REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : SIOD 6/M2/2021

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Système d'Information Optimisation et Décision (SIOD)**

Analyse des opinions basées sur l'apprentissage en profondeur

---

Par :

AMEUR HMAIDA

Soutenu le 07 juillet 2021 Devant le jury composé de :

Djeffal Abdelhamid	Professeur	Président
Abdeli Belkacem	MCB	Rapporteur
Hoadjli Hadia	MAA	Examinateur

Année universitaire 2020-2021

Remerciement

Je voudrais d'abord remercier Dieu tout-puissant de m'avoir donné du courage et de la force patience et optimisme pour mener à bien ce travail.

...

Je voudrais aussi exprimer mes remerciements à ceux qui, comme j'ai reçu du soutien, sont les fleurs de ma vie. Je me souviens en particulier de ma famille et commencer ma mère et mon père qui sont chers à mon cœur et ont le plus grand crédit pour ce que je suis aujourd'hui.Nous n'oublions pas les fleurs de ma vie, surtout Insaf, Asma et Maryiem. Et aussi mes frères Zino et Islam

...

Je tiens à exprimer mes remerciements à la personne qui m'a aidé à rédiger cette note, en particulier à mon encadreur « Abdeli Belkacem » pour le suivi de mon travail, ses conseils, ses encouragements, sa patience, son aide précieuse, et pour le temps qu'il a m'a été attribué.

...

Enfin, j'adresse mes sincères remerciements au chef du département, aux professeurs distingués, ainsi qu'à mes chers camarades étudiants et à tous ceux qui ont contribué à la réussite de près ou de loin

...

Merci

Dédicas

Je dédie cet humble travail en signe d'affection, de respect et d'admiration.
Tous ceux qui me sont chers, à mes chers père et mère, à mes frères et sœurs.
Amis et toutes personnes ayant contribué directement ou indirectement à la réalisation de ce mémoire mémoire ouvrage.

Résumé :

L'analyse des données est considérée comme l'un des domaines les plus importants de l'ère moderne, surtout après l'émergence des sites de réseaux sociaux et leur développement au cours de la dernière décennie, ce qui a conduit à la nécessité d'analyser ces données et de les utiliser dans divers secteurs tels que la santé, l'éducation, la sécurité et d'autres secteurs, et tout cela dans le but d'améliorer les produits ou les services.

Parmi les choses qui ont attiré l'attention des chercheurs, des gouvernements et des institutions privées et publiques, il y a l'analyse des sentiments et des opinions et l'étendue de leur impact sur l'amélioration et aider les décideurs à prendre les meilleures décisions.

Dans ce travail, nous avons tenté de mettre en œuvre une technique basée sur apprentissage en profondeur spécifiquement réseau de neurones récurrents avec mémoire à long terme.

Mots-clés : analyse des sentiments, analyse d'opinion, exploration d'opinion, exploration de texte, traitement du langage naturel, apprentissage en profondeur, réseau de neurones récurrents.

Abstract :

Data analysis is considered one of the most important fields in the modern era, especially after the emergence of social networking sites and their development in the last decade, which led to the necessity of analyzing this data and making use of it in various sectors such as health, education, security and other sectors, and all of this is in order to improve products or services.

Among the things that have attracted the attention of researchers, governments, and private and public institutions is the sentiment analysis and opinions analysis and the extent of their impact on improvement and help decision makers take the best decisions.

In this work, we have attempted to implement a technique based on deep learning specifically recurrent neural network with Long short-term memory.

Keywords : sentiment analysis, opinion analysis, opinion mining, text mining, natural language processing, deep learning, recurrent neural network.

ملخص :

تعتبر تحليل البيانات من اهم الميادين في العصر الحديث خاصة بعد ظهور مواقع التواصل الاجتماعي وتطورها في العقد الاخير مما ادى الى وجوب تحليل هذه البيانات والأستفاذة منها في قطاعات مختلفة كالصحة والتعليم والامن وغيرها من القاطاعات وهذا كله من اجل تحسين المنتجات أوالخدمات.

ومن بين الامور التي جلبت اهتمام الباحثين والحكومات والمؤسسات الخاصة والعامة هو تحليل المشاعر والاراء ومدى تأثيرها في التحسين ومساعدة صناع القرار في اتخاذ احسن القرارت.

في هذا العمل ، حاولنا تنفيذ تقنية تعتمد على التعلم العميق على وجه التحديد الشبكة العصبية المتكررة ذات الذاكرة طويلة المدى

الكلمات المفتاحية: تحليل المشاعر ، تحليل الرأي ، التنقيب عن الرأي ، التنقيب عن النصوص ، معالجة اللغة الطبيعية ، التعلم العميق ، الشبكة العصبية المتكررة

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction Générale

1.1 Introduction Générale

Dans les dernières années avec le développement technologique sur plusieurs domaines et spécialement dans les réseaux sociaux (Facebook, Twitter, YouTube ...), le monde a vu des mutations énormes et une révolution soit positive ou bien négative dans différents domaines (Politique, économique, médical, sport, Airlines, produit, éducation ...).

Parmi les grandes révolutions : le Big Data, ou bien « données massives » est le pétrole du 21e siècle, qui inciter les chercheurs à trouver de nouvelles méthodes pour analyser ces méga données. Aujourd'hui, l'internet dépasse 3 milliards d'utilisateurs dans le monde et selon les dernières statistiques il y a plus de 200 millions email et 20 millions tweet, chaque minute.

Avec l'apparition de la notion de Big data, Une nouvelle thématique du traitement du langage (TALN), est développée, connue sous le nom d'analyse des sentiments d'opinion (Opinion mining). Le but principal de l'analyse sentimental est d'extraire les sentiments et les opinions des utilisateurs à partir des contenus créés en utilisant des techniques d'extraction automatique pour déterminer leurs attitudes par rapport à un sujet, souvent exprimés sous forme textuelle.

L'analyse des opinions est plus intéressent pour les gouvernements et les entreprises pour extraire à partir les textes et les commentaires, l'orientation et les tendances des communautés et les nations pour optimisation réalisation des aspirations soit personnel ou social. Pour cette raison, les plus grandes Entreprises mondiales Sont en concurrence dans ce domaine, et investissent beaucoup d'effort et de temps, parmi ces entreprise on trouve les géants de l'informatique : (Facebook, Twitter, Amazone, Microsoft, ...) et surtout Google Qui est considéré comme le leader dans le domaine.

Le but de ce travail est d'étudier l'analyse des sentiments en traitant les textes de la langue naturelle. il existe plusieurs types d'émotions (la peur, Tristesse, joie, colère Anxiété, ex..). Et dans notre travail Nous allons se concentré sur les émotions positives, négatives et neutres.

L'analyse des sentiments comprend des défis et des problèmes que nous essaierons de résoudre dans ce travail. Parmi ces problèmes l'amélioration de la performance d'analyse, de gros corpus documentaire.

Nous essayons de fournir une plate-forme pour une analyse des sentiments haute performance et précise en utilisant l'apprentissage en profondeur pour obtenir des résultats meilleurs et précis. Nous connaitrons également l'impact des données sur l'exactitude du résultat en termes de qualité et de quantité de données.

Il existe 4 chapitres pour étudier ce travail 1e chapitre pour introduction et expliquer les analyses des sentiments et opinions, 2e chapitre pour l'apprentissage en profondeur ,3e chapitre nous expliquerons la conception de noter le projet avec détaille, 4e chapitre pour expliquer l'implémentation de noter projet avec des exemples.

Chapitre 2

Analyse des sentiments

2.1 Introduction

L'analyse des sentiments est l'un des domaines qui connaît un grand intérêt depuis une quinzaine d'années, et de nos jours il est très utilisé par les grandes firmes et les grands acteurs de l'informatique comme Google, Facebook, Microsoft, ...etc. Et il est utilisé même pour prédire le comportement des personnes.

Dans ce chapitre nous représentons les notions fondamentales d'analyse des sentiments, Après la définition générale du domaine de traitement automatique du langage naturel la définition analyse des sentiments et d'une opinion, nous expliquerons les types d'opinions, les tâches de l'analyse des sentiments, les techniques utilisés pour réaliser cette analyse.

2.2 Définitions

Dans cette section, nous expliquons le domaine de traitement automatique du langage naturel où l'analyse des sentiments fait une partie de ce domaine, puis nous clarifions c'est quoi un sentiment et une opinion et la différence entre eux.

2.2.1 Le traitement naturel du langage NLP

2.2.1.1 Historique

Le Natural Language Processing, ou Traitement Automatique du Langage, n'est pas une discipline neuve. Son origine remonte à la fin de la deuxième guerre mondiale, avec des recherches portant principalement sur la traduction automatique entre différentes langues. En 1954, un ordinateur réussit à traduire automatiquement 60 phrases du Russe à l'Anglais. La publication en 1957 du livre Syntactic structures par Noam Chomsky fut une révolution pour le domaine. Il y montra notamment qu'il existe des caractéristiques communes à tous les langages et inventa un type de grammaire qui convertit le langage naturel en une forme compréhensible par des ordinateurs.

A partir des années 80, l'augmentation de la capacité de traitement des ordinateurs, puis le développement d'Internet et de la communication textuelle numérisée (sms, emails, réseaux sociaux...), ainsi que plus récemment l'émergence d'infrastructures Big Data et d'algorithmes d'Intelligence Artificielle ont permis une explosion des capacités et des applications du Natural Language Processing.[13].

2.2.1.2 Définition

Le Traitement Automatique du Langage (ou « Natural Language Processing » en Anglais) correspond à un cycle automatisé par l'informatique lecture/correction/analyse de données textuelles pour en retirer différents types d'information. Une de ses déclinaisons fréquemment utilisées pour la recherche de données s'appelle le « Text Mining ». De plus, le Traitement Automatique du Langage est de nos jours souvent supporté par des algorithmes d'Intelligence Artificielle ou Machine Learning.[13]

2.2.2 Opinion

2.2.2.1 Définitions

L'opinion est un jugement que l'on porte sur un individu, un être vivant, un phénomène, un fait, un objet ou une chose. Elle peut être considérée comme bonne ou mauvaise. L'opinion peut influencer et peut donner de bonnes ou mauvaises informations sur un sujet étudié au sein d'un groupe, d'une personne, d'un objet.

Une opinion est un jugement, un point de vue ou une déclaration qui n'est pas concluante. Il peut traiter de questions subjectives dans lesquelles il n'y a pas de conclusion concluante, ou traiter des faits qui sont contestés par l'erreur logique que l'on a droit à leurs opinions. Ce qui distingue le fait de l'opinion, c'est que les faits sont plus susceptibles d'être vérifiables, c'est-à dire qu'ils peuvent être acceptés par le consensus des experts. Un exemple est : "l'Algérie a été colonisé par la France" contre "la France a eu raison de coloniser l'Algérie". Une opinion peut être étayée par des faits et des principes, auquel cas elle devient un argument.

Des personnes différentes peuvent tirer des conclusions opposées (opinions) même si elles sont d'accord sur le même ensemble de faits. Les opinions changent rarement sans que de nouveaux arguments soient présents. On peut raisonner qu'une opinion est mieux soutenue par les faits qu'une autre en analysant les arguments à l'appui. Dans un usage occasionnel, le terme d'opinion peut être le résultat de la perspective, de la compréhension, des sentiments particuliers, des croyances et des désirs d'une personne. Il peut se référer à des informations non corroborées, contrairement aux connaissances et aux faits.[12].

2.2.2.2 L'objective de la fouille d'opinion

La fouille d'opinion, en particulier à partir de données des réseaux sociaux, est un excellent substitut nettement moins coûteux des enquêtes d'opinion ou ils montrent à travers[33] :

— Evaluation des produits, d'une personnalité.

— Améliorer les systèmes de recommandation.

— Analyse de la popularité, des tendances.

— Positionnement par rapport à un sujet délicat.

2.2.2.3 Type Opinion

On peut distinguer deux types d'opinions la première s'appelle opinion régulière. L'autre type est appelé opinion comparative. En fait, nous pouvons également classer les opinions en fonction de la façon dont ils sont exprimés dans le texte, l'opinion explicite et l'opinion implicite.[8]

Opinion régulière : Une opinion régulière est souvent simplement considérée comme une opinion dans la littérature et il y a deux sous-types principaux.[28]

- Opinion directe : Une opinion directe fait référence à une opinion exprimée directement sur une entité ou un aspect de l'entité, par exemple, "La résolution de cet écran est excellente".[28]

- Opinion indirecte : Une opinion indirecte est une opinion exprimée indirectement sur une entité ou aspect d'une entité en fonction de ses effets sur d'autres entités. Ce sous-type se produit souvent dans le domaine m´médical. Par exemple, la phrase "Après l'injection du médicament, mes articulations senties pire" décrit un effet indésirable du médicament sur" mes articulations ", ce qui donne indirectement une opinion négative ou un sentiment au médicament. Dans le cas, l'entité est le médicament et l'aspect est l'effet sur les articulations.[28]

Opinion comparative : Un avis comparatif exprime une relation de similitudes ou de différences entre deux ou plusieurs entités et/ou une préférence du détenteur d'opinion

sur la base de certains aspects partagés des entités.[24]

- Opinion explicite : Une opinion explicite est une déclaration subjective qui donne une opinion régulière ou comparative, par exemple : "Le couscous a bon goût" et "Facebook est mieux que twitter.[24]

- Opinion implicite : Une opinion implicite est une déclaration objective qui implique une opinion régulière ou comparative. Une telle déclaration objective exprime habituellement un fait souhaitable ou indésirable, par exemple : "La durée de vie de la batterie de l'ordinateur portable Toshiba est plus longue que celle de l'ordinateur portable HP".[24]

2.3 Analyse des Sentiments

2.3.1 Définition

L'analyse des sentiments, également appelée exploration d'opinions, est le domaine d'étude qui analyse les opinions, les sentiments, les évaluations, les appréciations, les attitudes et les émotions des personnes envers des entités telles que des produits, des services, des organisations, des individus, des problèmes, des événements, des sujets et leurs attributs. Il représente un grand espace de problème. Il existe également de nombreux noms et des tâches légèrement différentes, par exemple, analyse des sentiments, exploration d'opinion, extraction d'opinion, exploration de sentiment, analyse de subjectivité, analyse d'affect, analyse d'émotion, exploration d'examen, etc.[29]

L'analyse de sentiments détermine l'orientation globale du sentiment d'un locuteur ou d'un écrivain vers une entité spécifique ou vers une caractéristique spécifique d'une entité spécifique.

Une tache fondamentale de l'analyse du sentiment est la classification des sentiments qui visent à classer automatiquement le texte opiniâtre comme positif, négatif et neutre.

2.3.2 Tâches de l'analyse des sentiments

Il existe différentes tâches dans l'analyse des sentiments :

2.3.2.1 Analyse de la subjectivité et détection de l'opinion

L'analyse de la subjectivité et détection de l'opinion consiste à déterminer si un texte donné contient une opinion ou non. Ce problème a été aborde dans un premier temps indépendamment de l'analyse de sentiments avant de devenir une tâche de base, mais elle n'en reste pas moins l'une des plus difficiles.

La recherche dans la détection automatique de l'opinion 'a partir du texte a été initie par (Wiebe et al., 1999 [23]) avec des travaux où ils proposent des méthodes discriminatives entre le texte objectif et le texte subjectif au niveau document, phrase et expression en utilisant un classifieur Naïve Bayes. Ce classifieur utilise un ensemble de caractéristiques à savoir la présence ou l'absence de classes syntaxiques particulières, la ponctuation et la position des phrases. Ces caractéristiques sont jugées indicatrices de subjectivité.

Par la suite, (Hatzivassiloglou and Wiebe, 2000 [21]) démontrent que les adjectifs gradables 1automatiquement détectés sont une caractéristique utile pour la classification de la subjectivité. Plus récemment, (Wilson et al., 2005 [38]) ont effectué un travail pour la classification de la subjectivité au niveau document en utilisant l'algorithme des k plus proches voisins basé sur le nombre total de mots et expressions de subjectivité dans chaque document.

