisés pour la génération de vecteurs de caractéristiques et SVM pour la classification des sentiments.

Enfin, l'auteur a conclu que :

- Le modèle proposé avec skip-gram et CBOW surpasse les autres modèles de représentation dans tous les ensembles de données.
- Le combiné proposé surpasse tous les autres modèles [CNN et SVM].
- Une amélioration significative du score F1 du modèle proposé dans tous les ensembles des données où le plus élevé est atteint dans le troisième ensemble de données avec environ 8% par rapport à CNN et SVM lorsque skip-gram est utilisé et environ 7% lorsque CBOW est utilisé.
- Le modèle proposé fonctionne très bien en terme de rappel et de précision dans tous les ensembles de données avec un peu de compromis, où le rappel est le plus élevé par rapport à tous les modèles utilisés avec 91,6% , d'autre part, la valeur de précision la plus élevée est d'environ 91% et atteinte par CNN .

➢ Synthèse des travaux

- Le premier travail de Biniz avec l'algorithme CNN sur les textes arabes a montré que plus la taille du dataset est grande , la précision est meilleur , ce qu' 'il nous mène à choisir ce dataset dans notre travail en utilisant RNN et le modéles de représentation "Bag of words " et " stemming " avec la pondération TF-IDF pour arriver à comparer nos modèles RNN à ceux du CNN .
- Le deuxième travail ou les auteurs ont utilisé l'algorithme CNN -Gstem sur les textes arabes a montré que plus le nombre de mots distincts est réduit, la précision est meilleur, ce qui nous incite à utiliser des méthodes de réduction dans notre approche.
- Le troisième travail qui concerne la classification des textes chinois a montré que le modèle GRU combiné avec SVM a présenté de meilleurs résultats en termes de précision, rappel et F1 score par rapport aux approches CNN et BILSTM, donc, on doit voir si ça va donner de bon résultat aussi avec un texte arabe.
- Le dernier travail de la classification en utilisant l'algorithme BI-LSTM combiné avec CNN, l'auteur a trouvé que le modèle word2vec a donné de meilleurs résultats en terme de F1 score, avec un

certain compromis de la précision et le rappel. Ces résultats sont obtenus avec un test sur différents ensembles de données et par rapport aux deux algorithmes, l'un de l'apprentissage automatique SVM et l'autre de de l'apprentissage profond CNN. L'introduction de la représentation en word2vec peut donc apporter un plus dans la classification des textes arabes.

V. Conclusion :

Dans ce chapitre nous avons présenté la classification des textes tout en expliquant son processus, ensuite nous avons abordé les concepts fondamentaux de l'apprentissage profond, qui sont considérer comme étant la technologie la plus récente et la plus avancée dans le domaine de d'IA, enfin nous avons cité quelques travaux connexe pour avoir une idée sur le domaine et les travaux achevées.

Le chapitre suivant concerne la conception de notre système de classification de texte arabe basé sur les modèles de l'apprentissage profond.

Chapitre 2 : LA CONCEPTION DU MODELE DE LA CLASSIFICATION

I. Introduction

Notre travail sert à classifier des textes arabes en utilisant l'apprentissage profond et plus précisément les réseaux de neurones récurrents (RNN).

Dans ce chapitre nous présentons l'architecture de notre système et ses différents composants, qui seront détaillés. Nous présenterons aussi par la suite les modèles de RNN proposé

II. .Architecture global de notre système

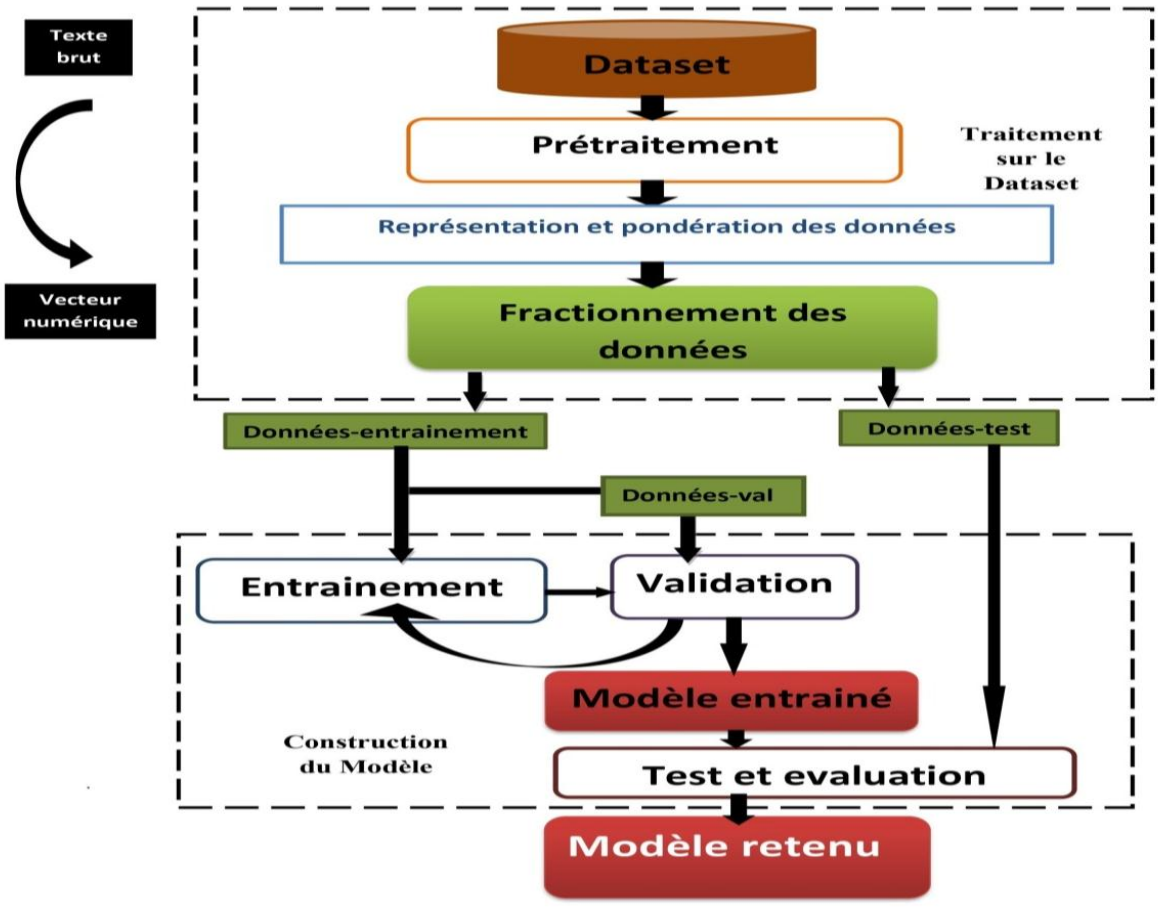

Figure 40 : architecture global du system

III. Description de l'architecture

La Figure 41représente l'architecture qui est adopté comme solution au problème posé. Elle est composée de deux parties principales .La première partie concerne le traitement sur le dataset et la deuxième porte sur construction du modèle. Chaque partie du système est revue plus en détail dans la section qui suit.

III.1 Traitement sur le Dataset

D'après l'architecture proposée, l'entrée du système est un ensemble de documents texte « dataset » en format CSV, cet ensemble sera lu comme des textes bruts pour lui appliquer les méthodes de prétraitement puis représentation et enfin fractionnement de données

III.1.1 Prétraitement des textes

Le prétraitement des textes est une phase capitale du processus de classification, puisque dans les documents textuels de nombreux mots apportent peu (voir aucune) d'informations sur le document concerné. Pour cela nous avons procédé par les étapes suivantes :

III.1.1.1 Encodage unique des textes

L'encodage unique des textes en format standard permet de représenter les textes sans aucune déformation au niveau du caractère lors de la lecture. Tous les textes de notre corpus sont représentés avec le codage (UTF-8).

III.1.1.2 Suppression des caractères inutiles

Cette étape consiste à supprimer :

- **Les signes de ponctuation**

Supprimer toute séquence de caractères de ponctuation délimitée par des lettres ou des espaces comme la virgule et le point-virgule …etc. Nous prenons en considération l'orientation de quelques caractères qui s'écrivent de droite à gauche dans les textes arabes comme le signe d'interrogation « ؟», et la virgule « ،».

- **Les nombres et les caractères latins**

Nous éliminons toutes les séquences de caractères situées entre deux espaces et contenant des chiffres e « ٩….١,1….9, I.…IX », et nous avons éliminé aussi les caractères latins « A...Z, a …z »

- **Les abréviations et les lettres isolées**

Les abréviations de mot arabe, comme « خpourتاريخ (date), « صpourصفحة (page),ج pourجواب» réponse», سpour سؤال« question».

Ou de coordination comme ك،ل(bi-, wa-, fa-, li-, ka-…) ou dans les formules mathématique comme(3=ع+س0).

III.1.1.3 Suppression des mots fréquents ou élimination des ”Mots Outils”

Les mots qui apparaissent le plus souvent dans un corpus sont généralement les mots vides (empty words) ou mots outils (stop words) : les articles, les prépositions, les mots de liaisons, les déterminants, les adverbes, les adjectifs indéfinis, les conjonctions, les pronoms et les verbes auxiliaires etc.., qui constituent une grande part des mots d'un texte, mais malheureusement sont faiblement informatifs, sur le sens d'un texte puisqu"ils sont présents sur l'ensemble des textes.

