Remerciements

Au nom de Dieu, le Tout Miséricordieux, et que le salut soit sur le prophète Mohamed, que la bénédiction et la paix soient sur lui.

je remercie Dieu, le tout-puissant, pour son aide et sa protection, et de m'avoir donné le courage d'aller jusqu'au bout de ce travail.

Je tiens à remercier toutes les personnes qui ont contribué au succès de cette période en apprentissage et à l'obtention des connaissances que je possède aujourd'hui. J'adresse mes remerciements à ***Gymglish****, pour m'avoir donnée la possibilité de travailler avec eux. Je tiens à remercier vivement mon maitre d'apprentissage* ***M. Benoit Pernot****et* ***M. Bogdan Sima****, ainsi que toute membre de l'équipe technique. Je souhaite exprimer ma gratitude à tous les membres de l'équipe de Gymglish qui m'ont beaucoup aidé à réaliser mon projet et qui m'ont soutenu jusqu'au bout.*

Je souhaite également exprimer ma gratitude envers tous les professeurs du ***CFA-INSTA*** *Paris, qui m'ont accompagné tout au long de ce parcours magnifique et rempli de découvertes. Leur écoute attentive et leurs conseils précieux m'ont permis d'améliorer mon organisation de travail et de mieux gérer les tâches à accomplir.*

Enfin, je tiens à exprimer tout mon amour et ma reconnaissance à mes chers parents et à mes proches pour leur encouragement, leur patience, leur soutien indéfectible et leur aide tout au long de ces années d'études.

Table des matières

Liste des tableaux

Table des figures

Introduction générale

L'utilisation d'Internet a profondément modifié le monde des affaires dans de nombreux secteurs, et cela inclut également le secteur des agences d'intérim. Ces agences sont confrontées à des défis uniques tels que la gestion d'un grand volume de candidatures, la recherche de profils et la mise en relation efficace des candidats avec les opportunités d'emploi.

Cependant, le processus de mise en correspondance entre les candidats et les missions comporte des défis, notamment en ce qui concerne la recherche de profils et de talents, parce que beaucoup d'approches sont actuellement limitées à la recherche par mot-clé, qui n'est plus efficace lorsque la taille des données devient énorme.

En France, les plateformes de recrutement en ligne sont devenues le principal canal utilisé par de nombreuses entreprises. Bien qu'elles réduisent le temps de recrutement, elles ne garantissent pas la sélection du profil idéal pour un poste spécifique. Par conséquent, de nombreux recruteurs ont manqué l'occasion de recruter des talents et les demandeurs d'emploi ont raté l'occasion d'être recrutés.

C'est dans ce contexte que l'utilisation de l'Intelligence Artificielle (IA) pour le recrutement est devenue courante. Elle offre des solutions efficaces pour gérer les volumes importants de CV et faciliter la mise en relation des candidats avec les offres d'emploi. Cette tendance s'applique également aux agences d'intérim qui doivent faire face à une demande croissante en matière de gestion des candidatures et de mise en relation avec des missions spécifiques.

Dans ce projet, nous proposons une architecture de réseau de neurones artificiels pour la mise en correspondance des profils de candidats (CV) avec les missions (offres d'emploi) dans les agences d'intérim. Ce modèle de réseau de neurones artificiels peut également être utilisé sur des plateformes de recrutement pour aider les recruteurs à trouver les meilleurs candidats pour un emploi donné, tout en aidant les chercheurs d'emploi à trouver des postes correspondant à leur profil tel qu'il est présenté dans leur CV.

Le réseau de neurones artificiels est conçu selon une architecture composée de deux composantes principales. Tout d'abord, un réseau de neurones classificateur est utilisé pour organiser les CV et les offres d'emploi de manière structurée. Ensuite, un autre réseau de neurones est ex-

ploité pour évaluer les similitudes entre les profils des candidats (CV) et les missions disponibles (offres d'emploi). Grâce à la combinaison de ces modèles, il devient possible d'automatiser le processus de mise en correspondance des candidats avec les missions proposées, améliorant ainsi l'efficacité et la précision globale du processus de recrutement.

Ce rapport est organisé comme suit :

Après avoir présenté cette introduction, le premier chapitre aborde l'identité de la société Gymglish, son personnel et t sa localisation en France. De plus, il présente le² service informatique ainsi que les responsabilités quotidiennes liées à mon poste au sein de Gymglish.

e chapitre 2 se concentre sur l'introduction des concepts fondamentaux et de l'état de l'art de l'apprentissage automatique. Les différents types d'apprentissage automatique sont abordés, ainsi que les algorithmes fréquemment utilisés, avec une attention particulière portée aux réseaux neuronaux artificiels. Enfin, une approche spécifique basée sur les réseaux de neurones artificiels est expliquée pour le calcul de similarité.

Au troisième chapitre, nous décrivons notre méthodologie pour apparier les missions et les candidats en utilisant une architecture spécifique de réseau de neurones. Nous fournissons également un aperçu des données utilisées, expliquant la manière dont elles ont été collectées et les différents traitements qui leur ont été appliqués. De plus, nous détaillons les étapes d'entraînement d'un modèle de classification ainsi que d'un modèle de similarité. Les outils employés pour le développement sont également présentés dans ce chapitre.