2.3.2.2 Catégorisions des sentiments

Le premier objectif en manière d'analyse des sentiments consiste généralement à distinguer les phrases subjectives des phrases objectives. Si une phrase donnée est classée comme objective, aucune autre tâche fondamentale n'est pas requise, tandis que si la phrase est classée comme subjective, sa polarité (Positive, négative, neutre) doit être estimée. La classification de subjectivité est la tache distingue les phrases exprimant des informations objectives (ou factuelles) (phrases objectives) des phrases exprimant des vues et opinion subjectives (phrases subjectives).[19]

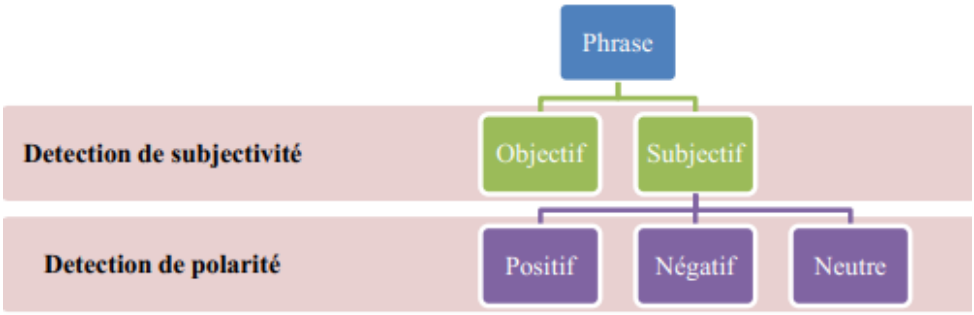


FIGURE 2.1: Catégorisions des sentiments

2.3.2.3 Identifications de sujet et du porteur d'opinion

Une autre tâche de base de l'analyse de sentiments est la détection du porteur d'opinion et l'identification du sujet. L'avantage de cette tâche est de pouvoir filtrer les opinions selon un sujet particulier ou alors de regrouper les opinions d'une personne particulière pour des fins de personnalisation en sélectionnant les sujets que ce dernier préfère.[7]

2.3.3 Techniques

Dans les modèles d'analyses il existe Trois catégories d'analyse :

2.3.3.1 Apprentissage automatique

Appelé aussi approche statistique, cette approche se basé sur l'apprentissage automatique. Elle utilise la technique de classification pour classer le texte en des classes déférentes. Il existe principiellement deux types e technique d'apprentissage [15]

Apprentissage supervise : Il est basé sur les donnés libellées et par conséquent, les étiquettes sont fournies au modèle au cours du processus d'apprentissage. Ces données libellées sont utilisées par l'algorithme d'apprentissage pour donner un modèle qui sera utilisée lors de la prise de décision. Les techniques d'apprentissage automatique comme Naïve Bayes (NB), l'entropie maximal (ME), et les machines à vecteur de support (SVM) ont donné un grand succès dans l'analyse des sentiments.[16]

Apprentissage non supervise : Il ne consiste pas d'une classification précise, donc il se base sur le regroupement. Le succès de deux méthodes d'apprentissage dépend princi-

palement de la sélection et l'extraction de l'ensemble des descripteurs utilisé pour détecter le sentiment (le classe), les algorithmes d'apprentissage non supervisés classification hiérarchique ascendante, canters mobiles, règles d'association ... etc.[16]

2.3.3.2 Apprentissage Lexique

Les approches basées sur le lexique reposent principalement sur un lexique de sentiment, c'est-à-dire une collection de termes, phrases et même idiomes de sentiment connus et précompilés, développes pour des genres de communication traditionnels.

Approche basé sur le dictionnaire : Le premier est généralement basé sur l'utilisation d'un ensemble initial de termes (graines) qui sont habituellement collectés et annotés de manière manuelle. Cet ensemble se développe en recherchant les synonymes et les antonymes d'un dictionnaire. Un exemple de ce dictionnaire pourrait être WordNet, qui a été utilisé pour développer un thésaurus appelé SentiWordNet.

Le principal inconvénient de ce type d'approche est l'incapacité de traiter les orientations spécifiques au domaine et au contexte, même ainsi, cela pourrait être une solution intéressante selon le problème.[30]

basé sur le corpus : Les techniques basées sur le corpus ont pour objectif de fournir des dictionnaires liés à un domaine spécifique. Ces dictionnaires sont générés à partir d'un ensemble de termes d'opinion de la graine qui se développe à travers la recherche de mots apparentés au moyen de l'utilisation de techniques statistiques ou sémantiques.

Des méthodes basées sur des statistiques telles que l'analyse sémantique latente (LSA), ou simplement la fréquence d'occurrence des mots dans une collection de documents peuvent être utilisées. Et d'autre part, les méthodes sémantiques telles que l'utilisation de synonymes et d'antonymes ou de relations à partir de thésaurus comme WordNet peuvent également représenter une solution intéressante.[30]

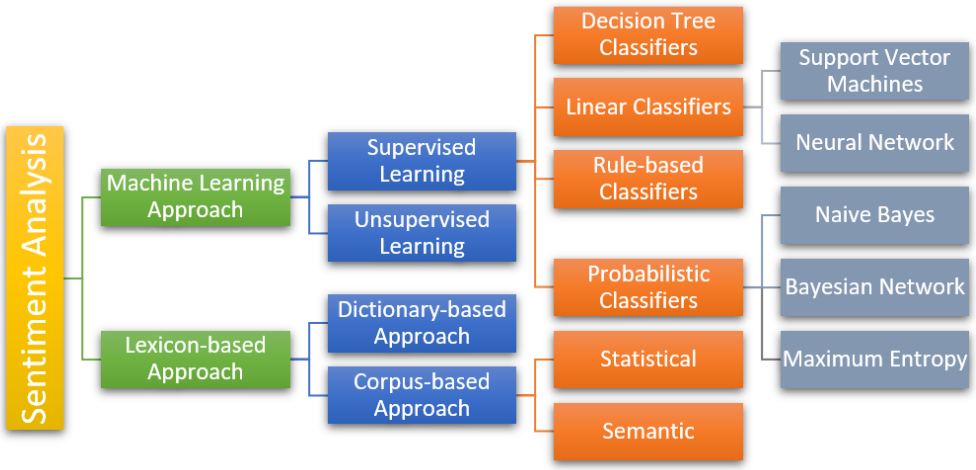


FIGURE 2.2: les Techniques analyse des sentiments

2.3.4 Les domaines d'applications

L'importance de la détection d'opinion est présentée dans plusieurs domaines ainsi plusieurs applications ont vu le jour dans ce contexte. Nous citons brièvement quelques applications ci-dessous :

2.3.4.1 Marketing et production

Le but d'analyse des sentiments est la connaissance des opinions à propos de la production sur les commentaires tweet dans les réseaux sociaux ou autres ressources pour optimiser les paramètres dans le futur et connaitre les points Faibles et éviter les erreurs

2.3.4.2 La politique

Les politiciens sont parmi les personnes les plus intéressées par l'analyse des sentiments,pour Apporter le plus grand nombre de votes des élections à travers tendance et opinion chaque personne et chaque région et Ceux qui sont contre eux et ceux qui sont avec eux, et sur les statistiques bien préparer pour l'élection.

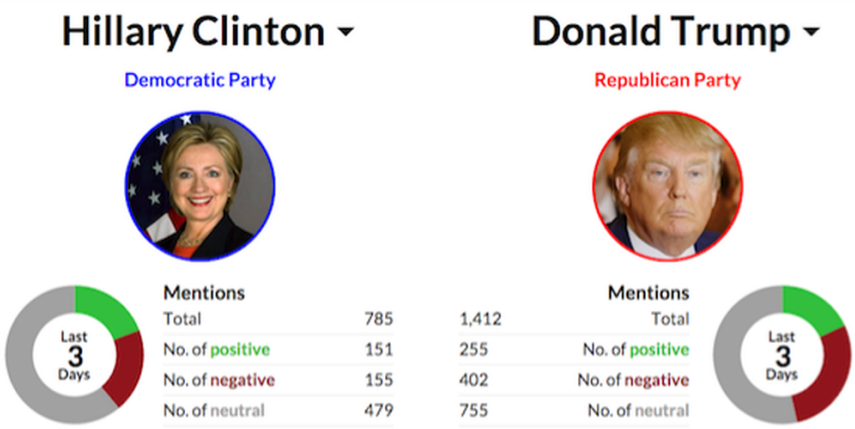


FIGURE 2.3: Exemple d'analyse dans le domaine du l'élection America

2.3.4.3 Réseaux Sociaux

Sont les plus grands domaines qui utilisent l'analyse des sentiments pour extraire les opinions et les tendances des utilisateurs.

L'analyse des sentiments permet la classification des utilisateurs dans les groupes par la même orientation pour la satisfaction des utilisateurs et optimiser les paramètres.

L'analyse des sentiments est utilisée aussi dans d'autres applications (ex : Le sport, Les film, Le médicaux, Les restaurant. . .).

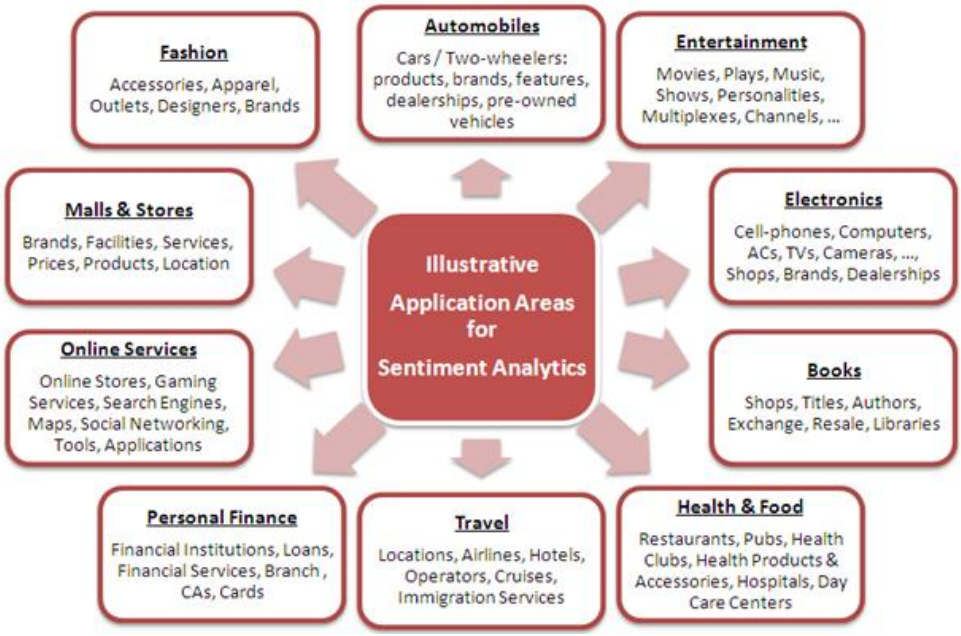


FIGURE 2.4: Les domaines d'applications analyse des sentiments

2.4 Conclusion

L'analyse des sentiments est un domaine intéressant, où ce domaine est largement utilisé par les grandes entreprises et les grandes firmes pour avoir une idée de la façon dont les clients sont heureux avec les produits à partir du rapport entre les tweet positifs et négatifs à leur sujet. Il peut également être utilisé pour trouver des personnes qui sont satisfaites des produits ou services et leurs expériences peuvent être utilisés pour promouvoir ces produits.

Chapitre 3

Apprentissage profond

3.1 Introduction

L'apprentissage profond (en anglais deep Learning, deep structured Learning) est un ensemble de méthodes d'apprentissage automatique tentant de modéliser avec un haut niveau d'abstraction des données grâce à des architectures articulées de différentes transformations non linéaires. L'apprentissage automatique champ d'étude de l'intelligence artificielle.

Dans ce chapitre, nous présentons les notions fondamentales de l'apprentissage profond. Après la définition générale de L'apprentissage automatique, l'apprentissage profond et des réseaux de neurones, nous clarifions les paradigmes d'apprentissage, les architectures d'apprentissage profond et la motivation derrière l'utilisation de l'apprentissage profond. Ensuite nous expliquons les applications de l'apprentissage profond, et aussi les défis de l'apprentissage profond avec les plus et les moins du l'apprentissage en profondeur.

3.2 L'apprentissage automatique

L'apprentissage automatique concerne tout type de programme informatique qui peut « apprendre » par lui-même sans avoir à être explicitement programmé par un humain. Aujourd'hui, l'apprentissage automatique est un terme largement utilisé qui englobe de nombreux types de programmes que vous rencontrerez dans l'analyse des mégadonnées et l'exploration de données. En fin de compte, les « cerveaux » alimentant la plupart des programmes prédictifs – y compris les filtres anti-spam, les recommandations de produits et les détecteurs de fraude – sont des algorithmes d'apprentissage automatique.[39]

3.2.1 Les types d'apprentissage automatique

Les algorithmes d'apprentissage automatique peuvent être divisés en catégories d'apprentissage supervisé et apprentissage non supervisé, mais il y a aussi d'autres types comme l'apprentissage par renforcement et l'apprentissage semi-supervisé.

3.2.1.1 Apprentissage supervise

Les algorithmes d'apprentissage supervisé subissent des données contenant des caractéristiques, mais chaque exemple est également associé à une étiquette ou une cible. Par

exemple, notre ensemble de données est annoté avec positive ou négative. Un algorithme d'apprentissage supervisé peut étudier l'ensemble de ces données et apprendre à classer les commentaires en classes différentes en fonction de leurs sentiments soit négative soit positive.[7]

3.2.1.2 Apprentissage non-supervise

L'apprentissage non supervisé décrit une classe de problèmes qui implique l'utilisation d'un modèle pour décrire ou extraire des relations dans les données.
Par rapport à l'apprentissage supervisé, l'apprentissage non supervisé fonctionne uniquement sur les données d'entrée sans sorties ni variables cibles. En tant que tel, l'apprentissage non supervisé n'a pas d'enseignant corrigeant le modèle, comme dans le cas de l'apprentissage supervisé.[9]

3.2.1.3 Apprentissage semi-supervisé

L'apprentissage semi-supervise utilisent également des données non étiquetées pour l'apprentissage, généralement une petite quantité de données étiquetées avec une grande quantité de données non étiquetées. L'apprentissage semi-supervisé se situe entre un apprentissage non supervisé (sans données d'entraînements étiquetées) et un apprentissage supervisé (avec des données d'entraînements complètement ´étiquetées). De nombreux chercheurs en apprentissage automatique ont découvert que les données non étiquetées, lorsqu'elles sont utilisées conjointement avec une petite quantité de données étiquetées, peuvent entraîner une amélioration considérable de la précision de l'apprentissage.[35]

3.3 l'apprentissage en profondeur

3.3.1 Défintion

L'apprentissage profond est un type particulier d'apprentissage automatique qui atteint une grande puissance et flexibilité en apprenant à représenter le monde comme une hiérarchie imbriquée de concepts, chaque concept étant défini par rapport à des concepts plus simples et des représentations plus abstraites calculées en termes moins abstraits.
L'apprentissage profond permet à l'ordinateur de construire des concepts complexes à par-

tir de concepts plus simples. La figure 2.3 montre comment un système d'apprentissage profond peut représenter le concept d'image d'une personne en combinant des concepts plus simples, tels que des coins et des contours, qui sont à leur tour définis en termes d'arêtes.[22]