Pour l'arabe, la liste de stopwords inclut des pronomsالتي هو الذي هي (هما..) , les mots liaisons (من، مع، و، في) adverbes (..بين ،حتت، فوق), jours de semaine (السبت الاحد الاثنين...) , mois de l'année(..جانفي فيفري مارس)

Ces mots doivent être supprimés de la représentation des textes vu que :

- D'un point de vue linguistique, ces mots ne comportent que très peu d'informations. La présence ou l'absence de ces mots n'aident pas à deviner le sens d'un texte. C’est dans ce sens qu’ils sont communément appelés « mots vides».
- D'un point de vue statistique, ces mots se retrouvent sur l'ensemble des textes sans aucune discrimination et ne sont d'aucune aide pour la classification.

Leur élimination lors d'un prétraitement du document permet par la suite de gagner beaucoup de temps lors de la modélisation et l'analyse du document.

III.1.1.4 Le traitement morphologique

Cette étape est spécifique à la langue arabe. Elle consiste à effectuer un traitement au niveau de chacun des mots en fonction de leurs variations morphologiques : flexion, dérivation, composition afin de rassembler les mots de sens identiques. Donc, le but est de regrouper par exemple les termes سافرet مسافرou les termesلعبet لاعبcar ils ont la même signification.

Il s'agit de prétraitements relatifs à la Normalisation Morphologique et l'analyse morphologique de certains caractères arabes.

- **La Normalisation Morphologique**

Hamza et Alif : cette normalisation consiste à convertir el « أ ، إ » et « آ » en« ا»car la plus part des textes arabes négligent l'ajout d'Elhamzasur El Alif. Dans cette étape nous proposons aussi de supprimer totalement AlifElTanwin« ا» si elle existe, voici quelques exemples de mots avant et après la normalisation d'Elhamza(voir la Tableau 2) :

Pour « أ»	أمر	امر
Pour « إ»	إستعمال	استعمال
Pour *el tanwin*« ا»	استعمالا	استعمال
Pour el *alif avec mada*« آ»	آمر	امر

Tableau 2: La normalisation El Hamza

Yâ' et el tâmarbouta : hamza ajoute une confusion, si elle situe à la fin de mot, entre

ي(lettre yâ 'finale) etى(ou'alifmaqsûra):Le motنادي nâdî«club» ,peut être noté

نادى, (lu comme nâdâ, « convier, convoquer »).

comme dans les moteurs de recherches (google par exemple) lorsqu'on écrit dans la barre de recherches le mot أولي les premiers pages qui s'affichent contiennent le mot أولىavec alif maqsûra.

ة » relLemêmeprincipepou(tâ marbouta) » tel que la normalisation consiste à remplacer el « ة» par « ه(lettre hâ) » comme par exemple:جنه ⟶ جنة

Des signes de vocalisation : dans cette étape on supprime tous les signes de vocalisation (tableau3)

Double Constante	Aucune Voyelle	Nounation			Voyelle		
بّ /bb/	بْ /b/	بٍ /bin/	بٌ /bun/	بً /ban/	بِ /bi/	بُ /bu/	بَ /ba/

Tableau 3 : les diacritiques arabes

- **L'analyse morphologique**

Light stemming (Enracinement léger): Dépend de la suppression des lettres additionnelles du mot seulement c-à-dire : supprimer le préfixe et le suffixe du mot sans transformer le terme à la racine de l'original.

Heavy stemming (enracinement lourd):Dépend de la suppression des caractères additionnels d'un mot, puis convertir le terme à la racine de tout mot revient à son origine.

Mot	**Light stemming**
الشجاعة	شجاع
بلادي	بلاد
Mot	**Heavy stemming**
الشجاعة	شجع
بلادي	بلد

Tableau 4 : Comparaison entre les deux stemmers

Remarque : Le Light stemming conserve le sens du mot contrairement au Heavy stemming qui peut affecter la signification des mots.

Tokenisation : il s'agit de décomposer le texte en un ensemble de mots nommés jetons .

III.1.2 Représentation et pondération

Cette partie consiste à représenter les méthodes de représentation et pondération des documents textuels pour obtenir des vecteurs interprétable par la machine ce qui permet l'apprentissage de nos modèles. Les principales représentations proposées sont :

III.1.2.1 Représentation basée sur Pondération de fréquence :

La pondération de fréquence TF-IDF peut être appliquée sur plusieurs méthodes de représentation .Nous présentons les quatre méthodes proposées pour notre travail :

- **Sac de mots :** Cette représentation consiste à transformer les documents du corpus en un ensemble de mots**.**

- **N gramme** : La représentation N-gramme a plusieurs avantages comme mentionné dans le chapitre précédent, nous avons utilisé le Bi-gramme de mots pour la segmentation des textes, c'est à dire chaque segment est une séquence de deux mots.

- **Sac des stems: Cette** technique transforme les mots en leurs racines. Nous avons utilisé deux représentations du stemming léger définie comme suit :

  Tashfin stemming : c'est considéré comme une bibliothèque , elle possède une liste de préfixes et suffixes par défaut pour donner toutes les segmentations possibles. Il offre l'extraction de tiges et de racines en même temps contrairement aux autres types de stemming. (49)
  Assem stemming : est un algorithme de recherche de racine basé sur snowball stemming (la boule de neige) il applique quelques règles pour bien extraire la racine du mot.

Pour calculer les poids, nous avons appliqué la pondération TF- IDF qui consiste à donner des fréquences des mots pour chaque document, ces fréquences associées aux représentations citées plus haut seront l'entrée du modèle d'apprentissage.

III.1.2.2 Représentation basée sur la pondération de prédiction :

La prédiction concerne les représentations du (Word Embedding), nous nous intéressons à la méthode Word2Vec (avec ces deux types CBOW (continuous bag of words) et SG (skip-gram).

Afin de les utiliser, nous avons introduit un modèle pré-entrainé de l'encyclopédie Wikipédia (Word Embedding Wikipédia Arabic articles). Wikipédia est une encyclopédie qui fournit plus de 45 millions d'articles catégorisés ciblant 285 langues, dont l'arabe fait partie. La partie arabe de Wikipédia compte plus de 520 000 articles. Les articles sont segmentés en paragraphes, ce qui a donné 1 800 000 paragraphes. Chacun de ces paragraphes représentant un document à utiliser pour la construction du modèle.

Modèle	Nb documents	Nb vocabulaires	Dimension	Technique
Wikipedia-CBOW	1,800,000	140,319	300	CBOW
Wikipedia-Skip Gram	1,800,000	140,319	300	Skip-Gram

Tableau 5 : caractéristique du modèle de wékipédia

III.1.3 Fractionnement des données :

Cette partie sert à diviser l'ensemble de données en trois sous-ensembles à savoir, les données d'entrainement, les données de validation et les données de test.

III.1.3.1 Les données d'entrainement

Ces données représentent les données d'entrée de l'algorithme. Le modèle évalue ces données à plusieurs reprises pour en savoir plus sur le comportement des données, puis s'ajuste pour servir son objectif.

III.1.3.2 Les données de validation

Les données de validation sont de nouvelles données par rapport aux données précédentes (données d'entrainement), elles sont exécutées au cours de l'entrainement. Les données de validation constituent le premier test, permettant aux programmeurs d'évaluer dans quelle mesure le modèle fait des prédictions basées sur les nouvelles données.

III.1.3.3 Les données de test

Une fois le modèle est construit, le test des données confirme à nouveau qu'il peut faire des prédictions précises. Si les données d'entraînement et de validation incluent des étiquettes pour surveiller les mesures de performance du modèle, les données de test ne doivent pas être étiquetées, elles fournissent une vérification finale et réelle.

III.2 Construction du modèle

Dans cette partie, nous présentons les trois modèles proposées basées sur les réseaux de neurones récurrents, il s'agit de LSTM,BI-LSTM et GRU avec différentes méthodes de représentation.

Ces modèles partagent quelques points communs que nous allons essayer de présenter dans le premier modèle puis citer la différence seulement dans les deux autres modèles dans le but d'éviter une répétition, ces points communs sont les couches et les hyper-paramètres:

III.2.1 Le Modèle LSTM

Dans cette section, nous allons présenter notre premier modèle RNN qui est LSTM .Ce modèle a été proposé vu ses avantages par rapport aux RNN, d'un côté il permet d'éviter les problèmes de l'explosion ou dissipation du gradient et d'un autre, il permet de traiter des séquences longues. L'architecture de ce modèle est représentée selon la pondération appliquée comme suit :

III.2.1.1 Le cas du modèle LSTM avec pondération de prédiction :

L'architecture de ce modèle est illustrée dans la Figure 42, elle est décomposée en quatre couches principales qui sont « couche d'entrée », « couche embedding », « couche caché » « couche de sortie » et entre eux il y a trois couches secondaires. Le fonctionnement de chaque couche est détaillé par la suite :