En conclusion, un résumé de l'ensemble du travail et diverses suggestions pour l'amélioration future de ce projet seront présentés.

Chapitre 1

Présentation de l'entreprise

Identité de l'entreprise :
Dans ce chapitre nous présentons l'entreprise Gymglish. Nous commencerons par aborder l'identité de l'entreprise, ses valeurs et ses objectifs. Ensuite, nous examinerons les effectifs et l'implantation de Gymglish en France, ainsi que le service informatique de l'entreprise. Enfin, nous décrirons les différentes tâches quotidiennes liées à mon rôle au sein de Gymglish. L'objectif de ce chapitre est de fournir une vue d'ensemble de l'entreprise, de son organisation et de son fonctionnement.

1.1 Identité de l'entreprise

La société Gymglish, fondée en 2004 par Antoine Brenner et Benjamin Levy, est une entreprise spécialisée dans l'apprentissage des langues en ligne. Elle propose des solutions innovantes et personnalisées pour aider les apprenants à développer leurs compétences linguistiques. Gymglish est notamment connue pour ses cours d'anglais, de français, d'espagnol, d'italien et d'allemand. L'objectif de Gymglish est de rendre l'apprentissage des langues plus efficace et intéressant pour ses utilisateurs.

FIGURE 1.1 – Gymglish Log.

Leur méthode d'enseignement est basée sur des leçons courtes et personnalisées, qui sont adaptées en fonction des compétences et des besoins individuels de chaque apprenant. Les leçons sont également conçues pour être divertissantes, en utilisant des histoires et des blagues pour rendre le processus d'apprentissage plus agréable.

Voici les différentes informations concernant la société Gymglish :

1. **Fondateurs :** Antoine Brenner et Benjamin Lévy
2. **Nom de l'entreprise :** Gymglish
3. **Date de création de l'entreprise :** 2004
4. **Lieu :** Paris, France
5. **Adresse et siège social :** 65 Rue de Reuilly, 75012 Paris
6. **N° de téléphone :** +33 1 53 33 02 40
7. **Site web :** www.gymglish.com

1.1.1 Produits et Services

Gymglish propose une gamme de produits d'apprentissage des langues en ligne. Ces produits sont basés sur le microlearning et l'adaptive learning. Le microlearning est une méthode qui consiste à proposer des leçons d'une durée maximale de 10 minutes par jour, comprenant des contenus écrits, audio et/ou vidéo. Avec l'adaptive learning, le moteur de Gymglish adapte les leçons en fonction du niveau de l'apprenant, offrant ainsi une expérience d'apprentissage personnalisée.

Les cours offerts par Gymglish, que ce soit dans leur version destinée aux entreprises ou aux universités, aboutissent à l'obtention d'un certificat officiel reconnu par l'État via France Compétences (Gymglish Certificate). A la différence d'examens immédiats comme le TOEIC (Test of English for International Communication) ou le BULATS (Business Language Testing Service), le diplôme Gymglish résulte d'une évaluation continue tout au long de la formation. Celle-ci permet de renseigner l'apprenant, mais également ses professeurs ou managers le cas échéant, sur son niveau, ses statistiques de participation, ses forces, faiblesses, points à réviser, etc.

Voici quelques-uns des principaux cours proposés par Gymglish :

(A) Gymglish :
C'est leur produit phare, un cours d'anglais en ligne qui utilise une histoire courte et des exercices adaptatifs pour aider les apprenants à améliorer leurs compétences en anglais.

(B) Frantastique :
C'est un cours de français en ligne qui utilise une approche similaire à celle de Gymglish, avec une histoire courte et des exercices adaptatifs.

(C) Wunderbla :
Il s'agit d'un cours d'allemand fun, concis et personnalisé adapté aux adultes non débutants.

(D) Rich Morning Show :
Il s'agit d'un cours d'anglais spécialement conçu pour les débutants, qui propose une série de leçons quotidiennes personnalisées avec une vidéo, suivies de questions, de mini-leçons et d'exercices de révision.

(E) Vatefaireconjuguer :
C'est une application de conjugaison de verbes en français, qui fournit des conjugaisons pour des milliers de verbes.

Les formations sont utilisées par plus de 6 millions d'utilisateurs dans le monde, 6 000 entreprises et 500 écoles de langues et universités partenaires.

1.2 Effectifs et Implantation des agences en France

Le siège principale de la société Gymglish est situé à Paris, et elle dispose d'un autre à Bordeaux. Ces locaux sont aménagés en open space. L'équipe de Gymglish est composée d'environ 50 personnes, comprenant des ingénieurs, des commerciaux, des professeurs, des stagiaires ET des alternants. Ces membres proviennent de plus de 20 nationalités différentes et parlent plus de 25 langues. L'équipe est divisée en 4 pôles distincts.