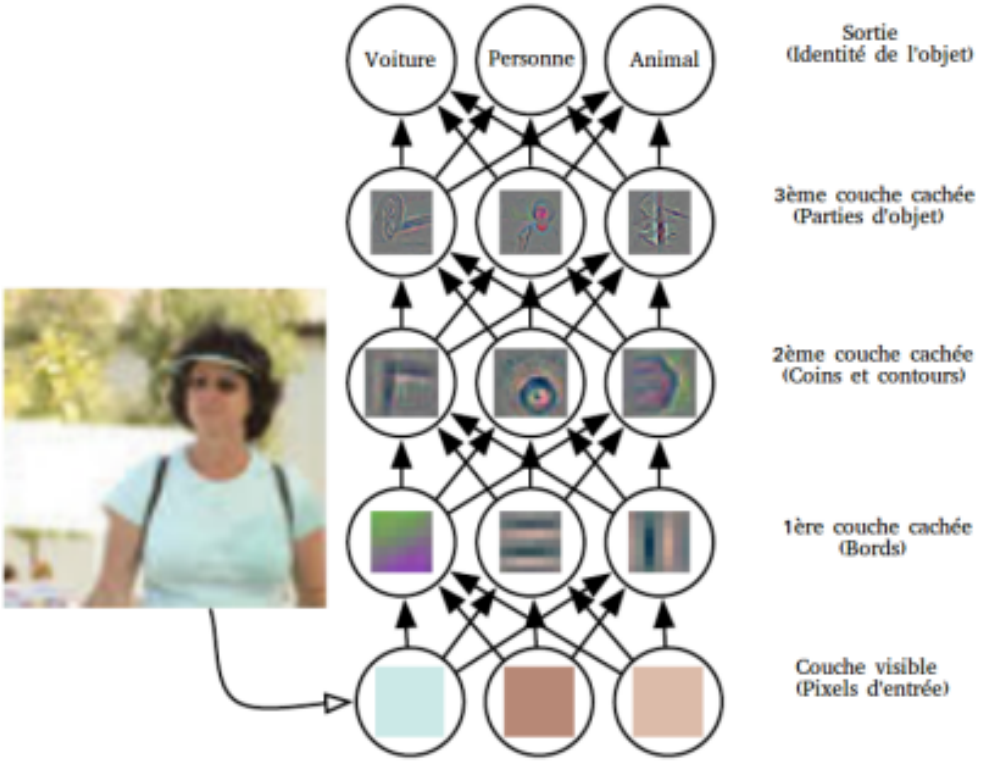


FIGURE 3.1: Illustration d'un model de l'apprentissage profond

3.3.2 Les réseaux des neurones

3.3.3 Neurone artificiel

Le neurone artificiel est un modèle de calcul dont la conception est inspirée du fonctionnement de vrais neurones. Ce neurone formel peut être considéré comme un opérateur recevant un nombre variable d'entrées du milieu extérieur ou d'autres neurones, chacune de ces entrées est pondérée par poids dit poids synaptique, et fournissant une sortie seulement quand la somme dépasse un certain seuil interne.[18]

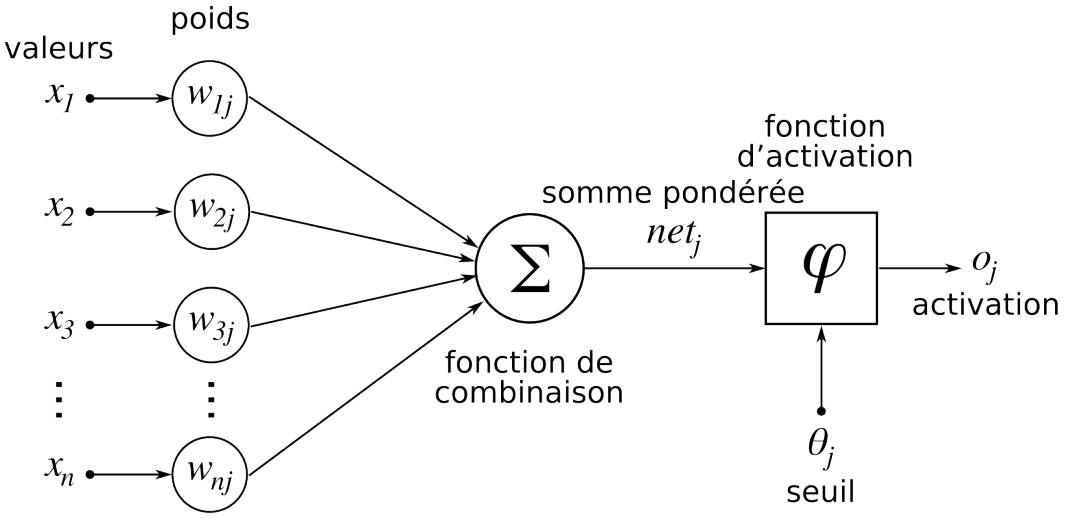


FIGURE 3.2: Neurone artificiel

L'évaluation de la sortie se fait typiquement par la somme pondérée des entrées, et le passage du résultat à travers une linéarité. Mathématiquement, ceci peut être modélisé par les équations suivantes :

$$S = \sum Wi.Xi + W0.X0$$

Y=f(S).

Xi : Composantes du vecteur d'entrée

Wi : Composantes du vecteur poids synaptique

S : Somme pondérée appelée potentiel

Le terme :

$$w0.x0 = \ominus \; avec \; x0 = 1$$

représente la valeur du seuil interne qui doit être dépassée pour l'activat1ion de la sortie du neurone. La non linéarité f(.) est appelée fonction d'activation.

La somme pondérée peut se réécrire sous la forme simple suivante[18] :

$$S = \sum Wi.Xi \; i = 0\,, n$$

3.3.4 Fonction d'activation

Après que le neurone a effectué le produit entre ses entrées et ses poids, il applique également une non-linéarité sur ce résultat. Cette fonction non linéaire s'appelle la fonction d'activation. La fonction d'activation est une composante essentielle du réseau neuronal.

Ce que cette fonction a décidé est si le neurone est activé ou non. Il calcule la somme pondérée des entrées et ajoute le biais. C'est une transformation non linéaire de la valeur d'entrée.

Après la transformation, cette sortie est envoyée à la couche suivante. La non-linéarité est si importante dans les réseaux de neurones, sans la fonction d'activation, un réseau de neurones est devenu simplement un modèle linéaire.[10] Les fonctions les plus couramment utilisées en deep learning sont :

3.3.4.1 La fonction Sigmoïde

Cette fonction est l'une des plus couramment utilisées. Il est borné entre 0 et 1, et il peut être interprété stochastique-ment comme la probabilité que le neurone s'active, et il est généralement appelé la fonction logistique ou le sigmoïde logistique.[6]

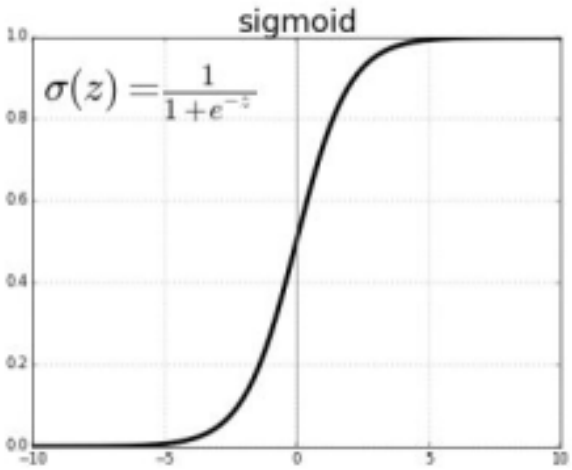


FIGURE 3.3: La fonction Sigmoide

3.3.4.2 La fonction ReLu

La fonction RELU est probablement la plus proche de sa correspondante biologique [41]. Cette fonction est récemment devenue le choix de nombreuses tâches (notamment en computer vision) [10]. Comme dans la formule ci-dessus, cette fonction renvoie 0 si l'entrée z est inférieure à 0 et retourne z lui-même si il est plus grande que 0.

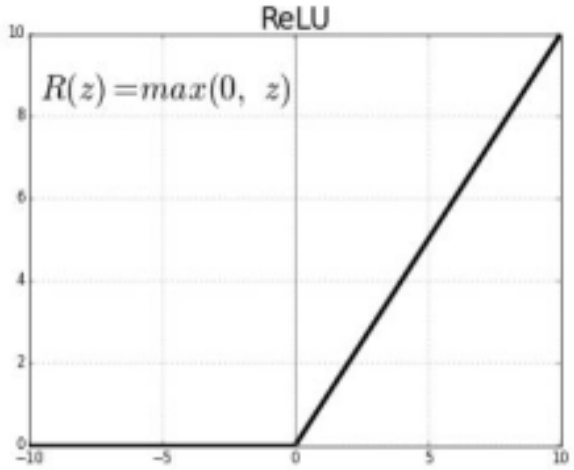


FIGURE 3.4: La fonction ReLu

3.4 Architecteur d'apprentissage profond

3.4.1 Réseau de neurones convolutif

Un réseau de neurones convolutifs ou réseau de neurones à convolution (en anglais CNN ou ConvNet pour Convolutional Neural Networks) est un réseau neuronal multicouche inspiré biologiquement du cortex visuel humain. L'architecture est particulièrement utile dans les applications de traitement d'image et les vidéos.
CNN est une séquence de couches, et chaque couche transforme un volume d'activations en un autre par une fonction différentiable. Les quatre principaux types de couches pour construire ce type de réseau sont : couche convolutive, couche de pooling, couche correction et couche entièrement connectée.[11]

3.4.1.1 Les étapes principales dans la conception CNN

Il existe 4 étapes ou couche principales dans la conception d'un CNN :

1.Couche convolutive : Les couches convolutives constituent le noyau du réseau convolutif. Ces couches se composent d'une grille rectangulaire de neurones qui ont un petit champ réceptif étendu à travers toute la profondeur du volume d'entrée. Ainsi, la couche convolutive est juste une convolution d'image de la couche précédente, où les poids spécifient le filtre de convolution.

La couche convolutive déterminera la sortie des neurones qui sont connectés aux régions locales de l'entrée par le calcul du produit scalaire entre leurs poids et la région connectée au volume d'entrée. ReLu vise à appliquer une fonction d'activation « élémentaire » telle qu'une fonction sigmoïde à la sortie de l'activation produite par la couche précédente.[36]

2.Couche polling (sous-échantillonnage) : Après chaque couche convolutive, il peut y avoir une couche de pooling. Cette couche sous échantillonne le long de la dimensionnalité spatiale de l'entrée donnée, ce qui réduira davantage le nombre de paramètres au sein de cette activation. Il y a plusieurs façons de faire cette mise en commun, comme prendre la moyenne ou le maximum, ou une combinaison linéaire prise par des neurones dans le bloc.[36]

3.Couche de correction (ReLU) : Souvent, il est possible d'améliorer l'efficacité du traitement en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique (fonction d'activation) sur les signaux de sortie.[36]

4.Couche totalement connectée : Après plusieurs couches de convolution et de max-pooling, le raisonnement de haut niveau dans le réseau neuronal se fait via des couches entièrement connectées. Les neurones dans une couche entièrement connectée ont des connexions vers toutes les sorties de la couche précédente (comme on le voit régulièrement dans les réseaux réguliers de neurones). Leurs fonctions d'activations peuvent donc être calculées avec une multiplication matricielle suivie d'un décalage de polarisation.[36]

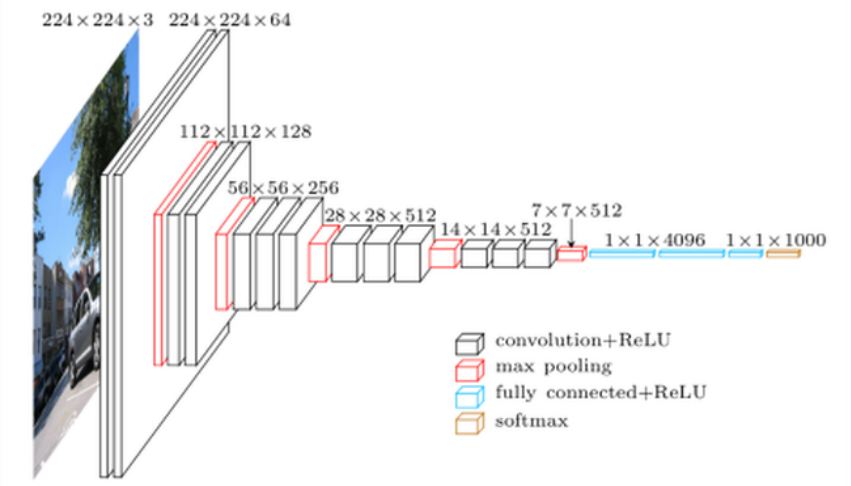


FIGURE 3.5: Réseau de neurones convolutif

3.4.1.2 Les avantages et inconvénients CNN

1.Avantages :

— Très bon pour la reconnaissance visuelle.

— Une fois qu'un segment dans un secteur particulier d'une image est appris, le CNN peut reconnaitre ce segment présent n'importe où ailleurs dans l'image.

2.Inconvénients :

— CNN dépend fortement de la taille et de la qualité des données de formation très sensibles au bruit.

3.4.2 Réseau de neurones récurrents

3.4.2.1 Définition

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

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

Les réseaux de neurones récurrents résolvent ce problème. Ce sont des réseaux avec des boucles qui permettent à l'information de persister. Dans la figure -dessus, un segment de réseau neuronal : « A » regarde une entrée « Xt » et fournit une valeur Ht. Une boucle permet de passer des informations d'une étape du réseau à l'autre.[34]

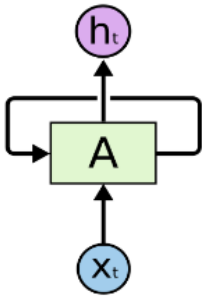


FIGURE 3.6: Les réseaux neuronaux récurrents ont des boucles.

Ces boucles rendent les réseaux neuronaux récurrents un peu mystérieux. Cependant, si vous pensez un peu plus, il s'avère qu'ils ne sont pas tous différents d'un réseau de neurones normal. Un réseau de neurones récurrent peut être considéré comme des copies multiples du même réseau, chacune transmettant un message à un successeur comme le montre la figure Considérez ce qui se passe si nous déroulons la boucle.

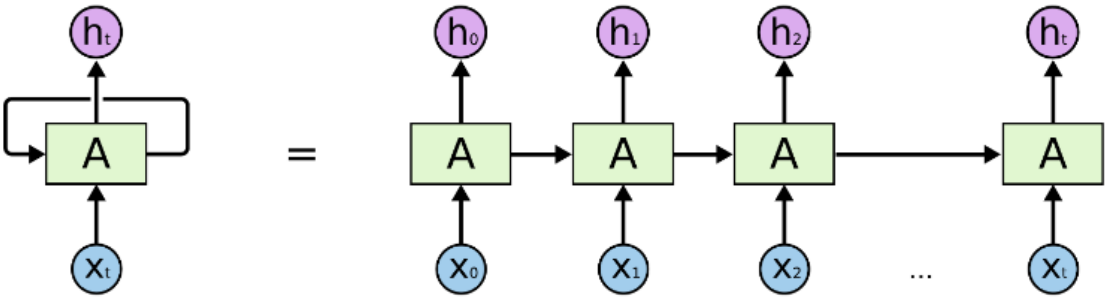


FIGURE 3.7: Un réseau neuronal récurrent déroulé.

Cette nature en chaîne révèle que les réseaux neuronaux récurrents sont intimement liés aux séquences et aux listes. Ils sont l'architecture naturelle du réseau de neurones à utiliser pour de telles données.

Au cours des dernières années, il y a eu un succès incroyable en appliquant les RNN à une variété de problèmes : la reconnaissance de la parole, la modélisation du langage, la traduction, le sous-titrage des ...etc.[34]

3.4.2.2 Les avantages et inconvénients RNN

1.Avantages :

— Contrairement à un réseau de neurones traditionnel, un RNN partage le même paramètre à toutes les étapes. Cela réduit considérablement le nombre de paramètre à apprendre.

— Les RNN peuvent être utilisé avec les CNN pour générer des descriptions précises d'image non étiquetées.

2.Inconvénient :

— Le temps de calcul est long.

— Difficulté d'accéder à des informations d'un passé lointain.