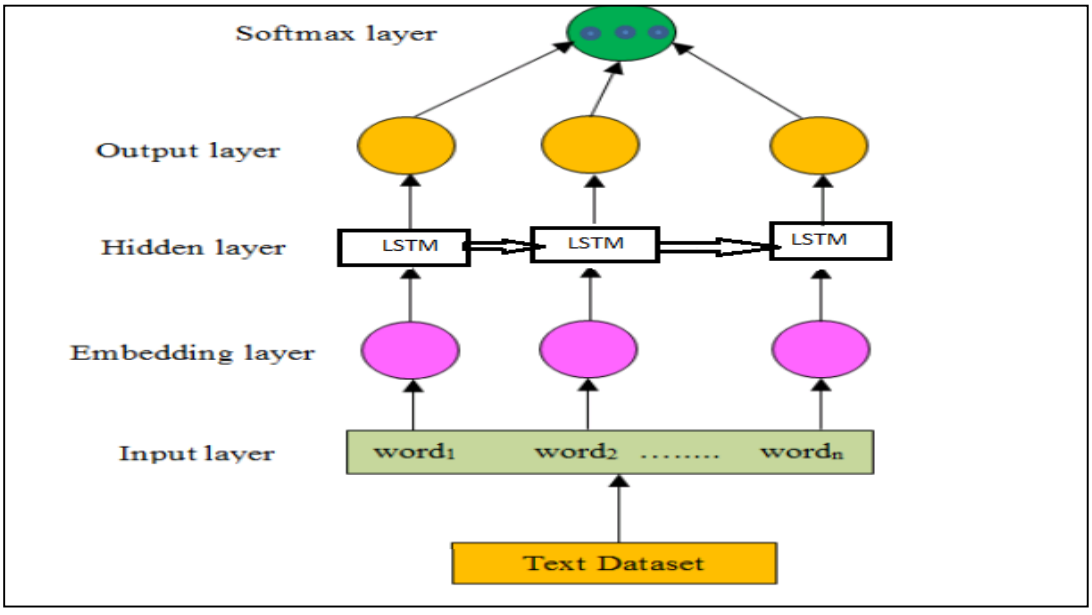

Figure 42 : architecture du ModèleLSTM avec prédiction

- **Les couches principales**

« Couche d'entrée » : c'est l'entrée de notre système, après avoir effectué tout le prétraitement nécessaire, nous obtenons un ensemble de vecteurs de mots, chacun représente un document.

« Couche embedding » : Etant donnée, les réseaux de neurones profonds ne peuv ent traiter du texte brut, nous devons convertir le texte en vecteurs numériques. Cette conversion est appelée "word Embedding", chaque mot est représenté par des nombres réels, projetés dans un espace vectoriel, cette technique nous permet de capturer le contexte d'un mot dans un document.

Parmi les techniques de Word Embedding, Nous nous intéressons au Word2Vec, avec les deux types CBOW (continuos bag of Word) et SG (skip-gram) expliqué précédemment. Les poids obtenus dans la phase de transformation des données avec word2vec sont utilisées dans cette couche pour les convertir en une matrice creuse.

« Couches LSTM » : Les réseaux de neurones LSTM font partie des réseaux de neurones récurrents qui fonctionnent sur des données séquentielles. L'entrée de chaque nœud LSTM est une sortie du nœud précédent .Dans cette couche, il y a deux types de paramètres à régler :

Les paramètres du modèle : sont plutôt appris pendant la formation du modèle.

Les Hyper paramètres : sont tous les paramètres qui peuvent être fixés arbitrairement par l'utilisateur avant de commencer l'entraînement.

Le réglage des modèles d'apprentissage automatique est un type de problème d'optimisation c'est à dire nous avons un ensemble d'hyper-paramètres et nous cherchons à trouver la bonne combinaison de leurs valeurs.(50) Les hyper paramètres du modèle LSTM sont les suivants:

a. **Fonction d'optimisation :** Nous avons utilisé l'optimiseur SGD (Descente du gradient stochastique), son objectif est de minimiser la fonction d'erreur en ajustant petit à petit les paramètres d'apprentissage représentés par les différents poids. (51)

b. **Fonction de perte (Loss) :** Une fonction de perte, ou « Loss function », est une fonction qui évalue l'écart entre les prédictions réalisées par le réseau de neurones et les valeurs réelles des observations utilisées pendant l'apprentissage. Plus le résultat de cette fonction est minimisé, plus le réseau de neurones est performant, ça se fait en ajustant les différents poids du réseau de neurones. (52) .Dans notre modèle, nous avons utilisé une fonction de perte appelée «Categorical Cross Entropy» .

Formellement, cette fonction est conçue pour quantifier la différence entre deux distributions de probabilité, la formule mathématique est la suivante:

$$loss = -\sum_{i=0}^{outputsize} y_i . logy'_i$$

ou :

y'i: est la valeur scalaire dans la sortie du modèle.

yi : est la valeur cible correspondante (la probabilité que l'événement i se produise) et la sommede tous les yi est égal à 1 .

La taille de sortie est le nombre de valeurs scalaires dans la sortie du modèle.

Le signe moins garantit que la perte diminue lorsque les distributions se rapprochent.(1)

c. **Taux d'apprentissage (Learning rate)** : Le taux d'apprentissage définit la rapidité avec laquelle un réseau met à jour ses paramètres. Un faible taux d'apprentissage ralentit le processus d'apprentissage mais converge en douceur. par contre un taux d'apprentissage plus élevé accélère l'apprentissage mais peut ne pas converger.

d. **Nombre d'époques :** Le nombre d'époques correspond au nombre de fois où l'ensemble des données d'entraînement est affiché sur le réseau pendant l'entraînement. Il faut augmenter le nombre d'époques jusqu'à l'augmentation de la précision de l'entraînement mais il ne faut pas qu'il arrive à un sur -apprentissage (Over Fitting).

e. **Taille du lot (Batch size) :** La taille du mini lot est le nombre de sous-échantillons transmis au réseau après que la mise à jour des paramètres se produit.

f. **Abandon(Drop out)** : L'abandon est une technique de régularisation qui sert à supprimer une partie des nœuds qui alimentent une couche pour éviter le sur-apprentissage et augmenter la précision de validation afin de pouvoir le généraliser . En règle générale, on peut conclure que :

- Utiliser une petite valeur de l'abandon (Drop out) pendant la phase d'entrainement (20 à 50 % des neurones) est mieux, car la valeur trop élevée entraîne un sous-apprentissage par le réseau.
- Utiliser un réseau plus large, pour obtenir probablement de meilleures performances, ce qui donne au modèle plus d'opportunités d'apprendre des représentations indépendantes.(54)

g. **Taille de la mémoire tampon (Buffer size)**: La mémoire tampon est une zone de mémoire vive de disque utilisée pour entreposer temporairement des données, notamment entre deux processus ou matériels ne fonctionnent pas à la même vitesse(55).La taille de la mémoire tampondésigne le nombre de documents d'entrainement.

h. **Taille d'encastrement (Embedding size)**: c'est la taille de l'espace vectoriel c'est à dire le nombre de descripteurs numériques utilisés pour décrire les mots (entre 100 et 1000 en général). (56)

i. **Nombre de couches LSTM**: il n y a pas de règle exacte pour choisir le nombre de couches à utiliser, mais il existe des règles empiriques, parmi celles-ci, la plus couramment invoquée est « la taille optimale de la couche cachée se situe généralement entre la taille de l'entrée et la taille des couches de sortie

j. **Nombre de nœuds LSTM** : est le nombre de cellules utilisées dans une couche cachée.

Remarque : Pour minimiser l'erreur et avoir un réseau qui se généralise bien, il faut choisir un nombre optimal de couches cachées, ainsi que des nœuds dans chaque couche cachée.

- Peu de nœuds entraîneront une erreur élevée pour un système car les facteurs prédictifs peuvent être trop complexes pour être capturés par un petit nombre de nœuds.
- Trop de nœuds s'adapteront aux données d'entraînement et ne se généraliseront pas bien.

Les valeurs des hyper paramètres sont présentés dans le tableau suivant :

Hyper paramètre	Définition
Fonction d'optimisation	Fonction SGD
Fonction Loss	Fonction categorical cross Entropy
Nombre de couches LSTM	2
Nombre de nœuds LSTM	32 / 16
Learning rate	0.01
Batch size	1000
Embedding size	300
Buffer size	67036
Drop Out	0.1

Tableau 6 : Les hyper-paramètres du modèle LSTM dans le cas de de prédiction

« couche sortie » : Les résultats de la couche précédente passent par cette dernière couche, elle agit comme un classifieur où elle utilise une fonction d'activation "soft max" qui donne une probabilité en sortie de chaque neurone, le neurone de sortie avec la probabilité la plus grande permettant alors de décider que sa classe associée est la classe prédite .

- **Les couches secondaires**

« Couche flatten » : Elle est considérée comme une opération d'aplatissement sur le tenseur du LSTM qui est en 3D pour avoir la forme 2D qui est égale au nombre d'éléments contenus dans le tenseur de la couche suivante (dense).

«Couche dense »: La plus simple couche, on parle de couche « fullyconnected (FC) » généralement utilisé pour faire la connexion entre les couches LSTM et la couche de sortie.

III.2.1.2 Le cas du modèle LSTM avec la pondération TF-IDF :

Dans ce modèle, la fréquence TF-IDF est appliquée pour chaque technique de représentation la pondération TF- IDF qui sert à donner des fréquences, ces fréquences sont utilisées pour la couche d'entrée du LSTM. L'architecture de ce modèle suivi de tableau des hyper-paramètres sont illustrées ci-dessous Figure 43ou la couche " embedding " n'existe pas.