1. Le pôle HR et Finance, sous la direction de M. Antoine Pics, est responsable de la gestion générale de l'entreprise et le recrutement ainsi que des flux financiers.
2. le pôle commercial constitue de deux sous équipe :
   (a) L'equipe BizDev (business development) sous la direction de M. Thoma Pernot et Mme. Anne Ginguay, assume les responsabilités liées aux relations avec les entreprises et universités partenaires, ainsi qu'au support et à la communication avec l'ensemble des clients particuliers de l'entreprise.
   (b) L'équipe Marketing a pour mission de promouvoir les produits Gymglish, de convaincre et de fidéliser les clients. Ses responsabilités comprennent la recherche de marché, l'analyse concurrentielle, la création de campagnes publicitaires, la rédaction de blogs, l'organisation d'ateliers et d'événements, ainsi que les relations publiques. Cette sous-équipe est supervisée par Mme. Lisa Sievers.
3. Le pôle produit, principalement composé d'enseignants de langues telles que l'anglais, le français, l'italien, etc., est responsable de l'édition de contenu textuel, graphique et audio pour les leçons. Ce service est dirigé par M. Adrien Soullier.
4. Le pôle technique est responsable de la création et du développement du produit informatique, notamment du site web et de l'application mobile. Il est dirigé par M. Bogdan Sima. Nous allons fournir plus de détails sur ce pôle dans la section suivante.

Les différentes équipes suivent une approche agile [7], où chaque vendredi, une réunion de présentation est organisée par une équipe. Tous les membres de l'entreprise y assistent, au cours desquels l'équipe explique son travail, son avancement, les prochaines étapes et discute avec le reste de l'équipe des différents projets et enjeux en cours. Cela permet également de répondre aux questions des participants.

1.3 Le Service Informatique

Le service informatique ou l'équipe technique est composé d'un total de 13 personnes. Ils adoptent la méthode Agile [7], où l'équipe est organisée comme suit :

1. Le rôle du Product Owner (PO) est essentiel au sein de l'équipe technique. Le PO travaille en étroite collaboration avec les parties prenantes, y compris les autres équipes, afin de comprendre les exigences et les problèmes rencontrés dans l'application. Il discute des améliorations à apporter et participe à la création de nouveaux produits linguistiques. Le PO a pour responsabilité de définir une vision claire et une stratégie pour atteindre les objectifs, en créant une liste de tâches. Il veille également à ce que les éléments les plus importants soient réalisés en priorité.
2. Les développeurs backend : Le rôle des développeurs Backend est de créer et de gérer la partie serveur de l'application mobile et le site web. Ils sont responsables de la mise en place de la logique métier, du traitement des requêtes et de la gestion des bases de données en utilisant le langages de programation Python, le framework Django et le gestionaire de base de données PostgreSQL. Leurs principales responsabilités comprennent :
   (a) Développement et maintenance du backend : Ils créent les fonctionnalités et la logique métier nécessaires à l'application (le site web et l'application mobile). Ils s'assurent que le backend est efficace, fiable et sécurisé.
   (b) Création d'API : Ils développent des API permettant aux applications mobiles d'interagir de manière sécurisée et structurée avec le backend.
   (c) Gestion des bases de données : Les développeurs Backend conçoivent et gèrent les bases de données, en créant des schémas, en définissant les tables et les relations, et en optimisant les requêtes pour assurer des performances optimales.
   (d) Intégration avec d'autres services et systèmes : Les développeurs Backend intègrent souvent d'autres services et systèmes au backend, tels que des services de paiement, de messagerie ou d'alerte.
   (e) Tests et débogage : Ils effectuent des tests unitaires et fonctionnels pour vérifier le bon fonctionnement du backend et s'assurer qu'il répond aux exigences spécifiées. En cas de bugs ou d'erreurs, ils effectuent des opérations de débogage et apportent les corrections nécessaires.

3. Les développeurs Front-end : Le rôle des développeurs Front-end est de concevoir et gérer la partie client du site web. Ils se focalisent sur l'interface utilisateur, l'expérience utilisateur et l'interaction avec les utilisateurs finaux. En étroite collaboration avec le designer, ils transforment les maquettes et les prototypes en interfaces utilisateur interactives. Les développeurs Front-end utilisent le langage JavaScript en combinaison avec le framework Angular pour réaliser leurs tâches.
4. Les développeurs Mobile : Ces développeurs sont responsables de la conception et de la mise en œuvre des fonctionnalités de l'application sur Android et IOS, en s'assurant qu'elle fonctionne de manière fluide et sans problème sur différents appareils mobiles. Ils utilisent des langages de programmation TypeScript et la framework React Native.
5. Devops : sont rôle est de faciliter le déploiement des fonctionnalités développées par les développeurs. Ils se concentre sur l'automatisation, la collaboration et l'intégration continue pour assurer un déploiement fluide. Le DevOps est responsables de la configuration et de la mise à niveau des outils tels que Jenkins, ainsi que des langages de programmation utilisés.