3.4.2.3 Long short-term memory networks (LSTM)

Les réseaux de mémoire à long terme à court terme généralement appelés simplement (LSTM : Long Short Term Memory) sont un type spécial de RNN. Ils ont été introduits par Hochreiter Schmidhuber (1997). Les Réseaux neuronaux récurrents présentés dans la section précédente sont capables d'apprendre des règles de mise à jour de séquence arbitraire en théorie. Dans la pratique, cependant, ces modèles oublient généralement rapidement le passé. C'est ce qu'on appelle le problème de la disparition de gradient [et c'est pourquoi ils ont inventé le LSTM. La cellule LSTM est une adaptation de la couche récurrente qui permet aux signaux plus anciens des couches profondes de se déplacer vers la cellule du présent.[25]

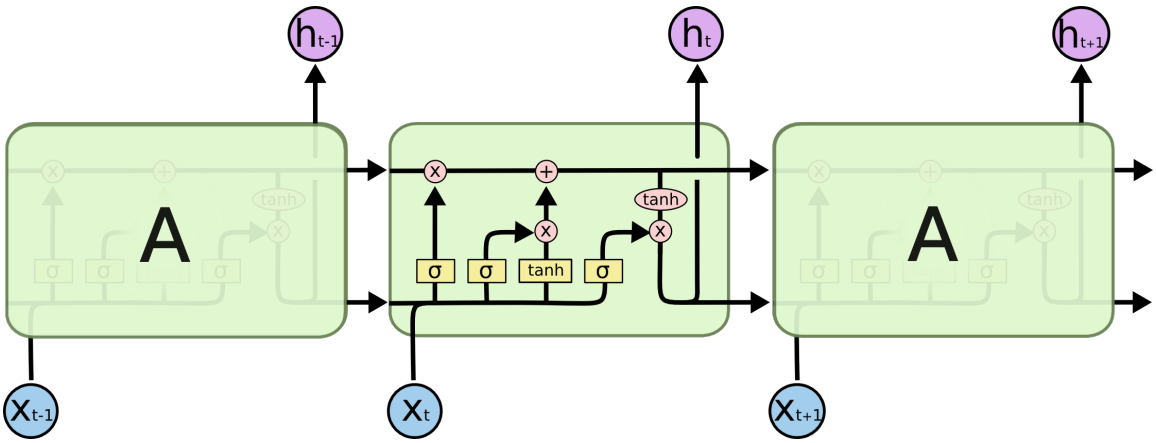
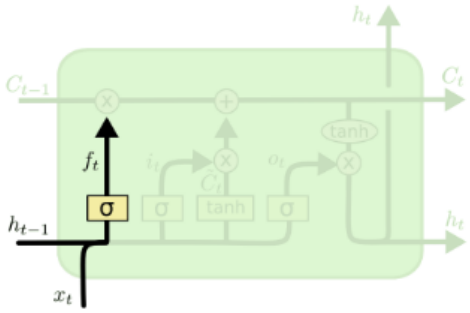


FIGURE 3.8: Une chaîne de cellules LSTM

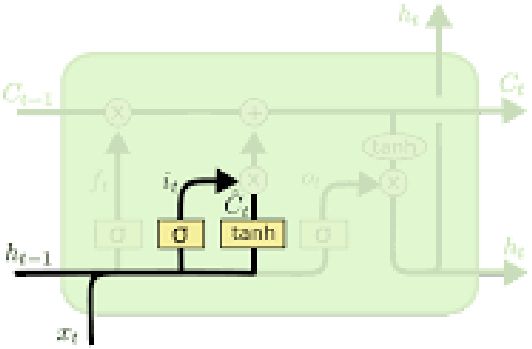
Les mathématiques derrière réseau LSTM : La porte d'oubli est la première étape par laquelle les informations qui seront exclues de la cellule sont déterminées. La fonction prend Ht-1(sortie de la couche précédente) et Xt (entrée actuelle), il produit nombre compris entre 0 et 1 et utilisé dans cette cas fonction sigmoïde, ou 1 signifie « garder complétement » et 0 « vider complètement » dans l'équation.[34]



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

FIGURE 3.9: La porte d'oubli d'une celle LTSM (Forget Get)

L´étape suivante consiste à décider quelles nouvelles informations nous allons stocker dans l'état de la cellule, Tout d'abord, une couche sigmoïde appelée "couche de la porte d'entrée" décide quelles valeurs nous allons mettre à jour. Ensuite, une couche tanh crée un vecteur de nouvelles valeurs candidates, Ct, qui pourraient être ajoutées à l'état. Dans l'étape suivante, nous allons combiner ces deux pour créer une mise à jour de l'état.[34]

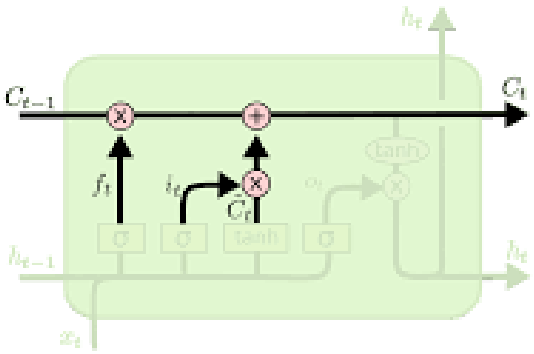


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

FIGURE 3.10: La porte d'entrée d'une celle LTSM (Input Get)

Il est maintenant temps de mettre à jour l'ancien état de cellule Ct-1 dans le nouvel état de cellule Ct sous le frome de l'équation. Notez que la porte oubliée ft peut contrôler le passage du gradient et permettre des suppressions et des mises à jour « en mémoire » explicites, ce qui permet de réduire le problème de gradient en cours de disparition ou

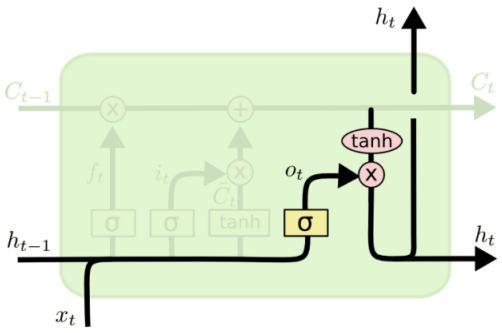
d'explosion de gradient dans un RNN standard.[34]



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

FIGURE 3.11: Mis à jour à l'état de la celle LTSM

Enfin, nous devons décider de ce que nous allons produire. Cette sortie sera basée sur notre état de cellule, mais sera une version filtrée. Tout d'abord, nous exécutons une couche sigmoïde qui détermine les parties de l'état de la cellule que nous allons produire. Ensuite, nous mettons l'état de la cellule 'a travers tanh (pour pousser les valeurs entre -1 et 1) et nous le multiplions par la sortie de la porte sigmoïde, de sorte que nous ne produisons que les parties que nous avons décidées.[34]



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

FIGURE 3.12: La porte sortie d'une cellule LSTM (Output Get)

3.5 Domaine d'application l'apprentissage en profonde

Ces techniques se développent dans le domaine de l'informatique appliquée aux NTIC (reconnaissance visuelle — par exemple d'un panneau de signalisation par un robot ou une voiture autonome — et vocale notamment) à la robotique, à la bio-informatique, la reconnaissance ou comparaison de formes, la sécurité, la santé, etc. . . , la pédagogie assistée par l'informatique, et plus généralement à l'intelligence artificielle.

L'apprentissage profond peut par exemple permettre à un ordinateur de mieux reconnaitre des objets hautement déformables et/ou analyser par exemple les émotions révélées par un visage photographié ou filmé, ou analyser les mouvements et position des doigts d'une main, ce qui peut être utile pour traduire le langage des signes, améliorer le positionnement automatique d'une caméra, etc. . . Elles sont utilisées pour certaines formes d'aide au diagnostic médical (ex. :reconnaissance automatique d'un cancer en imagerie médicale), ou de prospective ou de prédiction (ex. : prédiction des propriétés d'un sol filmé par un robot).[14]

3.6 Les plus et les moins du d'apprentissage profond

3.6.1 Les points forts de l'apprentissage en profondeur

— De meilleurs résultats qu'avec d'autres méthodes d'apprentissage machine.
— Une exécution efficace des tâches de routine, sans écarts de qualité.
— Le traitement des données non structurées.

3.6.2 Les points faibles de l'apprentissage en profondeur

— Le Deep Learning nécessite une grande puissance de calcul.
— Une technologie couteuse à mettre en place.
— Il nécessite une vaste base de données

3.7 Concusion

L'apprentissage profond est un domaine intéressant, où il est largement utilisé par les grandes entreprises et les grandes firmes pour avoir des bons résultats, et pour avoir des solutions aux problèmes complexes comme la création des véhicules autonomes, reconnaissance faciale...etc.

L'apprentissage profond est un peu complexe et il a besoin d'une grande masse de donnée, et des machines d'haute performance pour faire les calcules dans les meilleurs délais, comme les clusters ou l'utilisation de Cloud qui un peu cher.

Chapitre 4

Conception de Système

4.1 Introduction

Notre objectif est de réaliser un système fait extraction d'opinion à partir de sources textuelles sur de grandes quantités, qui déterminent si le sentiment dégagé par une phrase est positif ou négatif, le sentiment dégagé par une phrase dépend par une phrase dépend directement du contexte dans laquelle elle est utilisé.

Dans ce chapitre on va expliquer les étapes et les modules composant notre système, où nous présentons la conception de notre système en commençant par sa conception générale puis sa conception détaillée en expliquant les différents éléments du système et précisant leur fonctionnement.

4.2 Méthodologie suivie

Pour réaliser à notre système, nous avons appliqué une méthode supervisée de l'apprentissage profond, qui est le réseau de neurones récurrent, en anglais, Récurrent Neural Network (RNN), et nous avons choisi exactement la méthode de réseau récurrent à mémoire court et long terme, en anglais, Long Short-Term Memory Networks (LSTM).

Cette technique à besoin d'un grand corpus marque (base de données) et besoin d'une technique pour rendre ce corpus compréhensible pour la machine et pour cela nous avons utilisé bases des données de IMDB (Internet Movie Data Base) (2018 - 2017). Ensuit nous avons utilisé commentaires des sentiment positif et négative pour entrainer notre Word2Vec.

4.3 Conception globale du système

Dans notre système on a trois étapes principales avant de confirmer et utiliser le model : Le premier étape est collection des données de différent des sites web ou autre choses, les deuxième étape est préparation des données dans cette étape éliminer(les mots vides, ponctuation, les numéros ...) et aussi modifier dans la forme des mots par exemple(Steming, Lemmetazion ...), et ensuite on va entraîner notre système qui va essayer d'apprendre et de créer un modèle de catégorisation des sentiments à partir les données marquées, et ensuite il va sauvegarder le modèle, et si le modèle a une bonne précision on l'occupe sinon on refait l'entraînement avec d'autres paramètres

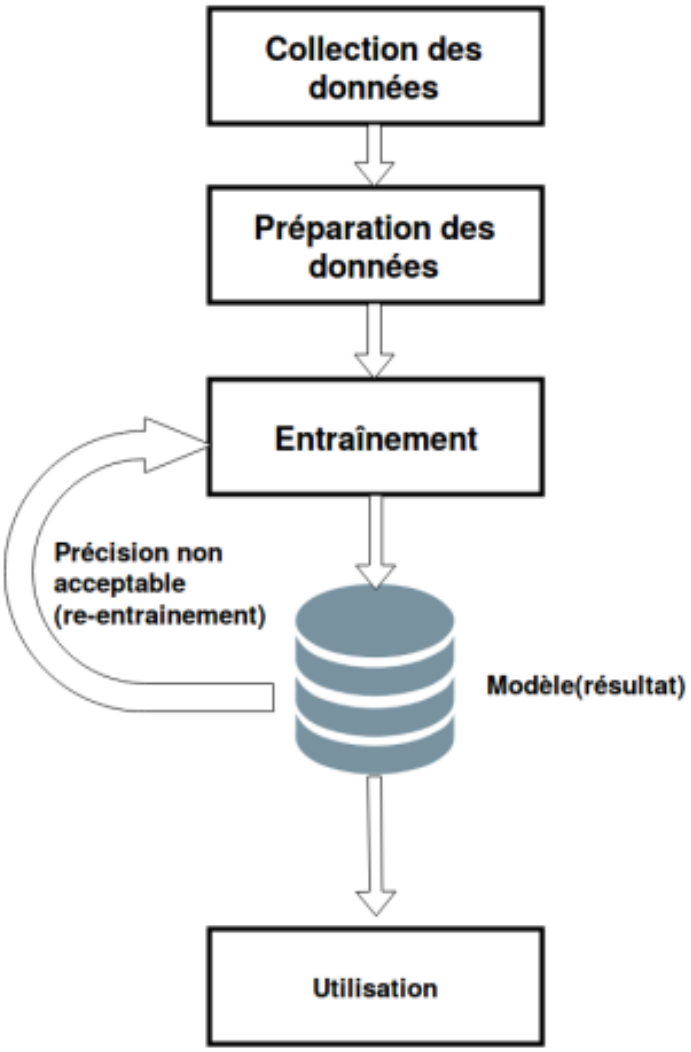


FIGURE 4.1: Conception globale du système

4.4 Conception détaillée du système

Dans cette partie, on va présenter séparément chaque partie du système proposé en détaillant et le principe des travaux, ou nous commençons par la collection des données puis la préparation des données textuelle et prétraitement, finalement le modèle pour l'utilisation.

4.4.1 Collection des données

Pour cette étape nous rassemblons les données sous forme CSV ou d'autres formes selon les besoins et les données sont collectées à partir de diverses sources et différent

exemple les réseaux sociaux (Facebook, Twitter, YouTube...) Et aussi les entreprises technologique et productive exemple (Google, Amazone, IMDB ...).

Dans ce travail nous appuierons sur ensembles des données contenant des avis et opinions exprimés en langue anglaise sur les films, ou nous baserons dans notre système sur un échantillon de ces données pour que chaque ligne contienne une phrase et une classe (négative ou positive) qui lui correspond.

4.4.2 Préparation des données

Cette étape vient après l'étape de collecte des données, à cette étape les données sont nettoyées et organisées selon les besoins. Cette étape est divisée en deux parties :

4.4.2.1 Prétraitement des données

Parmi l'étape de prétraitement les plus courantes, citons :la création de jetons, la suppression mot vides, l'accrochage, l'étiquetage des parties de parole, l'extraction et la présentation des caractéristiques.

- **La tokenisation :** est un processus de fractionnement ou de diviser le texte en petits block nommés jetons(token), nombre ou mots ou ponctuation et autres peuvent être considérés comme des signes.
- **Les Mots vides :** Les mots courants existant dans le texte réduisent les performances et n'apportent aucune valeur supplémentaire dans le sens, il est donc nécessaire de les supprimer, ces mots sont "à", "at","the" et ils sont appelés mots vides .
- **Stemming :** est une façon très simple de réduire un mot à sa racine, sa base ou sa racine, en identifiant son préfixe et en le supprimant, ce qui est important car cela réduit la taille du vocabulaire et augmente les performances, par exemple (computers = computer, wallked = walk).
- **Lemmatization :** comme le stemming essaie également de réduire les mots à une forme de base, mais il suit des approches différentes au lieu de les dépouiller, il utilise une connaissance lexicale pour obtenir la forme de base des mots.
- **Séparateurs de ponctuation :** Le processus de suppression des Séparateurs de ponctuation fait partie des choses importantes dans le traitement de texte car il ajoute un ajout au langage humain mais n'ajoute pas d'ajout à l'ordinateur, il est donc

très important de le supprimer afin de faciliter le traitement de texte.
Le formulaire suivant est pour une meilleure compréhension du processus :

text	lang	text_clean
Will Smith Joins Diplo And Nicky Jam For The 2...	en	smith join diplo nicky jam 2018 world cup offi...
Hugh Grant Marries For The First Time At Age 57	en	hugh grant marries first time age 57
Jim Carrey Blasts 'Castrato' Adam Schiff And D...	en	jim carrey blast castrato adam schiff democrat...
Julianna Margulies Uses Donald Trump Poop Bags...	en	julianna margulies us donald trump poop bag pi...
Morgan Freeman 'Devastated' That Sexual Harass...	en	morgan freeman devastated sexual harassment cl...

FIGURE 4.2: Un exemple de prétraitement des données

4.4.2.2 Marquage des données :

Pour le marquage des données on va déplacer chaque commentaire vers un dossier, où chaque dossier représente un label, et nous on a deux dossier :(négatif, positif) si le texte contient des sentiments négatifs on le déplace au dossier "négatifs" et s'il contient des sentiments positifs on le déplace au dossier "positifs", et si le texte est incompréhensible on va les éliminer.

4.4.3 Entraînement

4.4.3.1 Word2Vec

est une technique ou un ensemble de modèles utilisés pour produire ce que l'on appelle l'intégration de mots qui sont des représentations de mots sous une forme ou de grands vecteurs généralement de plusieurs centaines de tailles, word2vec nécessite un grand corpus de texte d'entrée pour construire ces représentations, avec l'avantage de garder le contexte du mot intact et le mot proche ont des représentations vectorielles fondamentalement similaires, techniquement word2vec est un modèle de réseau neuronal, ou spécifiquement un réseau neuronal peu profond une fois qu'il est formé, il peut effectuer cette tâche de vectorisation.