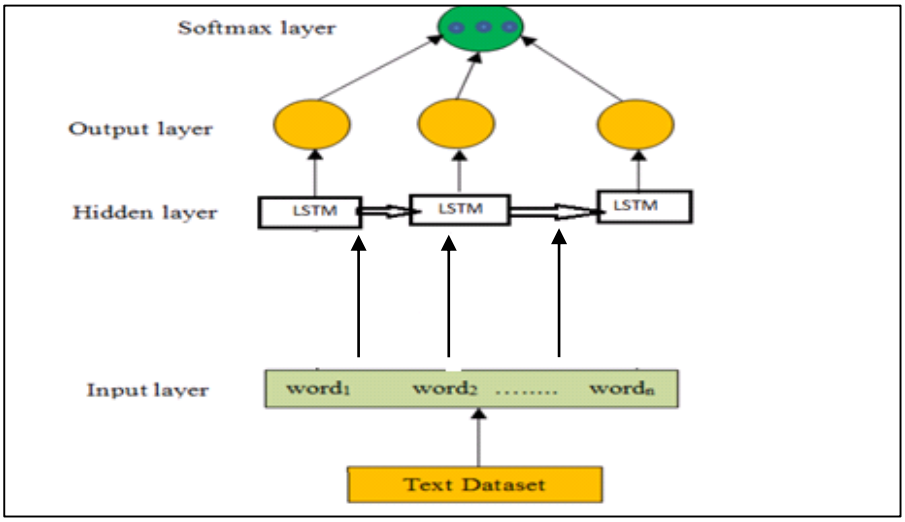

Figure 43 : architecture du modèle LSTM avec TF-IDF

Hyper paramètre	Définition
Fonction d'optimisation	Fonction SGD
Fonction Loss	Fonction categorical cross Entropy
Nombre de couches LSTM	2
Nombre de noeuds LSTM	32 / 16
Learning rate	0.01
Batch size	1000
Buffer size	67036
Drop Out	0.1

Tableau 7 : modèle LSTM dans le cas fréquence TFIDF

III.2.2 Le Modèle BILSTM

Ce modèle améliore le modèle LSTM en considérant tous les mots dans les deux sens pour bien prédire le mot , comme illustré dans la Figure 43 la décomposition de ce modèle est pareil à celle du LSTM sauf que le fonctionnement des couches cachées « hidden layer » est différent .

« Couches cachées BILSTM »

Ils se basent sur deux sous couches de LSTM, une couche s'exécutant en avance « Forward » (l'entrée de chaque nœuds LSTM est la sortie nœud traitant le mot précédent) et une autre s'exécutant en arrière « Backward » (l'entrée du chaque nœud LSTM est la sortie du nœud traitant le mot suivant). Les deux états cachés des deux couches combinées sont capables de préserver les informations du passé et du futur.

Le modèle BI-LSTM utilise les deux pondérations comme le modèle LSTM, la Figure 44 et le Tableau 8 illustre le cas d'une pondération de prédiction, la Figure 45 et le Tableau9 présente le cas de la pondération en fréquence TF-IDF.

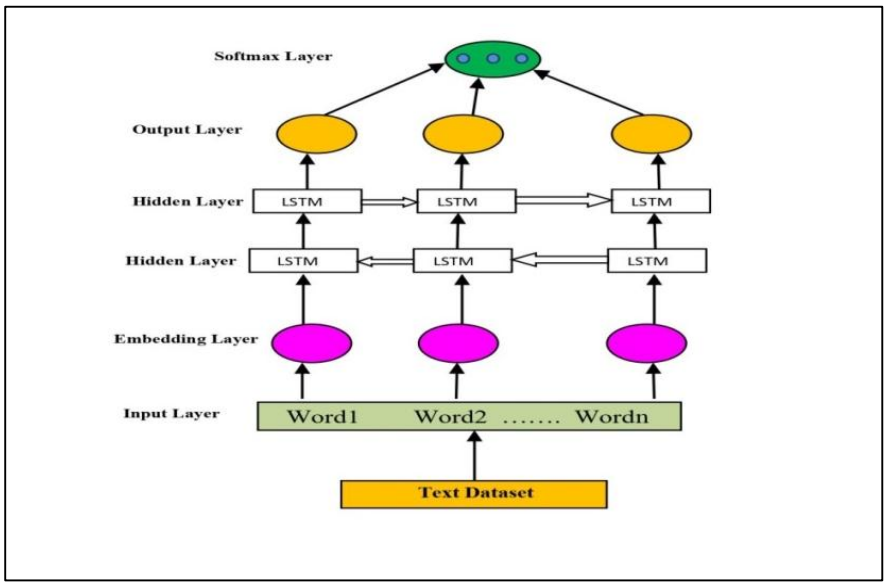

Figure 44 : architecture du modèle BI-LSTM avec pondération de prédiction

Hyper paramètre	**Définition**
Fonction d'optimisation	Fonction SGD
Fonction Loss	Fonction categoricalcrossEntropy
Nombre de couches LSTM	2
Nombre de nœuds LSTM	32 / 16
Learning rate	0.01
Batch size	1000
Embedding size	300
Buffer size	67036
Drop Out	0.1

Tableau 8 : Les hypers-paramètres du modèle BI-LSTM avec pondération de prédiction

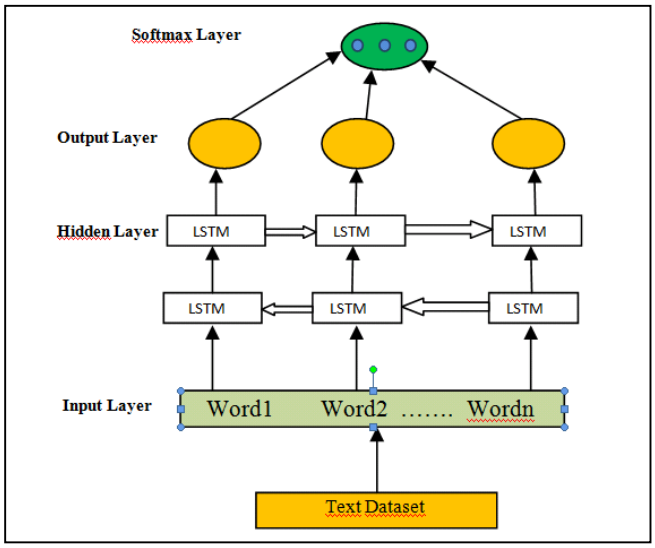

Figure 45 : l'architecture du modèle BI-LSTM avec pondération TF-IDF

Hyper paramètre	**Définition**
Fonction d'optimisation	Fonction SGD
Fonction Loss	Fonction categoricalcrossEntropy
Nombre de couches BI LSTM	2
Nombre de noeuds BI LSTM	32,16
Learning rate	0.01
Batch size	1000
Buffer size	89382
Drop Out	0.1

Tableau 9 : Les hyper-paramètres du modèle BI-LSTM avec TF-IDF

III.2.3 Le modèle GRU

Le modèle GRU est similaire au modèle LSTM sauf que dans la structure de cellule GRU comporte deux portes (cités dans le chapitre 1), l'utilité de cette structure est d'exposer son contenue de mémoire à chaque étape et équilibre la sortie entre l'état de mémoire précédente et le nouvel état de mémoire candidat. Le modèle GRU utilise les deux pondérations comme le modèle LSTM, la Figure 46 et le Tableau 10 illustre le cas d'une pondération de prédiction, la Figure47 etle Tableau 11 présente le cas de la pondération en fréquence.

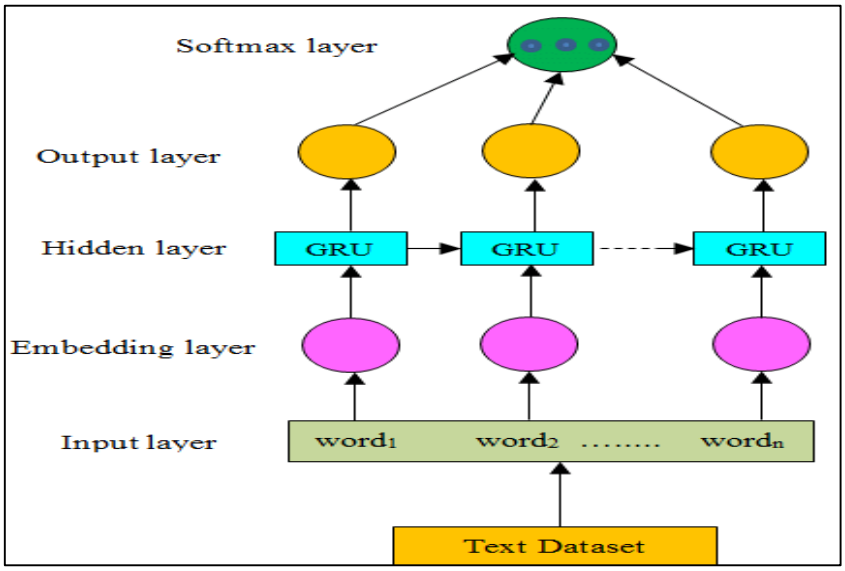

Figure 46 : architecture du modèle GRU avec prédiction

Hyper paramètre	Définition
Fonction d'optimisation	Fonction SGD
Fonction Loss	Fonction categoricalcrossEntropy
Nombre de couches GRU	2
Nombre de nœuds GRU	32 / 16
Learning rate	0.01
Batch size	1000
Embedding size	300
Buffer size	67036
Drop Out	0.1

Tableau 10 : Leshyper-paramètres du modèle GRU cas prédiction

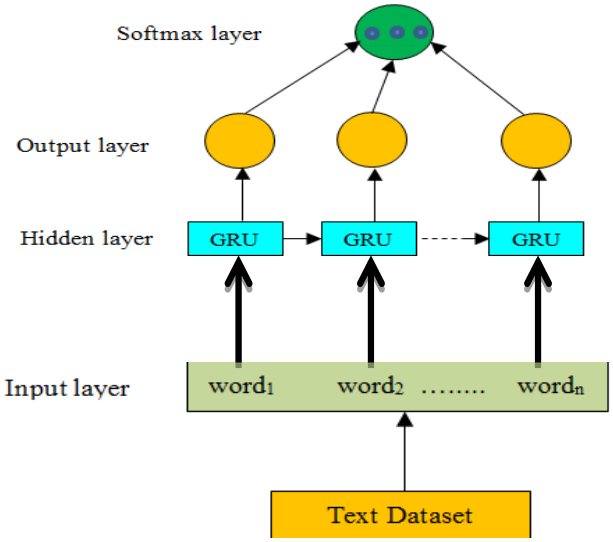

Figure 47 : architecture du modèle GRU avec TFIDF

Hyper parameter	Définition
Fonction d'optimisation	Fonction SGD
Fonction Loss	Fonction categorical cross Entropy
Nombre de couches GRU	2
Nombre de noeuds GRU	32 / 16
Learning rate	0.01
Batch size	1000
Buffer size	89382
Drop Out	0.1

Tableau 11 : Les hyper-paramètres du modèle GRU dans le cas pondération tfidf

IV. Conclusion

Dans ce chapitre nous avons présenté l'architecture de notre système avec ses composantes. Nous avons présenté aussi, les trois modèles basés sur les réseaux de neurones récurrents avec les différentes méthodes de représentations. Dans le chapitre suivant, nous allons entamer les résultats et l'évaluation après avoir implémenté les modèles proposés

CHAPITRE 03 : TESTS ET RESULTATS

I. . Introduction

Les phases de conception et d'implémentation d'un système doivent être suivies par une phase d'évaluation des résultats à l'aide des outils et des métriques.