Remarque 1.3.1 *Chaque semaine, un développeur est désigné pour réaliser une mission de réactivité. Pendant cette mission, le développeur est chargé de résoudre les problèmes techniques rapportés par les clients et les autres équipes de l'entreprise.*

Chaque mercredi, l'équipe technique se réunit pour partager les mises à jour, discuter des difficultés rencontrées, proposer des améliorations et présenter de nouveaux outils. Cette réunion permet également de communiquer sur les nouveaux projets à venir.

1.4 Mes missions quotidiennes

En tant que développeur Python full stack en alternance, j'ai été membre de l'équipe de développement Backend. Mes missions quotidiennes consistaient à développer, améliorer et supprimer des fonctionnalités du site web et de l'API de l'application mobile, ainsi qu'à effectuer des tests et le déploiement des fonctionnalités. Le langage de programmation utilisé est Python avec le framework Django et le langage SQL pour interroger la base de données. GitHub est utilisé comme gestionaire de version pour faciliter la gestion de notre code source, la collaboration entre les développeurs et l'intégration continue.

Pendant mon alternance, j'ai effectué diverses tâches liées au software engineering, tel que le processus de développement d'une fonctionnalité passe généralement par cinques étapes principales : la compréhension des spécifications, la programmation, la programmation de tests unitaires et enfin la soumission de demandes de fusion (pull requests) et le déploiement. La Figure 1.2 illustre quelques exemples de projets que j'ai réalisés.

index	Clé de ticket	Résumé	Type de projet	Rapporteur	État	Temps consacré	Sprint
35	TEK-11585	Automatically send B2B KPI report	software	Bogdan Sima	Résolu	13.0	September 2021
29	TEK-12558	All products activation keys - activationkey,producttypes_json	software	Clémence Larcher	Résolu	17.0	September 2022
31	TEK-12509	The 'hide user info' of user could be editable in the BO	software	Masha timofeeva	Résolu	11.0	September 2022
27	TEK-12600	Changer CTA / reusable block sur Wordpress	software	Masha timofeeva	Résolu	3.0	October 2022
26	TEK-12610	Automatize partner reports (linguaphone, lic)	software	Masha timofeeva	Résolu	8.0	October 2022
25	TEK-12617	idpartner as a possible option in price_grid_condition	software	Bogdan Sima	Résolu	11.0	October 2022
22	TEK-12634	Cancel and activate all licenses available of a bunch	software	Clémence Larcher	Résolu	10.0	October 2022
17	TEK-12728	Mobile API lesson - A/B test SEND button sticky	software	Bogdan Sima	Résolu	10.0	November 2022
28	TEK-12583	Signin/signup instead of Email (ex Address) page in B2C shop	software	Masha timofeeva	In Review	3.0	November 2022
5	TEK-13020	move optins attributes from website to a9engine	software	Ayoub Bouchachia	In Review	5.0	November 2022
24	TEK-12626	Shop confirmation: display tooltip when one off is disabled	software	Aurelien Matouillot	Résolu	5.0	November 2022
21	TEK-12641	Remove all 'draft' content from our twice-weekly xml reports	software	Masha timofeeva	Résolu	14.0	November 2022
18	TEK-12727	Mobile API product we need a new version with shop arguments updated.	software	Bogdan Sima	Résolu	22.0	November 2022
20	TEK-12698	'Interets Pédagogiques' page Native design - backend	software	Bogdan Sima	Résolu	18.0	December 2022
15	TEK-12788	Special bridge tip for Le Point	software	Bogdan Sima	Résolu	11.0	December 2022
13	TEK-12852	Remove Funky Friday from User Space / mobile app	software	Bogdan Sima	Résolu	2.0	February 2023
33	TEK-12025	Clean url with int by using django.url.path <int:xxx> to directly cast it	software	Clémence Larcher	Résolu	3.0	February 2023
14	TEK-12880	End of test: add promo discount and use another email if unsubscribed	software	Masha timofeeva	Résolu	2.0	February 2023
10	TEK-12914	Remove all forum occurence in a9engine	software	Aurelien Matouillot	Résolu	3.0	February 2023
11	TEK-12909	Lesson teaser: inject the translation of the title in the popover	software	Masha timofeeva	Résolu	2.0	February 2023
23	TEK-12759	Emojis for provide selection (WB, lesson, App) - SOCK only first	software	Masha timofeeva	Résolu	6.0	February 2023
12	TEK-12881	Suppression offre 6 mois Mémorable en février	software	Masha timofeeva	Résolu	1.0	February 2023
9	TEK-12950	Storyless Polish: No more dedicated INTRO lesson	software	Masha timofeeva	Résolu	9.0	March 2023
36	TEK-11417	Change in minimum due for call to invoice	software	Bogdan Sima	Résolu	19.0	March 2023
8	TEK-12951	Storyless Polish: Inverser Image et Tuto	software	Masha timofeeva	Résolu	4.0	March 2023