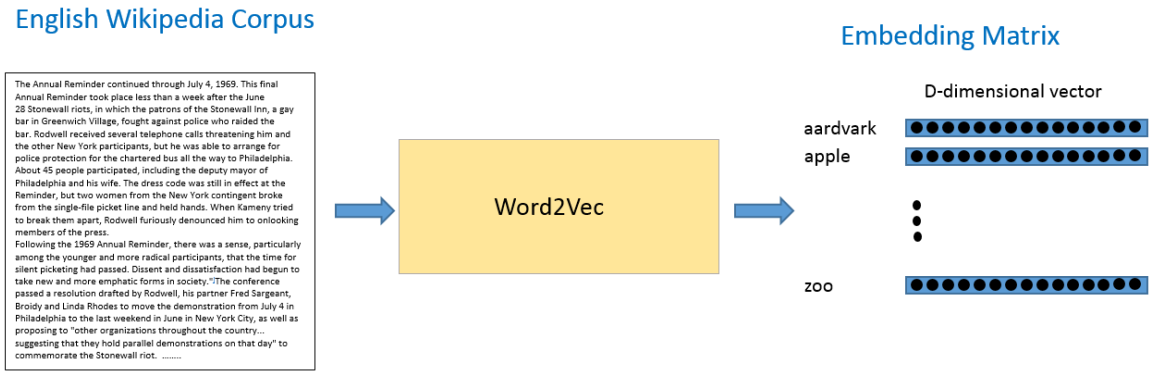


FIGURE 4.3: Entraînement de Word2Vec

4.4.3.2 Principe de fonctionnement pour Word2Vec

- La tâche principale de l'outil word2vec est de faire un regroupement de matrices de mots similaires, similaires et apparentés ensemble, ce qui se fait à travers les similitudes mathématiques de chaque mot. Ces similitudes et analogies sont comme (homme - garçon) == (femme - fille)
- Il est entraîné sur la base de l'inclusion de mots, et son objectif est de calculer l'importance et la valeur de chaque mot de la phrase, Et puis, on en déduit le reste du mot.
- Elle saura aussi que ce mot est au singulier et que celui-ci est au pluriel, ce qui permet par la suite de faire plus facilement une formulation complète des textes et de savoir s'il est censé être utilisé au singulier ou au pluriel, et ainsi de suite.
- Aussi l'outil word2vec, lorsqu'il prend une grande quantité de données, il a la capacité de prédire le sens des mots, en fonction de leur emplacement et de leur contexte

Le modèle word2vec peut être implémenté à l'aide de deux méthodes, ces méthodes sont :

Modèle CBOW : Dans cette technique prédire le mot correspondant à notre contexte, se fait par le contexte de chaque mot en entrée, prenons cet exemple "c'est une belle fleur" donc si on a le mot "belle" en entrée dans le réseau de neurones et on essaie prédire le mot fleur.

Ensuite, nous représentons spécifiquement notre mot d'entrée comme un vecteur et

mesurons et comparons l'erreur de sortie à ce mot cible codé, au fur et à mesure que le processus est terminé, nous obtenons ensuite la représentation du mot vectoriel de notre mot cible.[26]

Modèle Skip-Gram : Nous pouvons remarquer que dans Skip-Gram, il retourne le multi-CBOW qui est vrai dans une certaine mesure, l'entrée est un mot cible dans le réseau, et nous obtenons une distribution de probabilité et notablement.[26]

Les deux modèles utilisent la rétro-propagation pour le processus d'apprentissage, le formulaire suivant montre la différence entre CBOW et Skip-Gram :

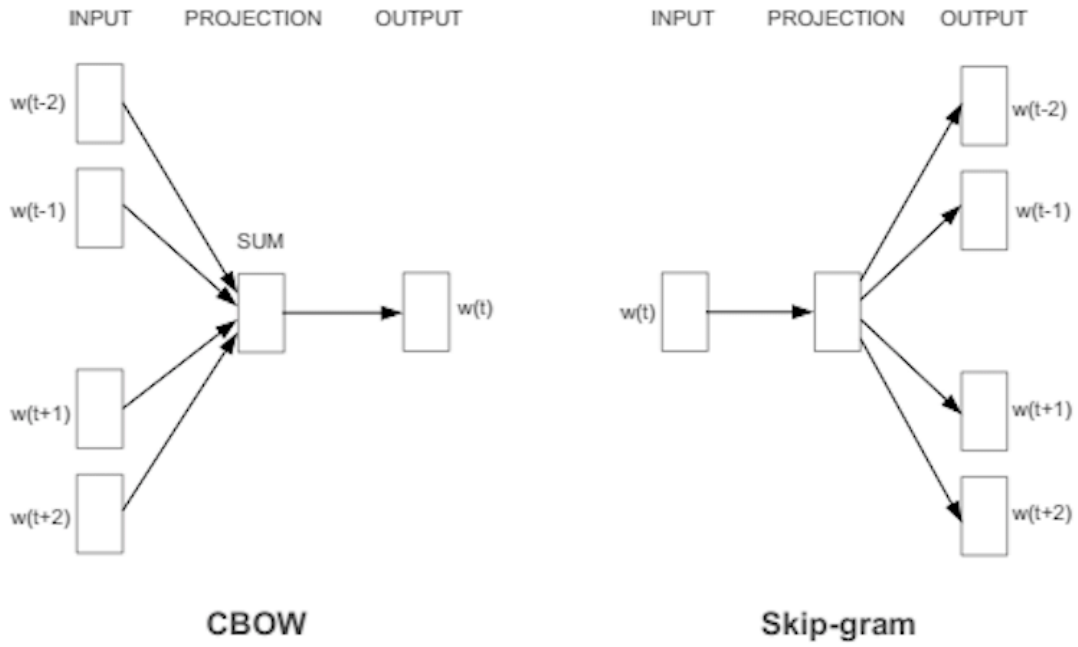


FIGURE 4.4: Modèle CBOW et Skip-Gram

4.4.3.3 Entraînement du modèle de catégorisation des sentiments

Pour un bon apprentissage du modèle, les données doivent d'abord être transformées par la librairie Numpy qui est créée et traitée dans word2vec ,C'est un tableau avec des dimensions [taille des données * longueur de séquence maximale] ou chaque ligne de matrice contient des index de chaque mot dans les vecteurs des mots car chaque mot a un vecteur de 300 dimensions, et chaque mot sera traité dans une cellule LSTM(Long short-term memory, en Français : réseau récurrent à mémoire court et long terme) , Après

avoir terminé la formation, nous vérifions la préparation et la validité du modèle. Si le modèle est acceptable et atteint ce qui est requis dans ce cas, nous pouvons utiliser ce modèle sinon on refait l'entraînements avec d'autres paramètres.

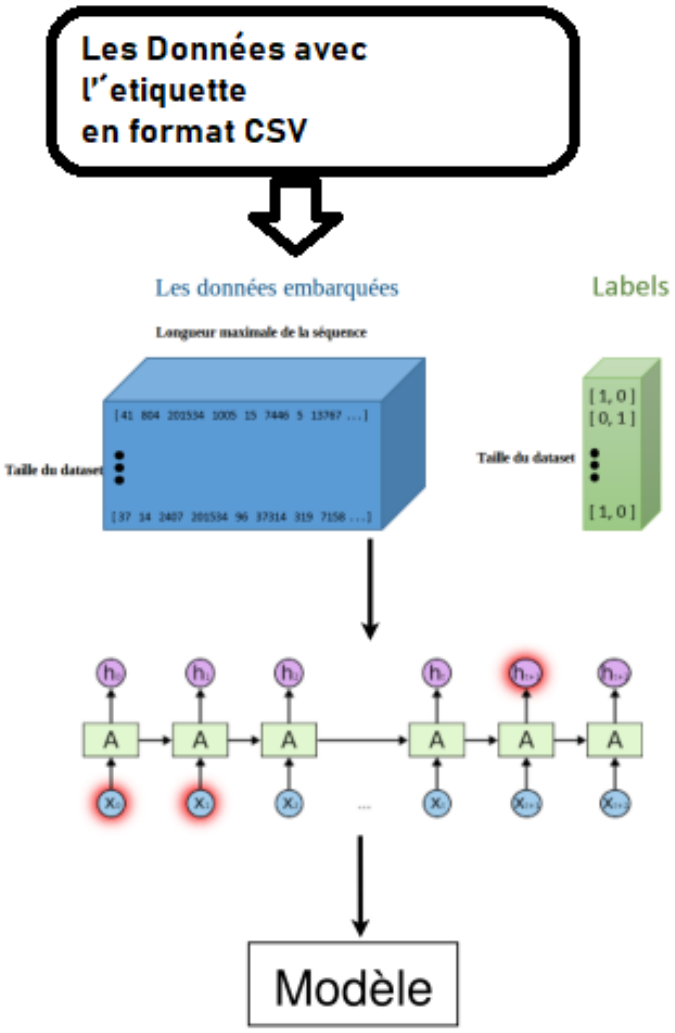


FIGURE 4.5: Entraînement du Modèle

Pour entraîner le modèle on est besoin aussi d'une fonction qui nous retourne (de commentaires) – en anglais : simple- avec un nombre d'échantillons, car on ne peut pas transmettre toutes les données dans un réseau de neurones en même temps, et ce lot sera transmis avec ces (l'étiquettes de chaque item : positif ou négatif) comme montré dans l'algorithme au dessus :

Algorithm 1 Algorithme d'entraînement du modèle de catégorisation des sentiments

```
matriceIds ← EmbarqueDonnees(dataset, vecteurMots,vecteurRéelles)
for i <iterations do
    lot, labels ← avoirLot(matriceIds)
    entrainerModele(lot,labels, nombreLSTM,longeurMax)
sauvegarderModele()
```

FIGURE 4.6: Algorithme d'entraînement du modèle de catégoration des sentiments

4.4.4 Teste du modèle

Pour le tester, on calcule la précision sur un corpus de test jamais vue par le modèle, et sila précision est élevée (plus de 70%) on occupe le modèle sinon, on refait le traitement avec d'autres hyperparamètres, où on essaie de perfectionner notre Word2vec par le réentraîner, et aussi de modifier les hyperparamètres.

4.4.5 Utilisation du modèle

Après avoir terminé le processus de collecte de données et également préparé et formé les données, et après avoir testé le modèle à la fin, nous avons un modèle utilisable, c'est-à-dire la prédiction, afin de l'utiliser pour prédire de nouveaux textes, et le résultat de ces textes est soit positif, soit négatif. La figure suivante montre comment utiliser le modèle :

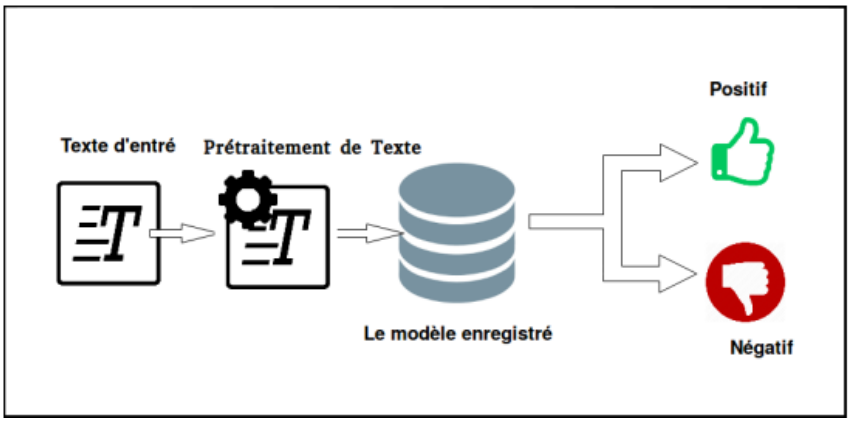


FIGURE 4.7: Utilisation du modèle

4.5 Conclusion

Dans ce chapitre, nous avons présenté le processus utilisé, de notre système ont été présentés avec une explication détaillée pour chaque étape du processus d'application, en commençant par l'étape de collecte, en passant par l'étape de prétraitement et se terminant par l'étape d'entraînement. Et dans le chapitre suivant, nous allons d´écrive l'implémentation de notre système.

Chapitre 5

Implémentation

5.1 Introduction

Dans ce chapitre, nous allons présenter l'environnement de travail, le langage de programmation, et les outils que nous avons utilisés pour construire le système. Par la suite nous allons expliquer toutes les expérimentations que nous avons appliquées sur la méthode proposée et les résultats obtenus.

5.2 Environnement et outils de développement

Pour développer notre système et valider notre proposition, nous avons utilisé le langage de programmation Python et l'environnement Pycharm pour écrire les programmes. Pour la collection des données nous avons utilisé différent site web (Kagle , Amazon ,IMBD ...) pour rassembler les commentaires. Comme nous avons utilisé de nombreuse bibliothèque.

5.2.1 Environnement de développement

5.2.1.1 Python

Python est un langage de programmation de haut niveau a été créé en 1989 par Guido van Rossum, aux Pays-Bas La première version publique de ce langage a été publiée en 1991, Ce langage de programmation présente de nombreuses caractéristiques intéressantes .[17]

5.2.1.2 Google Colab

Google Colab ou Colaboratory est un service cloud, offert par Google (gratuit), basé sur Jupyter Notebook et destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud.[32]

5.2.1.3 PyCharm

PyCharm est un environnement de développement intégré (IDE) utilisé dans la programmation informatique, spécifiquement pour le langage Python. Il est d´développé par

la société tchèque JetBrains. Il fournit une analyse de code, un débogueur graphique, un testeur d'unité intégrée, l'intégration avec des systèmes de contrôle de version (VCS), et prend en charge le développement web avec Django. PyCharm est multi-plateforme, avec les versions Windows, MacOs et Linux. L'édition de communauté est libérée sous la licence d'Apache, et il y a également l'édition professionnelle libérée sous une licence de propriétaire.[31]

5.2.1.4 Jupyter Notebook

Jupyter Notebook est un projet open source dérivé d'python et fournit une interface web riche dans le cadre d'une programmation interactive. Agnostique vis-à-vis des langages de programmation, l'application écrite en HTML trouve son utilité dans le traitement interactif des données scientifiques, et fonctionne souvent de pair avec Python pour Windows, Mac, Linux.[1]

5.2.1.5 Anaconda

Anaconda distribution open source est le moyen le plus simple d'exercice la science des données Python/R et l'apprentissage automatique sur linux, Windows, et Mac OS X. avec plus 11 millions d'utilisateurs dans le monde entier, il s'agit du standard de l'industrie pour le développement, les tests et la formation. Une seule machine, anaconda est une distribution python et R. il vise tout ce dont vous avez besoin (en python) pour la science des données, AI, Machine Learning, Deep Learning.[2]

5.2.2 Les outils utilisés

5.2.2.1 TensorFlow

TensorFlow est une bibliothèque logicielle open source pour le calcul numérique haute performance. Son architecture flexible permet un déploiement facile du calcul sur une variété de plates-formes (CPU, GPU, TPU), et des ordinateurs de bureau aux clusters de serveurs aux périphériques mobiles et périphériques. Développé à l'origine par des chercheurs et des ingénieurs de l'équipe Google Brain au sein de l'organisation AI de Google, il bénéficie d'un fort soutien pour l'apprentissage automatique et l'apprentissage en profondeur et le calcul numérique flexible est utilisé dans de nombreux autres domaines

scientifiques. TensorFlow a été développé pour une utilisation interne de Google. Et après il a été publié sous licence open source Apache 2.0 le 9 novembre 2015.[4]

5.2.2.2 NumPy

Le module NumPy est la boîte à outils indispensable pour faire du calcul scientifique avec Python, Pour modéliser les vecteurs, matrices, et plus généralement les tableaux à n dimensions, numpy fournit le type ndarray. Il y a des différences majeures avec les listes (resp. Les listes de listes) qui pourraient elles aussi nous servir à représenter des vecteurs (resp. Des matrices).[20]

5.2.2.3 Genism

Gensim est un outil robuste de modélisation de l'espace vectoriel open-source et de modélisation de sujet implémente en Python. Il utilise NumPy, SciPy et éventuellement Cython pour les performances. Gensim est spécialement conçu pour gérer de grandes collections de textes, en utilisant le streaming de données et des algorithmes incrémentaux efficaces, ce qui le différencie de la plupart des autres logiciels scientifiques qui ne ciblent que le traitement par lot et en mémoire.[37]

5.2.2.4 NLTK

NLTK est une plate-forme de premier plan pour la création de programmes Python fonctionnant avec des données en langage humain. Il fournit des interfaces faciles à utiliser vers plus de 50 corpus et ressources lexicales telles que WordNet, ainsi qu'une suite de bibliothèques de traitement de texte pour la classification, la tokenisation, le radicalisme, le balisage, l'analyse et le raisonnement sémantique, des wrappers pour les bibliothèques NLP de puissance industrielle, et un forum de discussion actif.[27]

5.2.2.5 Keras

Keras est une bibliothèque open source (licence MIT) écrite en Python qui est principalement basée sur le travail effectué par le développeur Google François Chollet dans le cadre du projet ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System). La première version de ce logiciel indépendant de la plateforme a été publiée le

28 mars 2015. L'objectif de cette bibliothèque est de permettre le développement rapide de réseaux de neurones. Dans ce cas, Keras n'est pas un Framework séparé mais une interface conviviale pour les débutants (API) pour accéder et programmer une variété de Framework d'apprentissage automatique. Theano, Microsoft Cognitive Toolkit (anciennement CNTK) et TensorFlow font partie des Framework pris en charge par Keras.[40]

5.2.2.6 Matplotlib

Matplotlib est une bibliothèque de traçage pour le langage de programmation python et son extension de mathématiques numériques Numpy, il facilite l'intégration de tracés dans des applications à l'aide d'outils d'interface graphique généraux tels que tkinter Pyqt, etc. avec une approche orientée objet.[3]

5.2.2.7 Flask

Flask est un Framework web. Cela signifie que Flask vous fournit des outils, des bibliothèques et des technologies qui vous permettent de créer une application Web. Cette application Web peut être constituée de pages Web, d'un blog, d'un wiki ou de la taille d'une application de calendrier Web ou d'un site Web commercial.[5]

5.3 Interface d'analyse des sentiments

Dans cette interface est montré le test de ce modèle LSTM, on a dans cette interface de zones de texte, dans le premier on écrit la phrase à laisser ensuite on click sur le bouton « Envoyer » et le résultat (Positif, Négatif) affiche dans la zone sentiment et probabilité.