Dans ce chapitre, nous commençons par présenter le dataset, les ressources matérielles et logiciels utilisées, ensuite nous décrivons les résultats des tests obtenus et nous conclurons par l'interprétation de ces résultats.

II. Description du dataset

Pour tester nos modèles proposés, nous avons utilisé le dataset de biniz (60) qui représente une collection de textes arabes utilisée dans les articles de journaux. Le texte contient des mots alphabétiques, numériques et symboliques. Cet ensemble de données se compose de 319 254 124 mots (Tableau 12) et de 111 728 documents (Tableau 13) structurés en fichiers texte et collectés auprès de 3 journaux arabes en ligne : Assabah (www.assabah.ma), Hespress (www.hespress. com) et Akhbarona (www.akhbarona.com) en utilisant un processus d'exploration Web semi-automatique.

Les documents de l'ensemble de données sont classés en 5 classes : sport, politique, culture, économie et divers. Le nombre de documents et de mots pour chaque classe varie d'une classe à l'autre (Tableau 12-Tableau 13).

Nous avons divisé le dataset en trois sections comme illustré dans le (Tableau 14): l'ensemble de données d'entraînement représente 60% de chaque classe, l'ensemble de données de validation représente 20 % et l'ensemble des données de test représente les 20 %. Les enregistrements de l'ensemble de données ont été stockés sous forme de matrices, où chaque mot est présenté sous forme de vecteur

Site	**Sport**	**Politique**	**Culture**	**Economie**	**Diverse**	**Total**
Assabah	90 182 142	10 693 166	19 795 393	9 309 981	20 402 016	150 382 699
Hespress	17 728 154	22 771 705	11.423.641	13 567 983	12 078 694	77 570 177
Akhbarona	9 465 337	33 293 023	10 997 014	19 543 260	18 002 614	91 301 248
TOTAL	**117 375 633**	**66 757 894**	**42 216 048**	**42 421 224**	**50 483 325**	**319 254 124**

Tableau 12: nombre de mots dans le dataset

Site	Sport	Politique	Culture	Economie	Diverse	Total
Assabah	34 244	2 381	5 635	2 620	9 253	**54 133**
Hespress	6 965	5 737	3 023	3 795	7 475	**26 995**
Akhbarona	5 313	12 387	5 080	7 820	NAN	**30,6**
TOTAL	**46 522**	**20 505**	**13 738**	**14 235**	**16 728**	**111 728**

Tableau 13: nombre de documents dans l'ensemble de données

Fractionnement	Entrainement (60 %)	Validation (20%)	Test (20 %)	Total
Nombre de documents	67036	22345	22345	111728

Tableau 14: Fractionnement du dataset

III. Ressources matérielles logicielles utilisées

La réalisation de notre système nécessite les ressources suivantes :

III.1 Matériel

Le matériel joue un rôle très important dans toutes les activités de recherche en particulier quand il s'agit de grandes quantités de données.

Dans notre travail, nous avons utilisé deux machine doté de Intel(R) Core (TM) i3-6006U CPU @ 2.00GHz avec 4GB de RAM et 500 GB de disque.

III.2 Logiciel

Afin de développer nos modèles de l'apprentissage profond, nous avons travaillé sur la plate-forme Google Colaboratory ou colab fournie par Google Company .

C'est un outil cloud simple et pour initier au Deep Learning ou collaborer des projets avec les collègues en science des données. Colab permet :

- d'améliorer les compétences de codage en langage de programmation Python.
- de développer des applications en Deep Learning en utilisant des bibliothèques Python populaires telles que Keras, TensorFlow, PyTorch et OpenCV.

- d’utiliser un environnement de développement (Jupyter Notebook) qui ne nécessite aucune configuration.

Mais la fonctionnalité qui distingue Colab des autres services est l’accès à un processeur graphique GPU, totalement gratuitement. De plus, les documents Colab (Jupyter Notebook) sont enregistrés directement votre compte Google Drive.

Nous avons commencé par utiliser la version gratuite de googlecolab mais avec un stockage de 12GB de RAM et 73GB de disque et un temps maximum d’exécution de 12 h par jour il nous a été difficile de développer nos modèles ce qui nous a amené à acheter le version googlecolab professionnel pour accélérer l’exécution de nos modèles de l’apprentissage profond. Un compte colab professionnel possède (25GB de RAM GPU) et (35GB de RAM TPU)avec temps d’exécution 24 h. Le disque peut avoir jusqu’à 107GB.

Nous avons travaillé avec le langage de programmation python qui présente plusieurs avantages tel que :

- la syntaxe est très simple.
- C'est un langage puissant dans de nombreux domaines.
- Il dispose de bibliothèques diverses et riches.

III.3 Librairies:

Nous citons ci-dessous librairies principales utilisées dans la construction de nos modèles :

III.3.1 Tenser flow

C’est une bibliothèque de logiciels open source pour le calcul numérique de haute performance. Développé à l'origine par des chercheurs et des ingénieurs de l'équipe Google Brain au sein de l'organisation IA de Google, Son architecture flexible permet un déploiement facile du calcul sur une variété de plateformes (CPUs, GPUs, TPUs), et des ordinateurs de bureau aux clusters de serveurs en passant par les périphériques mobiles. Il est livré avec un support solide pour l'apprentissage automatique et l'apprentissage en profondeur et le noyau de calcul numérique flexible est utilisé dans de nombreux autres domaines scientifiques.

III.3.2 Keras

Keras est une «API» de réseaux de neurones minimaliste et modulaire. Elle utilise Theano ou Tensor Flow comme back-end.

Elle a été développée dans le but de permettre une expérimentation rapide, prend en charge à la fois les réseaux basés sur la convolution et les réseaux récurrents (ainsi que les combinaisons des deux) et permet

de construire des réseaux à base de séquences et de graphes, il possède des algorithmes pour les couches d'optimisation, de normalisation et d'activation. keras fonctionne de manière transparente sur les périphériques «CPU» et «GPU ».

Grâce sa simplicité, Keras permet de mettre en place des projets rapidement, cependant une des limites est qu'il ne prend pas en charge les environnements multi-GPU.

III.3.3 Nltk

Natural Language Toolkit (NLTK) est une bibliothèque logicielle en Python permettant un traitement automatique des langues, développée par Steven Bird et Edward Loper du département d'informatique de l'Université de Pennsylvanie. En plus de la bibliothèque, NLTK fournit des démonstrations graphiques, des données-échantillon, des tutoriels, ainsi que la documentation de l'interface de programmation (API).Nltk offre un ensemble de fonctions qui aide au prétraitement des textes de différentes langues.

III.3.4 Scikit-learn

Connu par sklearn , c'est la principale bibliothèque d'outils d'intelligence artificielle (IA) dédiés à la machine learning et qui permet de concrétiser des projets en science. Ceframework comprend une bibliothèque libre initiale, avec la possibilité d'intégrer d'autres bibliothèques libres. Les développeurs et data scientistes peuvent s'en servir pour les usages suivants

- Générer des machines à vecteur de support.
- Utiliser des algorithmes de classification.
- Effectuer des opérations de régression logistique.
- Identifier et exploiter des forêts d'arbres décisionnels de type aléatoire.

Sklearn permet également de s'inscrire dans une optique de « feature engineering » ou de « clustering ». Il est essentiellement écrit en langage de programmation Python avec des occurrences en C, C++ et Cython.

III.3.5 Numpy

Est un module complémentaire destiné à offrir à Python des outils de calculs scientifiques avancés. Le package comprend, entre autres, un objet de calcul de tableau à n dimensions, des fonctions d'algèbre linéaire basique, des transformations de Fourier, des outils avancés de génération de chiffres aléatoires, etc.

Par ailleurs, ce module propose des outils destinés à intégrer du code Fortran et C/C++. Par-delà ses fonctions purement scientifiques, Numpy peut aussi être utilisé comme container de données multidimensionnel, un facteur d'intégration rapide pour une large variété de base de données.

III.3.6 Pandas

Est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles.

Cette librairie est principalement utilisée pour manipuler les fichiers de format CSV.

- **CSV (Comma Seperated Values) :** csv est un format ouvert très populaire, il a comme fonction l'importation et l'exportation des données tel que les feuilles de calcul, les données d'un fichier texte ou Excel. Ce format est sous la forme textuelle séparées par des virgules d'où le nom Comma Seperated Value.