index	Clé de ticket	Résumé	Type de projet	Rapporteur	État	Temps consacré	Sprint	Sprint-date
7	TEK-12981	Prepare CGV + shop/website/comm for auto-renewal	software	Masha timofeeva	En cours	5.0	March 2023	2023-03-01 00:00:00
6	TEK-12988	STORYLESS: Logique TUTO : avoir un 'first time' displayed et un autre par défaut	software	Masha timofeeva	Résolu	12.0	March 2023	2023-03-01 00:00:00
16	TEK-12784	Remove use of external fonts in template emails	software	Aurelien Matouillot	Résolu	6.0	March 2023	2023-03-01 00:00:00
3	TEK-13032	Add attributes to ProductypeSettings for PRODUCT_CATEGORY_LIGHT	software	Ayoub Bouchachia	Résolu	6.0	March 2023	2023-03-01 00:00:00
2	TEK-13040	remove show_provides_pro from producttype_settings	software	Ayoub Bouchachia	Résolu	10.0	March 2023	2023-03-01 00:00:00
19	TEK-12715	Remove set_abtesting_facebookads_discount	software	Bogdan Sima	Résolu	1.0	March 2023	2023-03-01 00:00:00
32	TEK-12433	Sync vitc verbs (js files) in the mobile apps	software	Bogdan Sima	Résolu	20.0	March 2023	2023-03-01 00:00:00
37	TEK-9655	Email footer : no more www.gymglish.com www.frantastique.com etc.	software	Leo Tingvall	Résolu	1.0	April 2023	2023-04-01 00:00:00
1	TEK-13088	Migrate optparse to argparse	software	Clémence Larcher	Résolu	9.0	April 2023	2023-04-01 00:00:00
4	TEK-13029	Change the name of the variable can_see_list_of_webcontacts to hide_list_of_webcontacts	software	Ayoub Bouchachia	Résolu	2.0	April 2023	2023-04-01 00:00:00
0	TEK-13182	BBSOCK FR : support adding authors	software	Masha timofeeva	In Review	5.0	June 2023	2023-06-01 00:00:00
30	TEK-12535	Remove get_image_mobile_details_deprecated	software	Bogdan Sima	Résolu	1.0	NaN	NaT

FIGURE 1.2 – Des exemples de projets que j'ai réalisés

1. La compréhension des spécifications : Pour la compréhension des spécifications, lorsque je me vois attribuer une tâche par un développeur ou le product owner (PO) via Jira [1], ma première étape consiste à lire attentivement les spécifications afin de comprendre l'objectif. Si je rencontre des difficultés, j'organise une réunion avec le rédacteur des spécifications pour obtenir des explications supplémentaires et clarifier la tâche à effectuer.
2. La programmation : Dans la phase de programmation, mon objectif est de développer des fonctions, des scripts ou des endpoints en fonction des besoins du projet. Je peux également supprimer des fonctionnalités ou effectuer des tâches de nettoyage (cleanup) si nécessaire. Pour cela, j'utilise le framework Django et le langage de programmation Python. De plus, je peux effectuer des modifications dans la base de données PostgreSQL lorsque cela est requis.
3. la programmation de tests unitaires : Après la phase de développement, j'écris des tests unitaires pour garantir le bon fonctionnement de la fonctionnalité ajoutée. À chaque mise à jour du code source, ces tests unitaires s'exécutent pour vérifier si la fonctionnalité a été cassées ou non. Si lors de la phase de programmation, j'ai supprimé une fonctionnalité, il est essentiel de supprimer les tests unitaires correspondants à cette fonctionnalité à cette étap. Les tests unitaires sont essentiels pour assurer la qualité et la stabilité du code.
4. La soumission de demandes de fusion (pull requests) : Après avoir confirmé que tous les tests unitaires ont été exécutés avec succès, je procède à la soumission d'une demande de fusion (pull request) afin d'intégrer la fonctionnalité développée dans le code déployé en production. Cette demande de fusion est ensuite examinée attentivement par un autre développeur, qui se charge d'évaluer la qualité, la cohérence et la conformité du code aux normes établies. Cette revue permet également de repérer d'éventuelles erreurs, de proposer des améliorations ou des corrections, et de vérifier que la fonctionnalité répond aux exigences spécifiées. Il s'agit d'une étape cruciale visant à garantir la qualité du code et à encourager la collaboration au sein de l'équipe de développement.
5. Le déploiement : Une fois que la demande de fusion a été validée, mon code est fusionné dans la branche principale (master) et je procède au déploiement en utilisant un script spécifique exécuté à partir d'un logiciel technique dédié. Après le déploiement, je veille à relancer les démons (crons) pour garantir le bon fonctionnement de la fonctionnalité. À ce stade, la fonctionnalité est désormais accessible aux clients.

Chaque lundi, j'ai également une mission d'une heure appelée "Sentry party". Durant cette mission, je travaille en collaboration avec deux autres développeurs pour analyser les erreurs complexes signalées en production. Nous cherchons à comprendre les causes de ces erreurs et proposons des solutions pour les résoudre.

1. Jira est un système de suivi de bugs, de gestion des incidents et de gestion de projets développé par Atlassian et publié pour la première fois en 2002.