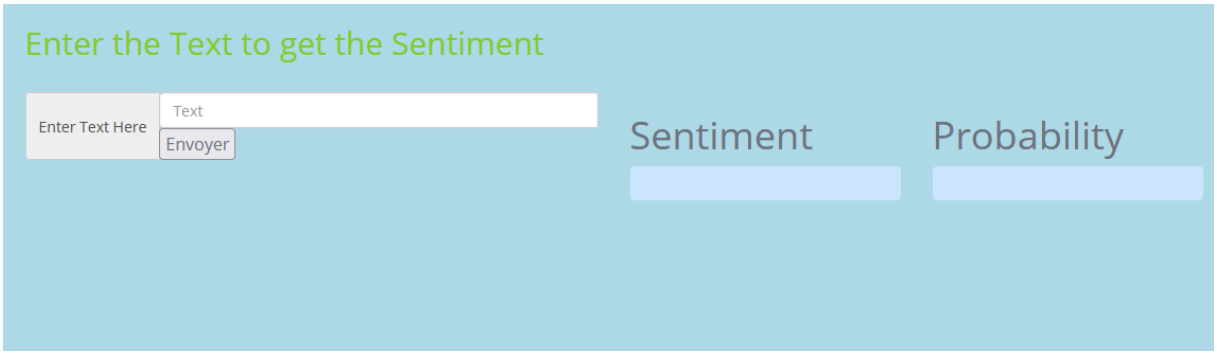


FIGURE 5.1: Interface analyse sentiment

5.3.1 Scénario d'utilisation simple

En tant que simple de notre application, nous choisissons de fournir certains commentaires pour un spécifique, puis nous soumettrons et verrons le résultat renvoyé, dans la figure suivante, nous avons fourni les critiques suivantes "good movie and good actors is the best film in this year" le résultat obtenu « POSITIF » avec un score de 98%.

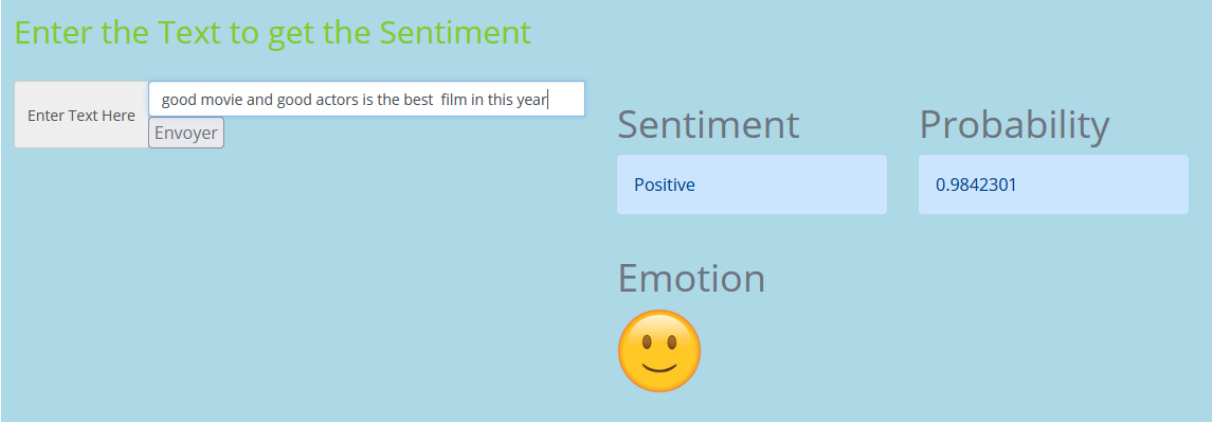


FIGURE 5.2: Positive Commentaire

D'autre part, dans la figure suivante, nous avons soumis le commentaire suivant "is verry bad movie and bad actors" et avons été correctement classés comme NÉGATIF avec un score de 11%.

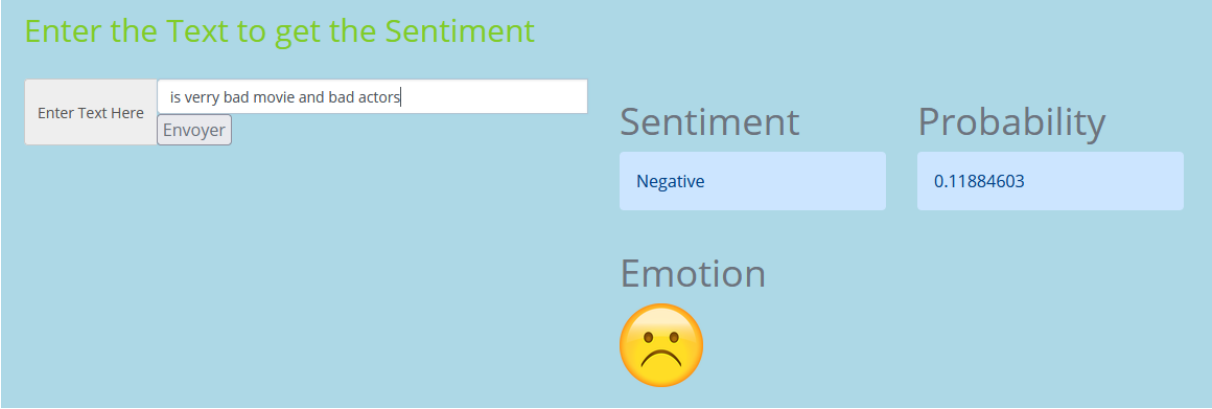


FIGURE 5.3: Négatif Commentaire

5.4 préparation des données collectées

Comme discuté dans le chapitre précédent, nous avons collecté 3 ensembles de données sur lesquels travailler, tous les 3 ont des critiques textuelles qui sont étiquetées ”positives” ou ”négatives” selon leur sentiment.

5.4.1 Prétraitement des données

Ensuite nous allons prétraiter ou nettoyer nos ensembles de données par tokenzation et supprimer les mots vides etc.. . . puis nous les combinons en un seul fichier, la fonction suivante est utilisée pour prétraiter les données.

```
def load_dataset():
  df = pd.read_csv(x)
  x_data = df['text']
  y_data = df['label']

  #Removing notion HTML
  x_data = x_data.replace({'</br> <br>',''},regex=True)
  x_data = x_data.replace({'[A-Za-z]',''},regex=True)

  #Supprimer le mot vide
  x_data = x_data.apply(lambda text:[w for w in text.split() if w not in english_stops])

  #Supprimer la ponctuation
  x_data = x_data.apply(lambda text:[c for c in text if c not in string.punctuation])


  x_data = x_data.apply(lambda text:[c for c in text if c not in string.digits])

  #Convertir le texte en minuscule
  x_data = x_data.apply(lambda text:[w.lower() for w in text])

  return x_data , y_data

def remove_punct(txt):
    txt_nopunct = "".join([c for c in txt if c not in string.punctuation])
    return txt_nopunct

def remove_stop_word(txt):
    txt_stop_word = "".join([c for c in txt if c not in english_stops])
    return txt_stop_word
```

FIGURE 5.4: fonction de nettoyage des données

5.4.2 statistiques d'ensemble de données combinées

La figure suivante montre les statistiques pour les données agrégées finales, où le nombre 1 représente les commentaires positifs et le nombre 0 représente les commentaires négatifs et comme nous pouvons le voir dans notre graphique, nous avons 35 014 commentaires négatifs et 34 985 commentaires positifs.

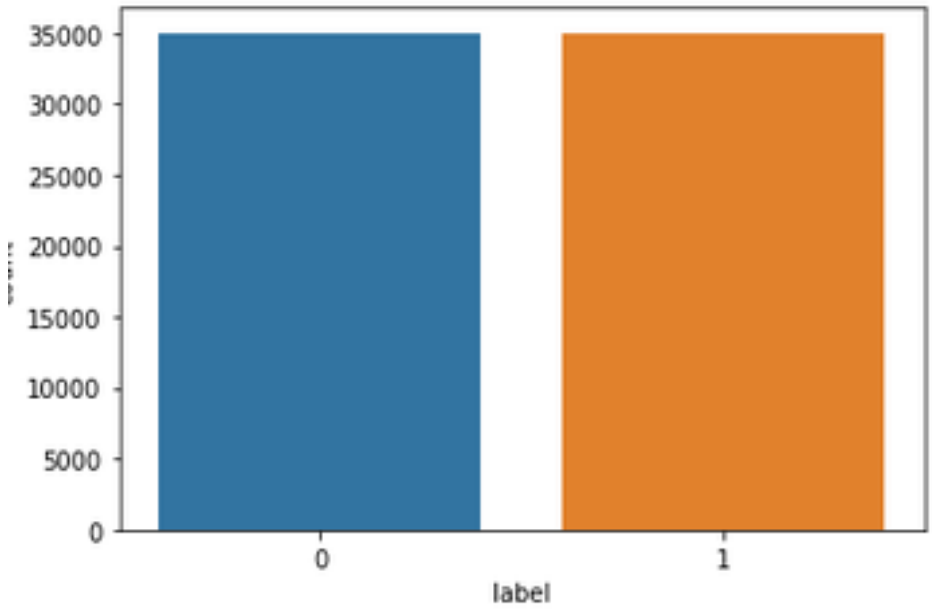


FIGURE 5.5: statistiques des commentaires positifs et négatifs

5.4.3 fractionnements des données

Enfin, nous avons divisé nos données en 80 % de données de train et 20% de données de test comme indiqué dans le script suivant :

5.5 utiliser RNN LSTM

5.5.1 vectorisations des données

Dans cette étape, nous allons transformer nos données en digestibles à partir du classificateur, un vectoriseur word2vec comme discuté précédemment est l'un des meilleurs vectoriseurs, nous utilisons donc et le prochain script dans la figure utilisant la bibliothèque Gensim entraînera et construira notre vectoriseur. l'étape ci-dessus, construit le

```
x_train,x_test,y_train,y_test = train_test_split(x_data,
                                                 y_data,
                                                 test_size=0.2 )
print('Data form:')
print(data.shape)
print("Train set:")
print(x_train.shape)
print("Test set:")
print(x_test.shape,)
```

```
Data form:
(69999, 2)
Train set:
(55999,)
Test set:
(14000,)
```

FIGURE 5.6: fractionnement du script de données

```
w2v_model = gensim.models.word2vec.Word2Vec(size=300 ,
                                            window=7 ,
                                            min_count=10 ,
                                            workers=8)
```

```
[19] w2v_model.build_vocab(x_data)
```

```
[21] w2v_model.train(x_data,total_examples=len(x_data),epochs=10)
```

```
(81545573, 93055100)
```

FIGURE 5.7: vectoriseur word2vec

vocabulaire et commence à former le modèle word2vec, word2vec est un réseau de neurones peu profond avec une seule couche cachée, et ce qui se passe dans les coulisses, c'est que nous entraînons ce réseau neuronal pour pouvoir prédire un mot basé sur son contexte, mais dans ce cas, nous allons abandonner le réseau de neurones et ne pas l'utiliser une fois le processus de formation terminé, nous allons plutôt apprendre le poids de la couche cachée, ce poids représente les vecteurs de mots dont nous avons besoin et que nous avons essayé d'apprendre , un autre nom de ces vecteurs est l'intégration.

5.5.2 tester Wor2vec

Le code suivant montre un résultat lors du test de similitude des mots

```
w2v_model.wv.most_similar("bad")
```

```
[('bad,', 0.5901995897293091),
 ('bad.', 0.5198606252670288),
 ('awful', 0.48957359790802),
 ('good', 0.4894154667854309),
 ('terrible', 0.48447108268737793),
 ('horrible', 0.4675818085670471),
 ('horrible.', 0.43769457936286926),
 ('crappy', 0.43454110622406006),
 ('stupid', 0.43401932716369 63),
 ('bad!', 0.4312223494052887)]
```

```
[28] w2v_model.wv.most_similar("good")
```

```
[('decent', 0.5910717248916626),
 ('good,', 0.5454968214035034),
 ('great', 0.5156739354133606),
 ('good.', 0.5051781535148621),
 ('bad', 0.4894154667854309),
 ('nice', 0.4525182247161865),
 ('fine', 0.4220234155654907),
 ('excellent', 0.4188818633556366),
 ('ok', 0.4181022346019745),
 ('lousy', 0.400378942489624)]
```

FIGURE 5.8: test Word2vec

5.5.3 construction et entraînement RNN LSTM

Définition de l'architecture du réseau LSTM :

— Embedding layer :une couche qui convertit notre mot jeton en intégration.

— Dropout Layer :nous avons utilisé une couche d'abandon qui supprime ou ignore les unités (dans les neurones) pendant la phase d'apprentissage de certains ensembles de neurones qui sont choisis au hasard, ceci est fait pour éviter le surapprentissage.

— LSTM Layer :(the long short term memory layer).

— Dense Layer une couche de réseau de neurones profondément connectée avec une fonction d'activation sigmoïde.

```
model = Sequential()
model.add(embedding_layer)
model.add(Dropout(0.5))
model.add(LSTM(100,dropout=0.2,recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))
model.compile( optimizer = "adam" ,loss = "binary_crossentropy",metrics=['accuracy'])
print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 134, 300)          104796300
_________________________________________________________________
dropout (Dropout)            (None, 134, 300)          0
_________________________________________________________________
lstm (LSTM)                  (None, 100)               160400
_________________________________________________________________
dense (Dense)                (None, 1)                 101
=================================================================
Total params: 104,956,801
Trainable params: 160,501
Non-trainable params: 104,796,300
_________________________________________________________________
None
```

FIGURE 5.9: vectoriseur word2vec

L'étape suivante consiste à entraîner le réseau LSTM à travers 10 époques. La figure montre le processus d'entraînement.

```
history =model.fit(x_train,y_train, epochs=10,batch_size=128 , validation_data=(x_test,y_test))
```

```
Epoch 1/10
438/438 [==============================] - 419s 957ms/step - loss: 0.2349 - accuracy: 0.9022 - val_loss: 0.2598 - val_accuracy: 0.8962
Epoch 2/10
438/438 [==============================] - 419s 957ms/step - loss: 0.2351 - accuracy: 0.9022 - val_loss: 0.2545 - val_accuracy: 0.8997
Epoch 3/10
438/438 [==============================] - 419s 956ms/step - loss: 0.2339 - accuracy: 0.9018 - val_loss: 0.2494 - val_accuracy: 0.8986
Epoch 4/10
438/438 [==============================] - 420s 959ms/step - loss: 0.2290 - accuracy: 0.9050 - val_loss: 0.2540 - val_accuracy: 0.8980
Epoch 5/10
438/438 [==============================] - 418s 955ms/step - loss: 0.2297 - accuracy: 0.9048 - val_loss: 0.2607 - val_accuracy: 0.8974
Epoch 6/10
438/438 [==============================] - 418s 954ms/step - loss: 0.2310 - accuracy: 0.9035 - val_loss: 0.2575 - val_accuracy: 0.8962
Epoch 7/10
438/438 [==============================] - 418s 954ms/step - loss: 0.2306 - accuracy: 0.9032 - val_loss: 0.2527 - val_accuracy: 0.8979
Epoch 8/10
438/438 [==============================] - 418s 955ms/step - loss: 0.2289 - accuracy: 0.9052 - val_loss: 0.2484 - val_accuracy: 0.8992
Epoch 9/10
438/438 [==============================] - 418s 955ms/step - loss: 0.2286 - accuracy: 0.9065 - val_loss: 0.2581 - val_accuracy: 0.9001
Epoch 10/10
438/438 [==============================] - 419s 956ms/step - loss: 0.2284 - accuracy: 0.9053 - val_loss: 0.2612 - val_accuracy: 0.8976
```

FIGURE 5.10: Entrainement du modéle

5.5.4 Evaluation de modèle

Dans le tableau suivant, nous avons des résultats pour différentes tailles de données. Les résultats sont les suivants :

Expre	Data	Positif	Negatif	entraînement	test	précision	perte
1	5000	2505	2495	4000	1000	0.73	0.57
2	25198	12592	12606	20158	5040	0.82	0.41
3	32000	16015	15985	25600	6400	0.85	0.33
4	69999	34985	35014	5999	14000	0.90	0.27

TABLE 5.1: Exprérimentation

Après le processus d'expériences avec le changement du volume de données, il y avait des différences en termes de résultats à la fois en termes de précision et de perte. Nous avons également constaté que plus le pourcentage de données était élevé, plus la précision et la diminution de la perte étaient élevées, ce qui signifie que la quantité et la qualité des données ont un impact significatif sur l'exactitude des résultats et l'apprentissage en profondeur.