III.3.7 Matplotlib

C'est une bibliothèque Python open source permettant de créer des visualisations de données, initialement développée par le neurobiologiste John Hunter en 2002pour visualisé les signaux électriques du cerveau de personnes épileptiques. Cette bibliothèque est particulièrement utile pour les personnes travaillant avec Python ou NumPy, il est aussi possible pour les développeurs d'intégrer des graphiques.

Pyplot est un module Matplotlib proposant plusieurs fonctions simples pour ajouter des éléments tels que des lignes, des images ou des textes aux axes d'un graphique. Son interface est très confortable, et c'est pourquoi ce module est très utilisé.

III.3.8 Gensim

Gensim est une bibliothèque logicielle Python de « topic modelling », présenté comme un progiciel de traitement du langage naturel qui fait de la « modélisation de sujets pour les humains ». Mais c'est pratiquement bien plus que cela. Il s'agit d'un progiciel de pointe pour le traitement de textes, l'utilisation de modèles vectoriels de mots (tels que Word2Vec, Fast Text, etc.) et la création de modèles thématiques.

III.4 Processus d'exécution

Notre système basé sur les réseaux de neurones récurrents RNN commence par lire le dataset par la bibliothèque Pandas, la Figure 48 représente un exemple du texte à l'état brut. Le texte subit un

ensemble de transformations dans l'étape de prétraitement comme illustré dans les figures (Figure 49, Figure 50, Figure 51, Figure 52)

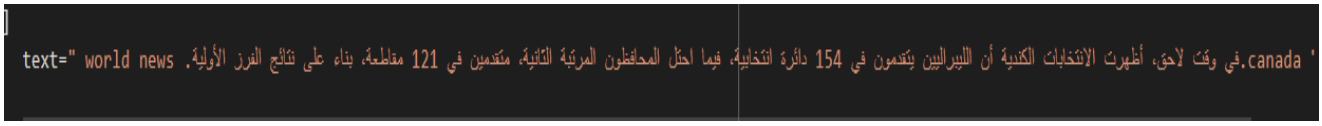

Figure 48 : Exemple du texte a l'état brut

Figure 49 : suppression des ponctuations

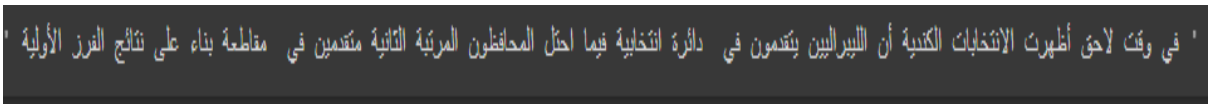

Figure 50 : suppression des nombres et des lettres latins

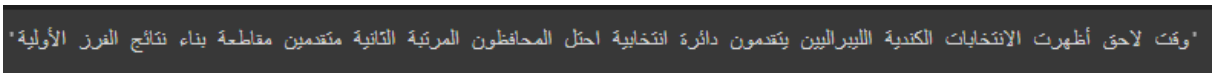

Figure 51 : suppression des mots outils

Figure 52 : la normalisation du texte

L'étape suivante est la création des vecteurs, dans lequel nous commençons par la tokenisation , puis la transformation des mots en vecteurs numériques , comme montré par les figures (Figure53, Figure 54)

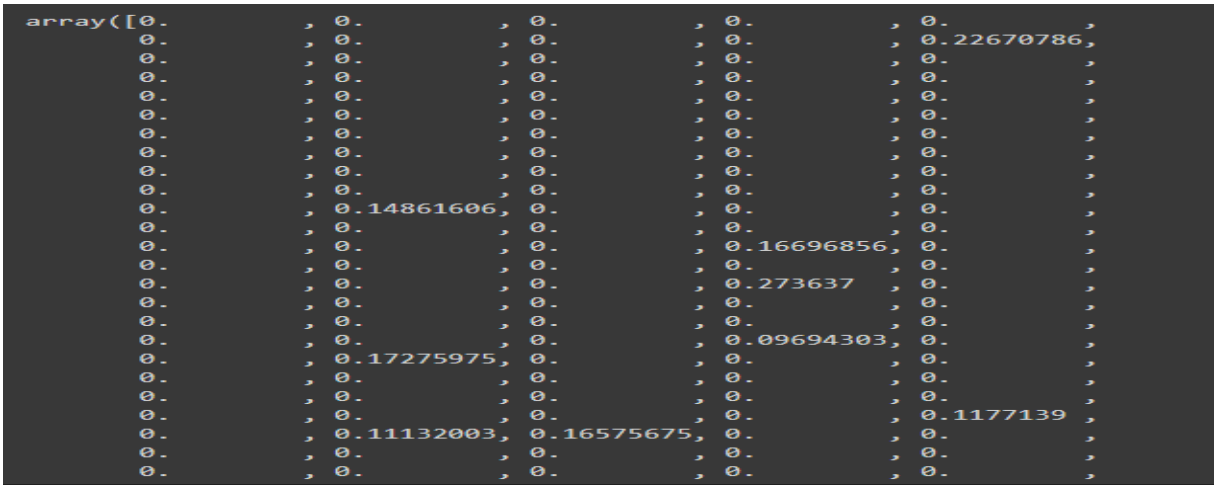

Figure 53 : Vecteurs numérique du texte dans les cas de pondération TF-IDF

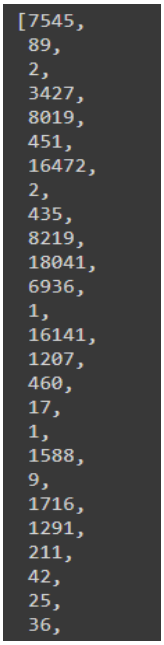

Figure 54 : Vecteur numérique du texte dans les cas de pondération de prédiction

Puis nous passons à l'étape d'entrainement du modèle pour aboutir à un modèle entrainé pouvant catégoriser de nouveaux textes en entrée.

4	politic	politic	True	ظر يتطرق رعيس مح...
5	sport	sport	True	اسين ايوب توصل ق...
6	sport	sport	True	خطاء تكرر استوعب...
7	politic	politic	True	م واضاف زيار ستت...
8	culture	culture	True	در مكان صارخ واض...
9	politic	politic	True	قليمي وامناء محل...
10	events(divers)	events(divers)	True	رصد بامر تلبس حق...
11	sport	sport	True	ريق ملاءم تنقل ك...

Figure 55: Exemple d'un document dans la bonne catégorie

IV. Tests et résultats

Nous donnons dans ce qui suit les résultats des tests réalisés sur notre dataset (60) avec les différents modèles proposés et avec différentes méthodes de représentation de texte.

Pour évaluer les performances des modèles d'entraînement afin de vérifier s'ils classent les documents dans la bonne catégorie (Figure 55), nous utilisons les mesures précision, rappel, F1 score, accuracy (exactitude) citées au chapitre 1.

Nous avons commencé par appliquer l'algorithme LSTM avec les représentations en sac de mot, en bi-gramme et en stemming léger (tashphyne et assem) qui sont utilisées toutes avec la pondération TF-IDF. Le Tableau 15 montre les résultats obtenus :

Métrique / Représentation	Précision	Rappel	F1_score	Accuracy(%)	Temps d'exécution
Sac de mot	0.91	0.90	0.90	90.50	15h40min
Ngramme	0.81	0.82	0.81	81.58	18h30min
Stemming léger (assem)	0.91	0.91	0.91	90.88	22h9min
Stemming léger (tashaphyne)	0.91	0.91	0.91	**91.01**	23h6min

Tableau 15 : Résultats obtenus pour le modèle LSTM les méthodes en utilisant la pondération TF/IDF

Les résultats montrent que l'algorithme LSTM a atteint une exactitude (accuracy) de 91.01 % en utilisant la méthode stem Tashphyne donc plus importante que les trois premières méthodes de représentation mais avec un temps beaucoup plus élevé (23h 6 min).

Après nous avons utilisé la méthode Tashphyne avec lesautres modèles proposées (BILSTM et GRU) les résultats obtenus sont illustrés dans Tableau 16.

Métrique / Représentation	Précision	Rappel	F1_score	Accuracy	Temps d'exécution
BILSTM	0.92	0.92	0.92	91.78	36h20min
GRU	0.92	0.92	0.92	**91.79**	26h7min
LSTM	0.91	0.91	0.91	91.01	23h6min

Tableau 16 : Les résultats obtenus pour la méthode Tashphyne avec les modèles RNN

Ce tableau montre que le modèle GRU a eu la plus grande valeur accuracy (91.79%) .

Ceci nous a permis de choisir le modèle GRU pour le comparer avec le résultat des travaux de Biniz (36) utilisant les algorithmes d'apprentissage automatiques LR et SVM ainsi que le modèle d'apprentissage profond CNN appliqué sur le même dataset avec la représentation stemming tashphyne .

Modèle	Accuracy (%)
LR	86,31
SVM	88,20
CNN	92.94
RNN (GRU)	91.79

Tableau 17 : Les résultats obtenus CNN et GRU utilisant le dataset de biniz

Les résultats montrent que notre modèle GRU est meilleur en terme d'accuracy par rapport aux algorithmes LR et SVM mais reste un peu en dessous des résultats obtenus par le modèle CNN.