1.5 Conclusion

Dans ce chapitre, nous avons présenté l'entreprise Gymglish, détaillant son identité et ses produits. Nous avons également discuté de la taille de l'équipe et de la répartition de ses agences en France. Par la suite, nous avons expliqué le fonctionnement de l'équipe technique ainsi que les différentes sous-équipes. Enfin, j'ai décrit les tâches quotidiennes que j'effectuais au sein de l'entreprise.

Dans les deux chapitres suivants, nous examinerons le processus et la technique utilisés pour réaliser le projet.

Chapitre 2

Apprentissage automatique pour l'authentification des réseaux UAVs

2.1 Introduction

Ces dernières années, les technologies de l'information ont occupé une place importante dans notre monde, les processus et les traitements qui doivent être gérés par les ordinateurs aujourd'hui sont devenus plus complexes, de sorte qu'ils ne peuvent pas être résolus par les techniques de programmation traditionnelles. Pour cela, des recherches en apprentissage automatique ont été proposées afin de trouver des solutions optimales et fiables pour les problèmes complexes, y compris la recherche d'information, le traitement du langage, la reconnaissance vocale, la reconnaissance faciale, etc.

Dans ce chapitre, nous présentons les définitions de base et quelques types d'apprentissage automatique. Ensuite, nous montrons les différents types de problèmes abordés par l'apprentissage automatique et certains algorithmes utilisés pour résoudre ces problèmes, ainsi nous détaillons les algorithmes de réseau de neurones. Nous expliquons en fine de ce chapitre une méthode de calcule de similarété basé sur les réseau de neurones artificil.

Remarque 2.1.1 *Ce chapitre est conçu pour faciliter la compréhension de notre travail, en fournissant une vue d'ensemble de la technique de réseau de neurones artificiels qui sera ensuite mise en œuvre dans le chapitre suivant.*

2.2 Définitions

Cette section introduit les définitions de base sur l'apprentissage automatique.

2.2.1 Apprentissage automatique

L'apprentissage automatique fait référence au développement, l'analyse et l'implémentation de méthodes qui permettent aux machines d'apprendre à partir des données (Dataset) plutôt que par programmation explicite [8]. L'objectif principal du processus d'apprentissage est de permettre aux ordinateurs d'apprendre automatiquement sans intervention humaine et d'ajuster les actions en conséquence. [9].

2.2.2 Dataset

Dataset est mots anglais qui désigne un ensemble de données relative à un ou plusieurs domaines bien spécifiques, regroupés ensemble pour permettre l'étude et la description des différents phénomènes tels que du texte linguistique, d'image, des données discrètes, etc. Le dataset peuvent être pré-traités ou non selon le besoin.

2.2.3 Modèle d'apprentissage

Un modèle d'apprentissage automatique est le résultat généré après l'entraînement de l'algorithme d'apprentissage automatique par un dataset. Lorsque le modèle reçoit les données d'entrée, il effectue une prédiction déterminée par les données qui ont servi pour le former.

2.2.4 Évaluation du modèle d'apprentissage

La dernière étape de conception de modèle d'apprentissage est l'évaluation de performances. Cette étape est généralement effectuée de manière expérimentale, sa difficulté réside dans le choix d'une mesure d'evaluation. De nombreuses mesures ont été utilisés pour les différents types de problèmes, comme la précision (voire l'equation (2.1)), le rappel et F-measure [10].

$$Prcision = \frac{Nombre\ de\ prdictions\ correctes}{Nombre\ total\ de\ prdictions} \tag{2.1}$$

2.3 Problèmes traités par l'apprentissage automatique

L'apprentissage automatique traite trois types de problèmes, les problèmes de régression, les problèmes de classification multiple et binaire.

2.3.1 Problème de régression

La régression est l'un des problèmes résolus par les algorithmes d'apprentissage automatique où les informations de sortie (les étiquettes) de ce problème sont des valeurs contenues $y \in R$. Par exemple, le problème de proposition de prix d'une maison en fonction de ses caractéristiques (nombre de chambres, l'adresse, etc).

2.3.2 Problème de multi-classification

Dans ce problème, l'espace des étiquettes est discret et fini, autrement dit $Y = 1, 2, ..., C$ où Y est l'ensemble des étiquettes et C est le nombre de classes possibles. Un exemple de ce problème est l'identification de la langue d'un texte.

2.3.3 Problème de classification binaire

Le problème de classification binaire est un problème d'apprentissage dans lequel l'espace des étiquettes est binaire $Y = 0, 1$. Un exemple de ce problème est de vérifier si un e-mail est un spam ou non.

2.4 Types d'apprentissage automatique

Les algorithmes d'apprentissage automatique peuvent se catégoriser selon le type d'apprentissage qu'ils emploient. Il existe plusieurs types d'apprentissage, tels que l'apprentissage supervisé, non supervisé et par renforcement.

2.4.1 Apprentissage supervisé

L'apprentissage supervisé exploite un dataset étiquetées, dans le but de prévoir les valeurs d'étiquettes sur des données supplémentaires non étiquetées. Cette technique est basée sur un algorithme qui peut apprendre en comparant sa sortie réelle avec les sorties apprises pour trouver des erreurs et modifier le modèle en conséquence. [11].