Aussi, parmi les choses que nous avons trouvées lors du processus d'expérimentation, le deep learning nécessite des appareils et des équipements à haute efficacité, afin de gagner du temps, car plus il y a de données, plus il faut de temps de traitement, et ce dernier consomme beaucoup de temps.

Au final, nous concluons que plus il y a de données, plus les résultats sont précis et plus l'efficacité des appareils est élevée, moins il y a de temps de traitement.

Le graphique suivant est les résultats des expériences, montrant la relation des données avec présicion et la perte des résultats

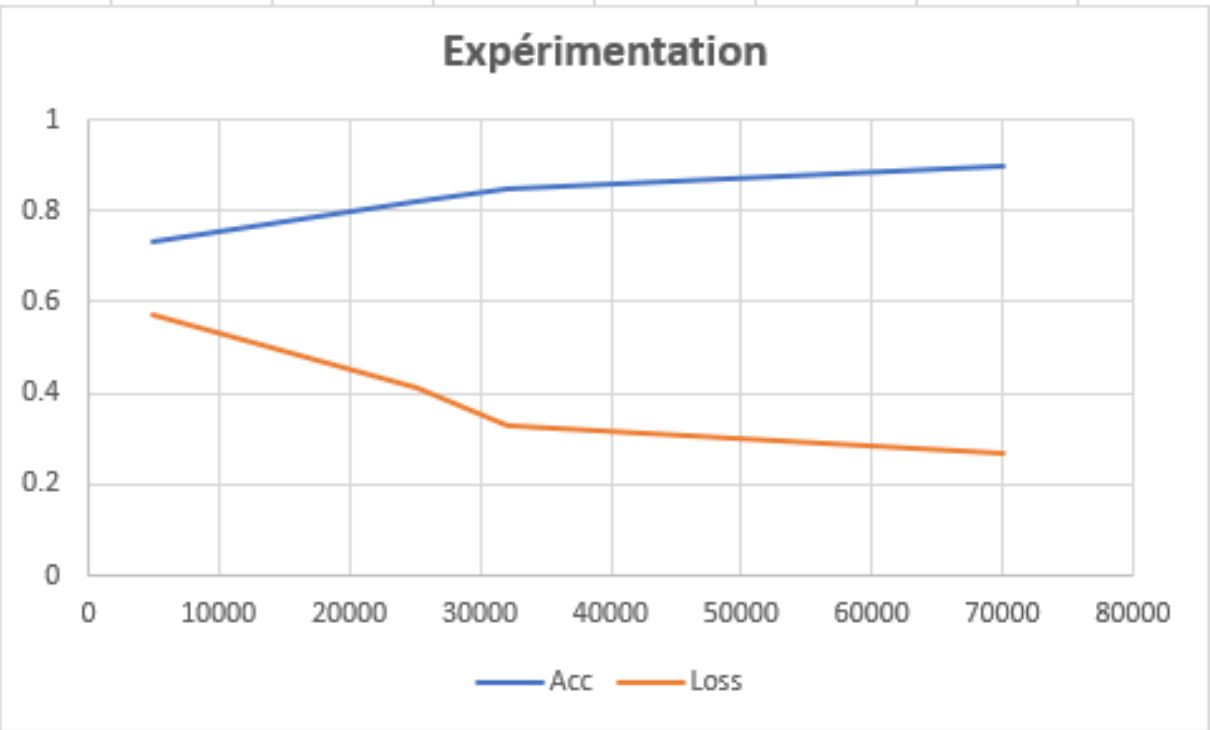


FIGURE 5.11: La relation entre les données et les résultats

5.6 lien pour notre travail

Afin de bien comprendre le travail et de mieux comprendre les étapes de traitement du système, nous avons partagé le lien sur le site Web de google colab pour faciliter le processus de participation et également rendre le processus de participation plus efficace.

Cliquez ici pour accéder au lien **colab.research.google.com**

5.7 Conclusion

Dans ce chapitre, nous avons appliqué l'application pratique de l'analyse des sentiments, car nous expliquons le travail étape par étape en utilisant l'apprentissage en profondeur LSTM, en commençant par expliquer les outils et les applications dont nous avons besoin, en passant par l'application de l'apprentissage en profondeur jusqu'à l'achèvement de la construction du application réelle de l'analyse des sentiments, Finalement nous avons expliqué les expérimentations et les résultats obtenues.

Chapitre 6

Conclusion générale

6.1 Conclusion générale

L'analyse des sentiments ou ce que l'on appelle l'extraction d'opinions est un domaine très important de l'IA et en particulier du domaine de la NLP, grâce à son importance pour les entreprises et la politique du monde réel et d'autres domaines, dans ce travail, nous avons essayé de développer un outil d'analyse de sentiments basée sur l'apprentissage en profondeur, ce qui fournit des performances élevées et des résultats précis.

Alors que de nouvelles technologies et de nouvelles recherches en NLP émergent et se développent, la nécessité d'étendre notre système d'analyse des sentiments est un impératif, compte tenu de son importance primordiale en termes de rapidité et de précision des résultats et d'assistance aux décideurs et d'autre part d'ambition et recherche d'amélioration continue dans divers domaines afin de fournir les meilleurs services aux autorités concernées, pour ajouter plus de travail à l'avenir, nous suggérons quelques points :

- étendre les modèles à l'arabe et à l'algérien ou à un autre dialecte.
- essayer de collecter plus de données et entraîner à nouveau notre modèle
- modifier notre système pour pouvoir prédire le sentiment « neutre »
- étendre notre système pour pouvoir détecter les sentiments dans les articles et les
- essayer de construire un système plus général pour analyser les sentiments pour n'importe quel contexte linguistique
- Étendre notre système pour pouvoir détecter d'autres sentiments tels que (peur, tristesse, enthousiasme, joie, douleur...etc.) pour donner plus de précision dans le contexte du texte.

Bibliographie

[1] de la société de logiciel. *https ://www.01net.com/telecharger/*.

[2] Site web anaconda. *https ://www.anaconda.com/distribution/*, 2021.

[3] Site web de matplotlib. *https ://matplotlib.org/*, 2021.

[4] Site web de tensorflow. *www.tensorflow.org*, 26/04/2018.

[5] Python flask tutorial. *Software Testing Help*, May 30, 2021.

[6] Medjdoubi Abdelkader. L'analyse du sentiment utilisant le deep learning. *Université Dr. TAHAR MOULAY SAIDA*, 2019.

[7] ABDELLI ADEL. Deep learning-based sentiment of analysis of algerian-arabic short texts. *Université Mohamed Khider – BISKRA*, 2018.

[8] ZIANI Amel. La recommandation via l'analyse d'opinions. *Université de Badji Mokhtar Annaba*, Année 2017/2018.

[9] Jason Brownlee. Macine learning mastery. *What's the July 2017 Conference : Data Scientist Innovation Data*, Nouvember 11 2019.

[10] Nikhil Buduma. . fundamentals of deep learning designing next-generation machine intelligence algorithms. 2017.

[11] convnet benchmarks. Convolutional neural networks for visual recognition. 2017.

[12] T Edward Damer. Attacking faulty reasoning. *Cengage Learning*, 2008.

[13] Axel de goursac. Natural language processing. *myriade*, 2017.

[14] Boughaba Mohammed et Boukhris Brahim. L'apprentissage profond (deep learning) pour la classification et la recherche d'images par le contenu. 2017.

[15] Soumia Elyakoute HERMA et Khadija SAIFA. thèses analyse des sentiments cas twitter. *Université de Ghardaia, Algerie*, 2016.

[16] Soumia Elyakoute HERMA et Khadija SAIFA. thèses analyse des sentiments cas twitter. *Université de Ghardaia, Algerie*, 2016.

[17] Patrick Fuchs et Pierre Poulain. Cours de python. *Université de Paris, France*, 16 décembre 2020.

[18] Tarhi Fatiha. Application pour dimensionnement d'une installation photololatique pour alimentation du laboratoire de recherche. *Université de Tizi-Ouzou Algerie*, 2011.

[19] Pozzi federico alberto. al challenge of sentiments analysis in social network : An overview. *sentiment analyses in social network. Morgan Kaufman*, 2017. 1-11.

[20] Jean-Michel Ferrard. Une petite référence numpy. 15 octobre 2013.

[21] Vasileios Hatzivassiloglou and Janyce M Wiebe. Effects of adjective orientation and gradability on sentence subjectivity. *In Proceedings of the 18th conference on Computational linguistics-Volume*, 2000.

[22] Yoshua Bengio Ian Goodfellow and Aaron Courville. Deep learning. *http ://www.deeplearningbook.org*, 2016.

[23] Rebecca F Bruce Janyce M Wiebe and Thomas P O'Hara. Development and use of a gold-standard data set for subjectivity classifications. *In Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, pages 246253. Association for Computational Linguistics, year=1999.*

[24] Nitin Jindal and Bing Liu. Identifying comparative sentences in text documents. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.

[25] Eugene Kang. long short term memory(lstm) concept. *UNIVERSITE KASDI MERBAH OUARGLA*, Sep 2-2017.

[26] Ria Kulshrestha. Introduction to word mbedding and word2vec. *Towards Data Science*, 2020-06-01.

[27] NLTK library. Nltk tutorial. *https ://www.nltk.org*, 2020.

[28] Bing Liu. Web data mining : exploring hyperlinks, contents, and usage data. *Springer Science et Business Media*, 2007.

[29] Bing Liu. Identifying comparative sentences in text documents. 2012.

[30] Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 2012.

[31] Henri Michel. de la société jetbrain. *www.jetbrains.com.*, 25/04/2018.

[32] Henri Michel. Google colab le guide ultime. *https ://ledatascientist.com/google-colab-le-guide-ultime/*, 4 Nov. 2019.

[33] mélanie corolleur. analyse des sentiment sur les réseaux sociaux : qu'est-ce que c'est quoi. janvier 11,2016.

[34] Christopher Olah. Understanding lstm networks. August 27, 2015.

[35] Bernhard Scholkopf Olivier Chapelle and Alexander Zien. Semi-supervised learning. *(chapelle, o. et al., eds. ; 2006)[book reviews]. IEEE Transactions on Neural Networks*, 2009.

[36] NASH Ryan. O'SHEA Keiron. An introduction to convolutional neural networks. 2015.

[37] Selva Prabhakaran. Gensim tutorial. *https ://pypi.org/project/gensim*, 2018.

[38] Swapna Somasundaran Jason Kessler Janyce Wiebe Yejin Choi Claire Cardie Ellen Riloff Theresa Wilson, Paul Hoffmann and Siddharth Patwardhan. Opinionfinder : A system for subjectivity analysis. *In Proceedings of hlt/emnlp on interactive demonstrations, pages 3435. Association for Computational Linguistics, year=2005.*

[39] Hans-Dieter Wehle. Machine learning, deep learning, and ai. *What's the July 2017 Conference : Data Scientist Innovation Data*, 2000.

[40] Karlijn Willems. Keras tutorial : Deep learning in python. *https ://www.datacamp.com*, Dec 10 2019.

[41] Daniel Slater Peter Roelants Zocca, Gianmario Spacagna. Python deep learning. 2017.

Le Code Source

1.Declaration les bibliothèques :

Dans cette cellule, toutes les bibliothèques dont nous avons besoin dans le projet sont appelées, y compris les bureaux d'apprentissage en profondeur (tensorflow et Keras et numpay et gensime ...etc.) et les bureaux de traitement du langage naturel (NLTK et String ...etc.), ainsi que Google Drive et graphiques. etc.

```
import pandas as pd
import numpy as np
import gensim
import nltk
from nltk.tokenize import RegexpTokenizer
import nltk.corpus
from nltk.tokenize import PunktSentenceTokenizer
from collections import Counter
from tensorflow.compat.v1 import ConfigProto
from tensorflow.compat.v1 import InteractiveSession
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import  pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding,LSTM,Dense
from tensorflow.keras.callbacks import ModelCheckpoint
from tensorflow.keras.models import load_model
from tensorflow.keras.datasets import imdb
import matplotlib.pyplot as plt
import seaborn as sns
import re
import string
from string import punctuation
from google.colab import drive
drive.mount('/content/gdrive')
```

2.Pour Declaration Google drive:

```
[ ]  from google.colab import drive
     drive.mount('/content/drive')

     Mounted at /content/drive
```

3.Pour utiliser Google Drive :

Rappeler un fichier CSV pré-trié dans Google Drive pour le traitement, l'analyse et la description du contenu du fichier.

Tous les fichiers dont nous avons besoin sont chargés dans Google Drive afin de faciliter le processus de rappel et également de conserver les résultats en cas de changement.

```
x= '/content/drive/MyDrive/Colab Notebooks/train_test1.csv'
data = pd.read_csv(x)
data.describe()
```

	label
count	69999.000000
mean	0.499793
std	0.500004
min	0.000000
25%	0.000000
50%	0.000000
75%	1.000000
max	1.000000

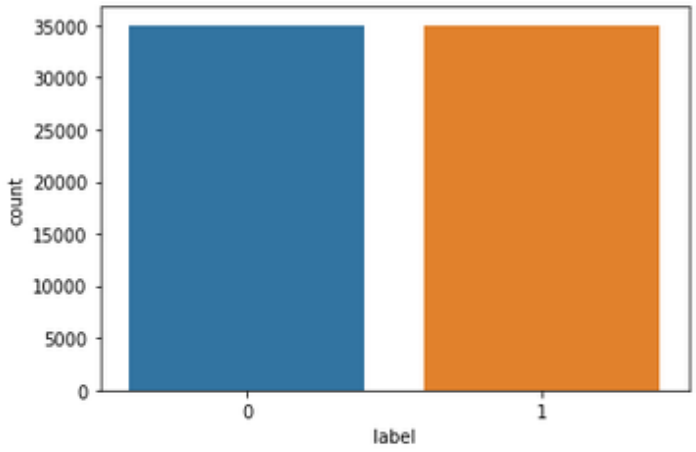
.