Dans l'étape suivante, nous avons appliqué le modèle LSTM avec la méthode word2vec (cbow , skip gram) et word2vec pré-entrainé . Les résultats sont illustrés dans le **Tableau 18.**

Métrique / **Représentation**	Précision	Rappel	F1_score	Accuracy	Temps d'exécution
Word2vec Pré-entrainé (cbow)	0.94	0.94	0.94	**94.12**	4h8 min
Word2vec Pré-entrainé (skip gram)	0.93	0.93	0.93	92.67	5h6min
Word2vec (cbow)	0.93	0.93	0.93	93.46	5h9min
Word2vec (skip gram)	0.92	0.92	0.92	91.66	18h7min

Tableau 18 : Résultats obtenus pour le modèle LSTM avec les méthodes basé sur la prédiction (wor2vec)

Nous observons que les résultats les plus élevés (94.12 %) sont obtenus lors de l'utilisation de la méthode word2vec pré-entrainé (cbow avec un temps d'exécution plus court (4h8min).

Ensuite nous avons utilisé word2vec pré-entrainé (cbow) avec les modèles RNN proposées, le Tableau 19 montre les résultats obtenus

Métrique / Modèle	Précision	Rappel	F1_score	Accuracy	Temps d'exécution
BILSTM	0.94	0.94	0.94	**94.16**	10h19min
GRU	0.94	0.94	0.94	94.07	5h9min
LSTM	0.94	0.94	0.94	94.12	4h8min

Tableau 19 : Les résultats obtenus pour la méthode word2vec pré-entrainé -cbow avec les différents modèles de l'apprentissage profond

Nous remarquons que le modèle BILSTM a la plus grande valeur accuracy (**94.16**%) mais avec un temps élevé (10h19min) , par contre le modèle GRU a la plus petite valeur accuracy (94.07 %) avec un temps plus court (5h 9 min) ceci s'explique par le fait que le modèle BILSTM traite les données dans les deux sens (prend du temps).

Dans la phase de l'implémentation du modèle, nous avons utilisé deux couches LSTM, la première contient 32 nœuds et la deuxième contient 16 nœuds. Le modèle BILSTM passe par la première couche en traitant tous les nœuds puis il revient pour traiter le cas inverse afin de bien prédire la classe. Pour le cas de 32 nœuds, ça donne 64 comme illustré dans la **Figure 56**, et c'est la raison que le temps d'exécution augmente en double par rapport à LSTM et GRU qui passe par les nœuds d'une couche dans un seul sens.