2.4.2 Apprentissage non-supervisé

Dans l'apprentissage non supervisé, les données utilisées pour entraîner le modèle ne sont ni classées ni étiquetées, de sorte que l'algorithme d'apprentissage trouve tout seul des points communs parmi ses données d'entrées [11]. L'apprentissage non supervisé réduit le problème du manque de données étiquetées.

2.4.3 Apprentissage par renforcement

L'apprentissage par renforcement (RL) est basé essentiellement sur l'interaction d'un agent avec un environnement. Chaque interaction de l'agent dans l'environnement est soit récompensée, soit pénalisée. Ce mécanisme récompense/pénalité permet à l'agent de modifier son propre comportement et son interaction avec l'environnement afin d'avoir un maximum de gain.

2.5 Algorithmes d'apprentissage automatique

Plusieurs algorithmes ont été proposés pour faire apprendre à une machine par elle-même, de sorte que chaque algorithme puisse être appliqué dans un type de problème d'apprentissage automatique (supervisé, non supervisé, renforcement). Ci-dessous, nous présentons quelques algorithmes d'apprentissage automatique qui sont largement utilisés.

2.5.1 Algorithme des k plus proches voisins (K-NN)

k plus proches voisins KNN (k-Nearest-Neighbors) [12] est l'un des algorithmes non paramétriques le plus simple à implémenter pour faire apprendre un problème d'apprentissage supervisé. L'objectif de l'algorithme KNN est de prédire les étiquettes des nouveaux points de données en induisant les points de données d'entraînement, la prédiction se fait en calculant la similarité entre le point à classifier et les points existants dans l'ensemble d'entraînement, il existe plusieurs mesures de similarité telles que la distance euclidienne et la similarité cosinus. Le point est affecté à la classe la plus présente dans les k plus proches voisins. Le nombre k doit être un nombre impair (Figure 2.1).

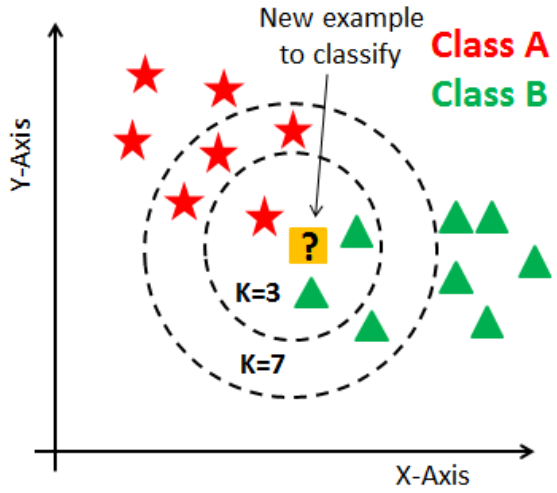

FIGURE 2.1 – L'image montre la classification d'une nouvelle instance pour $k = 3$ et $K = 7$, pour $k = 3$ la nouvelle instance est catégoriser par la classe B et pour $k = 7$ il est catégoriser par la classe A [1]

2.5.2 Machine à vecteurs de support SVM

La méthode machine à vecteurs de support SVM (support vector machine) [13] appartient à la classe des algorithmes d'apprentissage automatique supervisé. Le SVM cherche la surface de décision qui sépare le mieux les données dans un espace à n dimensions en maximisent la distance entre les points des différents classes, ce séparateur appelé hyperplan. Les éléments les

plus proches de la surface de décision sont appelés le vecteur de support, ils sont utilisés pour la détermination d'hyperplan (Figure 2.2).

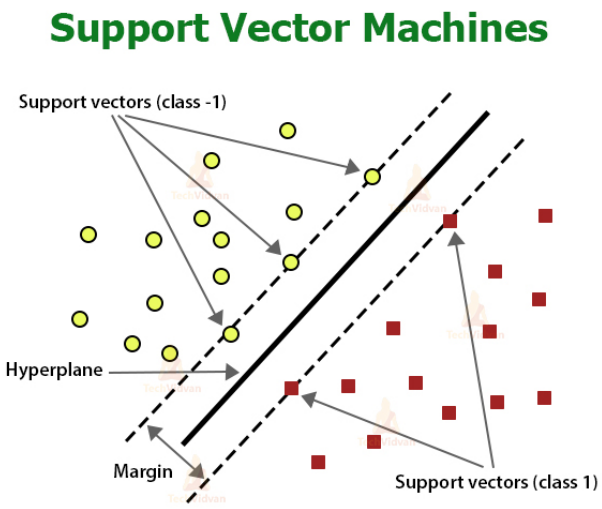

FIGURE 2.2 – La séparation de l'hyperplan par les SVM [2]

2.5.3 Régression linière et régression logistique

La régression linéaire est un algorithme d'apprentissage automatique basé sur des concepts issus des statistiques. La régression linéaire traite des problèmes de régression et de classification, où l'algorithme utilise des données d'apprentissage étiqueté pour trouver une ligne droite qui sépare mieux les données (Figure 2.3).