4 cellule pour les statistiques :

Dans cette cellule, des statistiques sont affichées pour les textes positifs et les textes négatifs, où le nombre 0 signifie les textes négatifs et le nombre 1 représente les textes positifs.

```
print(data['label'].value_counts())
sns.countplot(x = data['label'])
```

```
0    35014
1    34985
Name: label, dtype: int64
<matplotlib.axes._subplots.AxesSubplot at 0x7f71e2124fd0>
```



5.Nettoyage de texte

Dans cette cellule, le texte est traité à partir de matières qui n'ont aucun effet sur le sens du texte pour l'ordinateur, de virgules de ponctuation, ainsi que de mots vides, ainsi que de la racine et d'autres ajouts.

```
def load_dataset():
  df = pd.read_csv(x)
  x_data = df['text']
  y_data = df['label']

  #Removing notion HTML
  x_data = x_data.replace({'</br> <br>',''},regex=True)
  x_data = x_data.replace({'[A-Za-z]',''},regex=True)

  #Supprimer le mot vide
  x_data = x_data.apply(lambda text:[w for w in text.split() if w not in english_stops])

  #Supprimer la ponctuation
  x_data = x_data.apply(lambda text:[c for c in text if c not in string.punctuation])


  x_data = x_data.apply(lambda text:[c for c in text if c not in string.digits])

  #Convertir le texte en minuscule
  x_data = x_data.apply(lambda text:[w.lower() for w in text])

  return x_data , y_data

def remove_punct(txt):
    txt_nopunct = "".join([c for c in txt if c not in string.punctuation])
    return txt_nopunct

def remove_stop_word(txt):
    txt_stop_word = "".join([c for c in txt if c not in english_stops])
    return txt_stop_word
```

Le résultat est le suivant :

	text	lang	text_clean
	Will Smith Joins Diplo And Nicky Jam For The 2...	en	smith join diplo nicky jam 2018 world cup offi...
	Hugh Grant Marries For The First Time At Age 57	en	hugh grant marries first time age 57
	Jim Carrey Blasts 'Castrato' Adam Schiff And D...	en	jim carrey blast castrato adam schiff democrat...
	Julianna Margulies Uses Donald Trump Poop Bags...	en	julianna margulies us donald trump poop bag pi...
	Morgan Freeman 'Devastated' That Sexual Harass...	en	morgan freeman devastated sexual harassment cl...

6.fractionnement des données :

À ce stade, les données sont divisées en deux sections, et chaque section a un pourcentage spécifique. Nous avons 80% pour la formation et 20% pour les tests.

```
x_train,x_test,y_train,y_test = train_test_split(x_data,
                                                 y_data,
                                                 test_size=0.2 )
print('Data form:')
print(data.shape)
print("Train set:")
print(x_train.shape)
print("Test set:")
print(x_test.shape,)
```

```
Data form:
(69999, 2)
Train set:
(55999,)
Test set:
(14000,)
```

7.Modèle Word2Vec :

Dans cette cellule, le modèle word2vec est créé, puis nous construisons les règles du modèle en insérant les données, puis nous entraînons le modèle sur les données plusieurs fois, et le résultat était le suivant :

```
w2v_model = gensim.models.word2vec.Word2Vec(size=300 ,
                                            window=7 ,
                                            min_count=10 ,
                                            workers=8)
```

```
[ ] w2v_model.build_vocab(x_data)
```

```
[ ] w2v_model.train(x_data,total_examples=len(x_data),epochs=10)
```

```
(81543456, 93055100)
```

Enfin, on teste le formulaire et on teste ma parole et c'était "bad" et "good" et les résultats étaient les suivants

```
w2v_model.wv.most_similar("bad")
```

```
[('bad,', 0.589138388633728),
 ('bad.', 0.5629411935806274),
 ('terrible', 0.5138693451881409),
 ('awful', 0.5104974508285522),
 ('horrible', 0.47498178482055664),
 ('bad!', 0.4596254825592041),
 ('lousy', 0.45230570435523987),
 ('crappy', 0.4398758113384247),
 ('horrible,', 0.43473637104034424),
 ('stupid', 0.42961615324020386)]
```

```
w2v_model.wv.most_similar("good")
```

```
[('decent', 0.6026421785354614),
 ('good,', 0.5978475213050842),
 ('good.', 0.5455121397972107),
 ('fine', 0.4728296399116516),
 ('ok', 0.4497535228729248),
 ('nice', 0.44346365332603455),
 ('excellent', 0.41864457726478577),
 ('good!', 0.4117533564567566),
 ('great,', 0.40884798765182495),
 ('bad.', 0.4059061110019684)]
```

8. Convertisseur de texte :

```
token = Tokenizer(lower=False)
token.fit_on_texts(x_train)
x_train = token.texts_to_sequences(x_train)
x_test  = token.texts_to_sequences(x_test)

max_length = get_max_length()

x_train = pad_sequences(x_train,maxlen=max_length ,padding='post' , truncating='post')
x_test = pad_sequences(x_test,maxlen=max_length, padding='post' , truncating='post')

total_words = len(token.word_index)+1
skipped_words = 0
embedding_dim = 300
embedding_matrix = np.zeros((total_words, embedding_dim))
for word, index in token.word_index.items():
  try:
    embedding_vector = w2v_model[word]
  except:
    skipped_words = skipped_words+1
    pass
  if embedding_vector is not None:
    embedding_matrix[index] = embedding_vector
```

Dans cette cellule, le texte est généralement converti de mots et de phrases en nombres, grâce à l'utilisation de la bibliothèque NLTK (fonction Tokenizer) pour numéroter les mots dans chaque texte et la numérotation de chaque mot dans le texte et dans les données dans leur ensemble, puis nous utilisons la bibliothèque Numpy (fonction pad_sequence) afin de convertir le texte en un tableau numérique afin de faciliter le processus d'apprentissage en profondeur.

Les résultats étaient les suivants :

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:17: D
Embeddings Matrix shape :  (351071, 300)
Encoding Text Train :
 [[   16     3   261 ...     0     0     0]
 [68505     6   570 ...  3704   394   198]
 [  129   674   295 ...     0     0     0]
 ...
 [  420  1965    78 ...     0     0     0]
 [  447 14201 28960 ... 92172  5609  1577]
 [   16   954    40 ...     0     0     0]]

Encoding Text Test : [[   2 1019 1380 ...  932  612    2]
 [2212 3212  595 ...    0    0    0]
 [ 268  197    3 ...    0    0    0]
 ...
 [ 797  304 1612 ...    0    0    0]
 [ 100   70  618 ...    0    0    0]
 [  16   10    3 ... 4234  673   95]]
```

9. Modèle RNN-LSTM:

- Embedding layer : une couche qui convertit notre mot jeton en intégration.

- Dropout Layer : nous avons utilisé une couche d'abandon qui supprime ou ignore les unités (dans les neurones) pendant la phase d'apprentissage de certains ensembles de neurones qui sont choisis au hasard, ceci est fait pour éviter le surapprentissage.

- LSTM Layer:(the long short term memory layer).

- Dense Layer une couche de réseau de neurones profondément connectée avec une fonction d'activation sigmoïde.

```
model = Sequential()
model.add(embedding_layer)
model.add(Dropout(0.5))
model.add(LSTM(100,dropout=0.2,recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))
model.compile( optimizer = "adam" ,loss = "binary_crossentropy",metrics=['accuracy'])
print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 134, 300)          104796300
_________________________________________________________________
dropout (Dropout)            (None, 134, 300)          0
_________________________________________________________________
lstm (LSTM)                  (None, 100)               160400
_________________________________________________________________
dense (Dense)                (None, 1)                 101
=================================================================
Total params: 104,956,801
Trainable params: 160,501
Non-trainable params: 104,796,300
_________________________________________________________________
None
```

L’étape suivante consiste à entraîner le réseau LSTM à travers 10 époques. La figure montre le processus d'entraînement.

```
history =model.fit(x_train,y_train, epochs=10,batch_size=128 , validation_data=(x_test,y_test))

Epoch 1/10
438/438 [==============================] - 419s 957ms/step - loss: 0.2349 - accuracy: 0.9022 - val_loss: 0.2598 - val_accuracy: 0.8962
Epoch 2/10
438/438 [==============================] - 419s 957ms/step - loss: 0.2351 - accuracy: 0.9022 - val_loss: 0.2545 - val_accuracy: 0.8997
Epoch 3/10
438/438 [==============================] - 419s 956ms/step - loss: 0.2339 - accuracy: 0.9018 - val_loss: 0.2494 - val_accuracy: 0.8986
Epoch 4/10
438/438 [==============================] - 420s 959ms/step - loss: 0.2290 - accuracy: 0.9050 - val_loss: 0.2540 - val_accuracy: 0.8980
Epoch 5/10
438/438 [==============================] - 418s 955ms/step - loss: 0.2297 - accuracy: 0.9048 - val_loss: 0.2607 - val_accuracy: 0.8974
Epoch 6/10
438/438 [==============================] - 418s 954ms/step - loss: 0.2310 - accuracy: 0.9035 - val_loss: 0.2575 - val_accuracy: 0.8962
Epoch 7/10
438/438 [==============================] - 418s 954ms/step - loss: 0.2306 - accuracy: 0.9032 - val_loss: 0.2527 - val_accuracy: 0.8979
Epoch 8/10
438/438 [==============================] - 418s 955ms/step - loss: 0.2289 - accuracy: 0.9052 - val_loss: 0.2484 - val_accuracy: 0.8992
Epoch 9/10
438/438 [==============================] - 418s 955ms/step - loss: 0.2286 - accuracy: 0.9065 - val_loss: 0.2581 - val_accuracy: 0.9001
Epoch 10/10
438/438 [==============================] - 419s 956ms/step - loss: 0.2284 - accuracy: 0.9053 - val_loss: 0.2612 - val_accuracy: 0.8976
```

10.Résultats du modèle RNN LSTM

La figure suivante montre la fonction qui permet d'afficher les résultats de l'entraînement en termes de précision et de perte.

```
y_pred = model.predict_classes(x_test,batch_size=128)
score = model.evaluate(x_test,y_test ,batch_size=128)

true = 0
for i,y in enumerate(y_test):
  if y == y_pred[i]:
    true += 1
print('Correct Comment Prediction: {}'.format(true))
print('Wrong Comment Prediciton: {}'.format(len(y_pred) - true))
print("Accuraccy :",score[1])
print("Loss :",score[0])
```

Et à partir de là le résultat est le suivant :

```
  warnings.warn('`model.predict_classes()` is deprecated and '
110/110 [==============================] - 5s 47ms/step - loss: 0.2620 - accuracy: 0.8910
Correct Comment Prediction: 12474
Wrong Comment Prediciton: 1526
Accuraccy : 0.890999972820282
Loss : 0.26204150915145874
```

La cellule suivante contient la fonction qui détermine le pourcentage de texte

```
reuslt = model.predict(token_word)
print(reuslt)

[[0.31168163]]
```

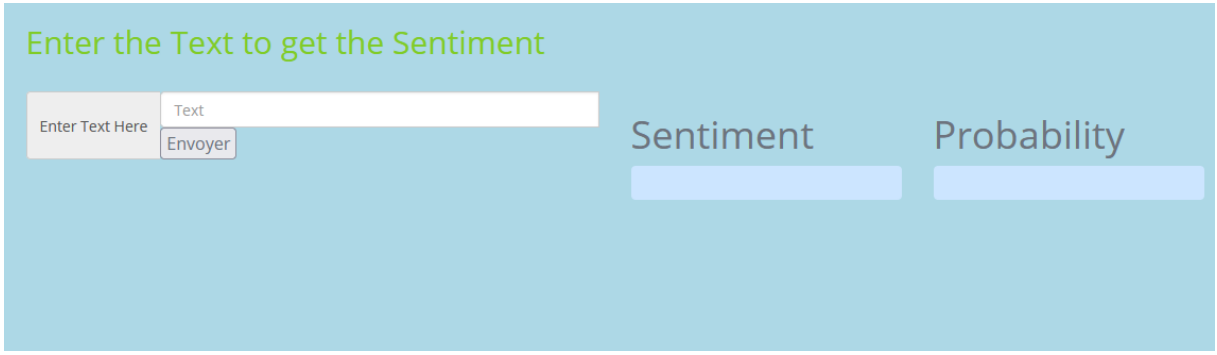
Comme on peut le voir dans la figure précédente que le pourcentage est inférieur à 70 %, cela signifie que le texte est négatif, et si le résultat est supérieur à 70 %, cela signifie que le texte est positif, et la figure suivante montre comment le le processus est effectué.

```
if reuslt >= 0.7:
  print('Postive review')
else:
  print('Negative review')

Negative review
```

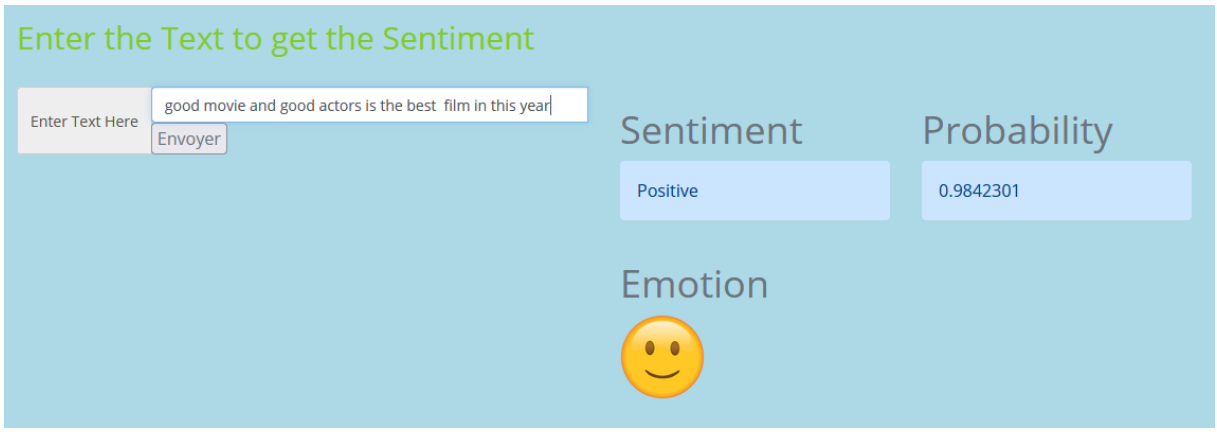
11. Interface d'analyse des sentiments :

Dans cette interface est montré le test de ce modèle LSTM, on a dans cette interface de zones de texte, dans le premier on écrit la phrase à laisser ensuite on click sur le bouton « Envoyer » et le résultat (Positif, Négatif) affiche dans la zone sentiment et probabilité.

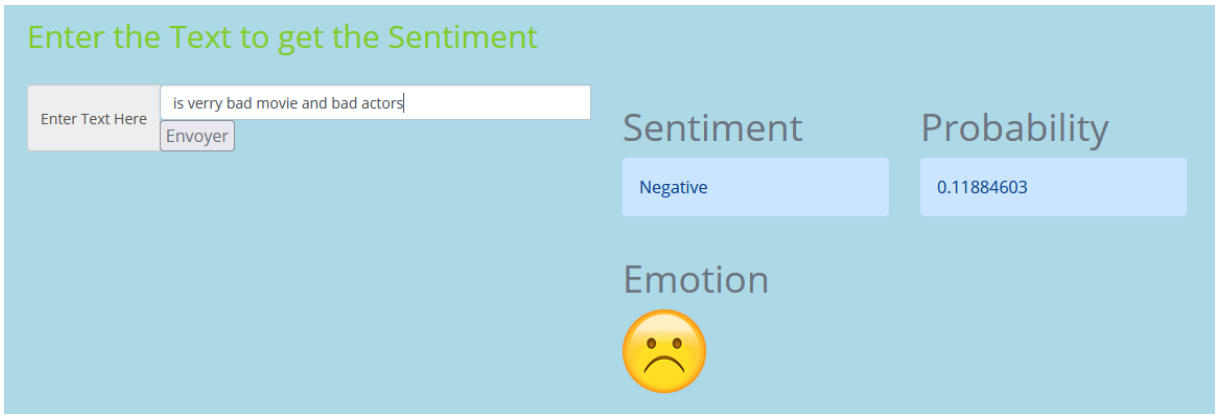


12.Scénario d'utilisation simple :

En tant que simple de notre application, nous choisissons de fournir certains commentaires pour un spécifique, puis nous soumettrons et verrons le résultat renvoyé, dans la figure suivante, nous avons fourni les critiques suivantes "good movie and good actors is the best film in this year" le résultat obtenu « POSITIF » avec un score de 98 %.



D'autre part, dans la figure suivante, nous avons soumis le commentaire suivant "is verry bad movie and bad actors" et avons été correctement classés comme NÉGATIF avec un score de 11 %.


Enter the Text to get the Sentiment
Enter Text Here
is verry bad movie and bad actors
Envoyer
Sentiment
Negative
Probability
0.11884603
Emotion