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 255)]             0
_________________________________________________________________
embedding (Embedding)        (None, 255, 300)          92287500
_________________________________________________________________
bidirectional (Bidirectional (None, 255, 64)           85248
_________________________________________________________________
dropout (Dropout)            (None, 255, 64)           0
_________________________________________________________________
bidirectional_1 (Bidirection (None, 255, 32)           10368
_________________________________________________________________
dropout_1 (Dropout)          (None, 255, 32)           0
_________________________________________________________________
flatten (Flatten)            (None, 8160)              0
_________________________________________________________________
dense (Dense)                (None, 200)               1632200
_________________________________________________________________
dropout_2 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 1005
=================================================================
```

Figure 56 : Résumé du modèle BILSTM

En dernier et pour tester l'efficacité de notre modèle pour la classification des textes, nous avons utilisé un autre dataset (OSAC) (66) utilisé dans le domaine (57) .Le dataset OSAC a été utilisé dans beaucoup de travaux. C'est est un ensemble de données arabe qui comprend 22 429 documents texte avec 10 catégories (Économie, Histoire, divertissements, éducation et famille.....). Ce corpus contient environ 18 183 511 (18 M) mots et 449 600 mots-clés de district (après suppression des mots vides), provenant de plusieurs sites Web (Tableau 20).

Tableau 20 : Les sources des catégories

Categorie	**Nombre De documents**	**Sources**
Economie	3102	bbcarabic.com - cnnarabic.com - aljazeera.net - khaleej.com - banquecentrale.gov.sy
Histoire	3233	www.hukam.net- moqatel.com- altareekh.com- islamichistory.net
Education et famille	3608	- saaid.netnaseh.net - almurabbi.com
Religions et Fatwas	3171	*CCA* corpus - *EASC* corpus moqatel.com - islamic-fatwa.com -saaid.net
Sport	2419	bbcarabic.com - cnnarabic.com - khaleej.com
Santé	2296	dr-ashraf.com - *CCA* corpus - *EASC* corpus - W corpus - kids.jo - arabaltmed.com
Astronomie	557	arabastronomy.com-- alkawn.net bawabatalfalak.com- nabulsi.com - www.alkoon.alnomrosi.net -
Loie	944	lawoflibya.com qnoun.com
Histoire pour enfant	726	*CCA*corpus -kids.jo - saaid.net
Cuisine	2373	aklaat.com - fatafeat.com
TOTAL	**22,429**	

Nous avons appliqué notre modèle sur le dataset OSAC (57).Les résultats obtenus sont présentés dans le Tableau 21 suivant :

Métrique / Datasets	Précision	Rappel	F1_score	Accuracy	Temps d'exécution
Notre dataset	0.94	0.94	0.94	**94.16**	10h19min
Dataset OSAC	0.89	0.88	0.88	88.52	2h50min

Tableau 21 : Les résultats deux datasets en utilisant le modèle BILSTM et word2vec (cbow)

Les résultats montrent que notre modèle a donné de bons résultats sur le dataset OSAC mais de meilleurs sur notre dataset. Ceci s'explique d'une part par le fait que la taille de l'ensemble de données de notre dataset est plus grande que celle du dataset OSAC ce qui rend l'apprentissage plus performant et d'autre part , les catégories sont différentes .

V. Conclusion :

Suite à cet ensemble de tests et de résultats, nous arrivons à plusieurs constations que nous développons dans les points suivants :

- La méthode de représentation de textes Word2vec (cbow) dépasse toutes les autres méthodes de représentations pour la tache de classification.

- Les modèles RNN améliorent la précision par rapport aux algorithmes d'apprentissage automatique classiques.

- Le modèle BILSTM donne des résultats satisfaisant en terme d'exactitude (accuracy)

- Plus la taille de l'ensemble de données utilisé est grande plus l'exactitude obtenue est meilleure.

Conclusion générale

Nous nous sommes intéressés dans ce travail à la classification des textes arabes en utilisant l'apprentissage profond dans le but d'améliorer la précision.

Nous avons commencé par présenter l'état de l'art sur la classification et l'apprentissage profond en plus nous avons effectué une synthèse des travaux connexes. D'après la synthèse effectuée, nous avons constaté qu'il y avait peu de travaux qui traitent le texte arabe et dans la majorité ce sont des travaux qui utilisent les réseaux de neurone convolutionnel . C'est pour cela que nous avons choisi d'explorer des modèles basés sur les réseaux de neurones récurrent RNN pour la classification des textes en langue arabe.

A partir de cette recherche bibliographique, nous avons exploré trois modèles RNN . Il s'agit de LSTM ,BILSTM ,GRU . Ces modèles sont appliqués avec différentes méthodes de représentations de données textuelles à savoir des représentations basé sur la fréquence TF-IDF en considérant le mot isolé qui sont " sac de mots" , "BI- gramme " ," le stemming léger " et d'autres représentations qui considère le mot avec ses voisins , nous avons utilisé Word2vec avec ses deux types CBOW et SKIP -GRAM .

Les modèles développés sont appliqués sur un large corpus de textes arabes , les résultats obtenus à la fin sont interprétés et évalués selon des métriques Précision , Rappel , F1-score et exactitude (Accuracy) . Les résultats obtenus ont montré que le modèle BILSTM avec la représentation word2vec a donné meilleur résultat en termes de précision.

Malgré les résultats très prometteurs, quelques limites ont été soulevées à savoir :

- L'utilisation d'un grand dataset avec Google colab prend trop de temps pour l'entrainement surtout si on utilise avec des machines utilisé pas très performantes.
- Le datastet arabe d'entrainement n'est pas riche en contenu, il ne traite pas beaucoup de domaines de plus il comporte trop d'erreurs grammaticales et syntaxiques. Le manque de ressources en langue arabe persiste encore comme un défi qu'il faut sérieusement considérer.

Afin de surmonter ces limitations et d'améliorer la qualité des modèles pour la classification des textes arabes, nous proposerons en perspective d'explorer de nouvelles techniques de l'apprentissage profond comme celle basée sur le mécanisme d'attention et d'utiliser des datasets avec une large couverture ce qui permet d'avoir un meilleur entrainement et une meilleure catégorisation.

VI. Bibliographie :

1. https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/categorical-crossentropy. [En ligne] 22/08/2021

2. R.JALAM, « Apprentissage automatique et catégorisation de textes multilingues », Thèse de doctorat, Université Lumière Lyon 2, France, Juin 2003.

3. SEBASTIANI, FABRIZIO.*Machine Learning in Automated Text Catégorisation.* Italie : Conseil recherché National, Mars 2002.

4. Ignat, Camelia.*Représentation de textes a l'aide d'étiquettes sémantiques dans le cadre de la classification automatique.* Strasbourg, France : European Commission, IPSC, 2007.

5. Jalam, Chauchat.*Pourquoi les n-grammes permettent de classer des textes Recherche de mots-clefs pertinents à l'aide desn-grammes caractéristiques.* MaloFrance : 6èmes Journées internationales d'Analyse statistique des Données Textuelles, St, 2002.

6. C.de Loupy, M.El-Bèze.*Using few cues can compensate the small amount of resources available for WSD.*

7. Hocine, MATALLAH.*classification automatique de textes Orienté Agent ».Mémoire de fin d'étude Magister.* Algérie : UNIVERSITE ABOUBEKR BELKAID-TLEMCEN. DEPARTEMENT D'INFORMATIQUE, 2011.

8. S.Scott, S.Matwin.*Feature Engineering for Text Classification.* 1999.

9. RÉHEL, Simon.*Catégorisation automatique de textes et Cooccurrence de mots provenant de documents non étiquetés.* Canada : Mémoire, Université Laval Québec, Janvier 2005.

10. J.Clech.*Contribution méthodologique à la fouille de données complexes.*

11. Sara, Gherabi.*CLASSIFICATION AUTOMATIQUE DES TEXTES ARABE (ARABIC OPINION POLARITY).* Msila : Mémoire de fin d'étude de master,université de msila, département de l'informatique, 2014.

12. Hocine, Mataalah.*classification automatique de textes Orienté Agent .* algerie : faculté des sciences, 2010-2011.

13. https://dataanalyticspost.com/Lexique/word-embedding/ .[En ligne] 15/08/2021 . [

14. G.Salton, C.Buckley.*Term-weighting approaches in automatic text retrieval.*

15. http://www.jybaudot.fr/Stats/contingence.html.[En ligne] 21/09/2021

16. https://deepai.org/machine-learning-glossary-and-terms/f-score.[En ligne] 10/09/2021

17. https://deepai.org/machine-learning-glossary-and-terms/f-score. [En ligne] 22/09/2021

18. Ouchiha, Lahlou.*CLASSIFICATION SUPERVISÉE DE DOCUMENTS ÉTUDE COMPARATIVE.* QUÉBEC : Mémoire . EXIGENCE PARTIELLE DE LA MAÎTRISE EN SCIENCES ET TECHNOLOGIES DE L'INFORMATION. , 2016.

19. S. J. Russell, P. Norvig.*Artificial Intelligence-A Modern Approach (3rd internat.edn.).* s.l. : Pearson Education, 2010.

20. Encyclopédie Larousse en ligne - intelligence artificielle.

21. Bisong, E.*Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners.* s.l. : GoogleColaboratory BT.

22. F, M. X. I. I. Courset.*Boudin, Machine Learning avec Weka.* 2012.

23. A.Yahi.*Clustering des données de puces à ADN,mémoire de maîtrise, Département informatique, Université de M'sila.* M'sila , Algérie : s.n., 2019.

24. Y.BENDAOUD.*Prédiction Des Résistances Mécaniques Des Bétons à Base Des Ciments Composés En Utilisant Les Réseaux Neurones Artificielsmémoire de maîtrise , .* Constantine , Algérie : s.n., 2014.

25. Enparticulierdans McCulloch, W. S.Pitts, W. A Logical Calculus of the Ideas Immanent in Nervous Activity", Bulletin of Mathematical Biophysics,.

26. D.Dahmani.*"Fouille des règles d'association guidée par des ontologies et des schémas de règles" ,.* Oran , Algérie : s.n., 2011.

27. R. Agrawal, T. Imielinsk,A. Swami*"Mining Association Rules Between Sets of Items in Large Databases"*..207-216, s.l. : SIGMOD Conference , 1993.

28. *"Des réseaux de neurones. Deuxième édition".* 1993.

29. M.BOUAZIZ.*"Réseaux de neurones récurrents pour la classification de séquences dans des flux audiovisuels parallèles»,.* VAUCLUSE : s.n., 2017.

30. Cruse, Holk.*"Neural Networks as Cybernetic Systems" ,.* 2nd and revised edition.

31. Elman, Jeffrey L.*"Finding Structure in Time ".* 179–211 , s.l. : Cognitive Science, 1990, vol. 14, no 2,.

32. A. Graves & J. Schmidhuber.*"Framewise phoneme classification with bidirectional LSTM and other neural network architectures".* s.l. : Neural Networks, 2005. 602–610.

33. A.Parthipan, S.Chandar et.*" On Challenges in Training Recurrent Neural Networks",thèse de doctorat, Université de Montréal,.* Montréal,Canada : s.n., 2019.

34. Felix A. Gers, JurgenSchmidhuber, and Fred A.Cummins.*"Learning to forget: ¨ continual prediction with LSTM. Neural Computation".*

35. Schmidhuber, A. Graves & J.*"Framewise phoneme classification with bidirectional LSTM and other neural network architectures".* 602–610, s.l. : Neural Networks, 2005, Vol. 18.

36. Elman, Jeffrey L.*“Finding Structure in Time ”.* s.l. : Cognitive Science, 1990. Vol. 14.

37. M. Biniz, S. Boukil, F. El Adnani, L. Cherrat, and A. E. El Moutaouakkil . *Arabic text classification using deep learning techniques.* s.l. : Int. J. GridDistrib. Comput., 2018., Vol. vol. 11. 09 .

38. A. Elnagar, O. Einea, and R. Al-Debsi.*Automatic text tagging of Arabic news articles using ensemble deep learning models.* 59 66, s.l. : Conf.Natural Lang. Speech Process, 2019.

39. Mohamed Galal, Magda M.Madbouly ,Adel El-Zoghby.*Classifiying Arabic text using deeplearning .* 23, s.l : Journal des technologies de l'information théoriques et appliquées, 15e Décembre 2019, Vol. 97.

40. Kim, Y.*Réseaux de neurones convolutifs pour la classification des phrases.* 1746-1751, s.l. : Actes de laconférence 2014 sur les méthodes empiriques dans le traitement du langage naturel, 2014.

41. W. Oui, X. He et C. Meek . *Semantic analyse for single-relation question répondre.* 643-648, s.l. : dans Actes de la 52eréunion annuelle de l'Association for Computational Linguistics, 2014, Vol. Volume 2: Short Papers.

42. N. Kalchbrenner, E. Grefenstette et P. Blunsom.*Un réseau de neurones convolutifs pour la modélisation de phrases* .2188, s.l. : arXiv, 2014.

43. Muhammad Zulqarnain, Rozaida Ghazali, YanaMazwinMohmadHassim, & Muhammad Rehan*Text classification based on gated recurrent unit combines with support vector machine.*. 3734–3742, s.l. : International Journal of Electrical and Computer Engineering (IJECE), 2020, Vol. 10(4).

44. Hamidzadeh, V. Hooshmand Moghaddam et J. Nouveau noyau polynomial orthogonal Hermite et noyaux combinés dans le classificateur.

45. Z.YangetX.Pang.*Recherche et misee en ouevre'un modèle de classification de texte basé surla combinaison de DAE et DBN.* p.190-193, s.l. : Symp.Calcul.Informer.Des, 2017.

46. Elharbi, Omar.*A Deep Learning Approach Combining CNN and Bi-LSTM with SVM Classifier for Arabic Sentiment Analysis.* 6, s.l. : journal international d'informatique avancée d'application, 2021 , Vol. 12.

47. Al-Harbi, O. *Using Objective Words in the Reviews to Improve the Colloquial .* 1-14, s.l. : International Journal on Natural Language Computing, 2017., Vol. 6(3).

48. A. B. Soliman, Eissa, K., and El-Beltagy, S. R *. A Set of Arabic Word Embedding Models for Use in Arabic NLP* .. 256-265, s.l. : Procedia Computer Science, 2017. 117.

49. A. B. Soliman, Eissa, K., and El-Beltagy, S. R.A Set of Arabic Word Embedding Models for Use in Arabic NLP . 256-265, s.l. :Procedia Computer Science, 2017. 117.

50. Sara, Gherabi.Classification automatique des textes arabes (arabic opinion polarity). Msila : Mémoire de fin d'étude de master : université de msila, département de l'informatique, 2014.

51. https://github.com/linuxscout/tashaphyne. [En ligne] 23/09/2021

52. https://ichi.pro/fr/optimisation-des-hyperparametres-90586375491117. [En ligne] 21/08/2021

53. https://www.editions-eni.fr/open/mediabook.aspx?idR=4e861591c002497aa6c46f6ce53687ec.

[En ligne] 22/08/2021

54. https://www.editions-eni.fr/open/mediabook.aspx?idR=f6e7a7353a3574180124387fa03fdc1c.

[En ligne] 22/08/2021

55. https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/categorical-crossentropy. [En ligne] 22/08/2021

56. https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917584a. [En ligne] 22/08/2021

57. https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445278-buffer-definition-et-fonctionnement-pratique/ #:~:text=La%20m%C3%A9moire%20tampon%20 (buffer % 20 ensein %20 d'un%20ordinateur%20moderne. [En ligne] 20/09/2021

58. https://dataanalyticspost.com/Lexique/word-embedding/ .[En ligne] 15 /08/ 2021

59. https://www.it-swarm-fr.com/fr/machine-learning/comment-determiner-le-nombre-de-couches-et-de-noeuds-dun-reseau-neuronal/824296035/.[En ligne] 08/09/2021

60. BINIZ, mohamed. Beni-Mella .DataSet for Arabic Classification V2.,Maroc : Université Sultan MoulaySlimane , Université ChouaibDoukkali, 2018.

61. https://analyticsinsights.io/top-10-des-librairies-de-deep-learning-sur-python/.[En ligne] 16/09/2021

62. https://www.01net.com/telecharger/windows/Programmation/sript_macro/fiches/49145.html.[En ligne] 16/09/2021

63. https://pandas.pydata.org/. [En ligne] .22/09/2021

64. https://datascientest.com/matplotlib-tout-savoir. .[En ligne] 20/09/2021

65.Motaz K. Saad, WesamAshour "Arabic Morphological Tools for Text Mining", EEECS'10 the 6th International Symposium on Electrical and Electronics Engineering and Computer Science..European University of Lefke, 2010. pp. 112-117.

66. https://osdn.net/projects/sfnet_ar-text-mining/downloads/Arabic-Corpora/osac-uft8.7z/[En ligne] . 20/09/2021 .