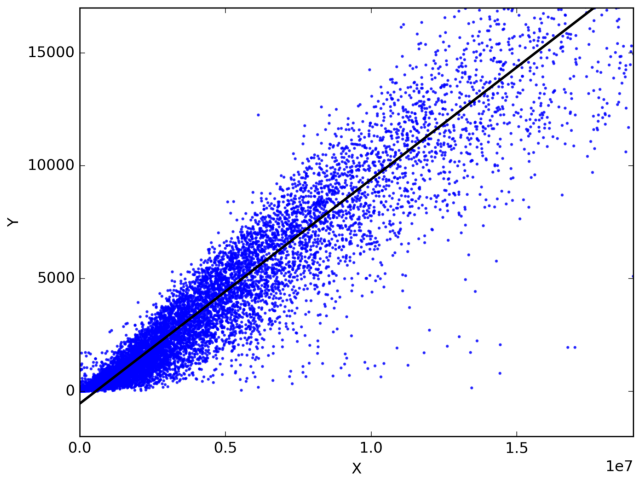

FIGURE 2.3 – La séparation de données par la régression linière [3]

La régression logistique est un algorithme appliqué pour les problèmes de classification bi-

naire et la régression. La régression logistique est un cas particulier de l'algorithme de régression linéaire, où pendant l'entraînement, il essaie de créer un séparateur non linéaire afin de séparer les données en deux classes (Figure 2.4).

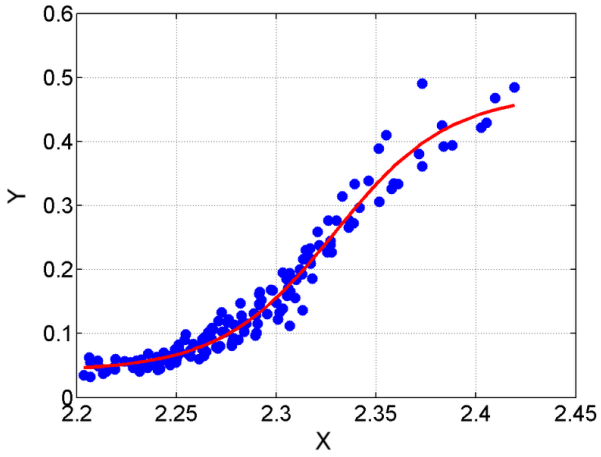

FIGURE 2.4 – La séparation non liniare de données par la régression logistique [4]

2.5.4 K-means

C'est un des algorithme d'apprentissage non supervisé de clustering (regroupement) les plus répandus. Il permet d'analyser un dataset caractérisées par un ensemble de descripteurs, afin de regrouper les données 'similaires' en clusters (groupes). k-means prend en entrée une valeur k qui représente le nombre de centroides [1] puis il affecte chaque point de données au cluster le plus proche. Au début les centroides sont choisi aléatoirement en suite l'algorithme essaye d'optimiser les positions des centroides, si les positions des centroides ne changent pas, l'algorithme se termine. La figure 2.5 montre la création de trois cluster.

1. Le centre du cluster

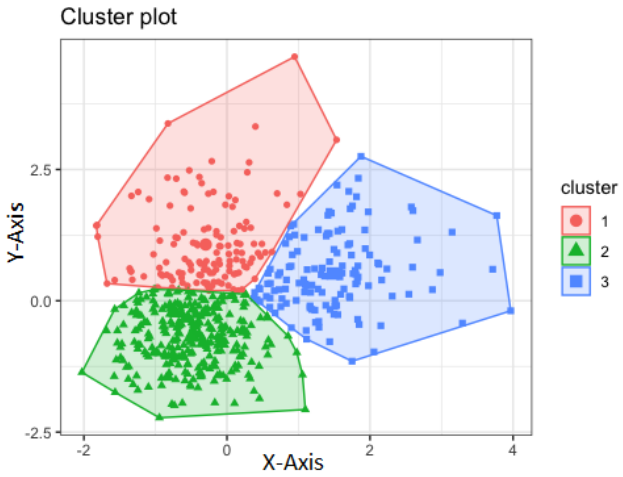

FIGURE 2.5 – La création de trois cluster par l'algorithme de k-means [5]

2.6 Les réseaux de neurones artificiels

Les réseaux de neurones artificiels ANN (artificial neural networks) [14], sont des techniques d'apprentissage automatique basés sur les notions mathématiques dont leur structure s'inspire de celle du système nerveux. Ces réseaux permettent à une machine d'apprendre à effectuer des tâches complexes en analysant des exemples étiquetés. Le réseau se compose d'un ensemble de neurones artificiels appelés perceptrons interconnectés traitant des sous-taches du problème principal.

2.6.1 Définitions de base

1. **Perceptron :** Le perceptron est la plus petite unité de construction d'un réseau de neurones, cette unité est composée de quatre parties, la couche d'entrée (input), des poids et une constante appelée biais, une pré-activation qu'est une somme de produit (la somme des entrées x^i multipliées par leurs poids w^i) et une fonction d'activation (Figure 2.6).

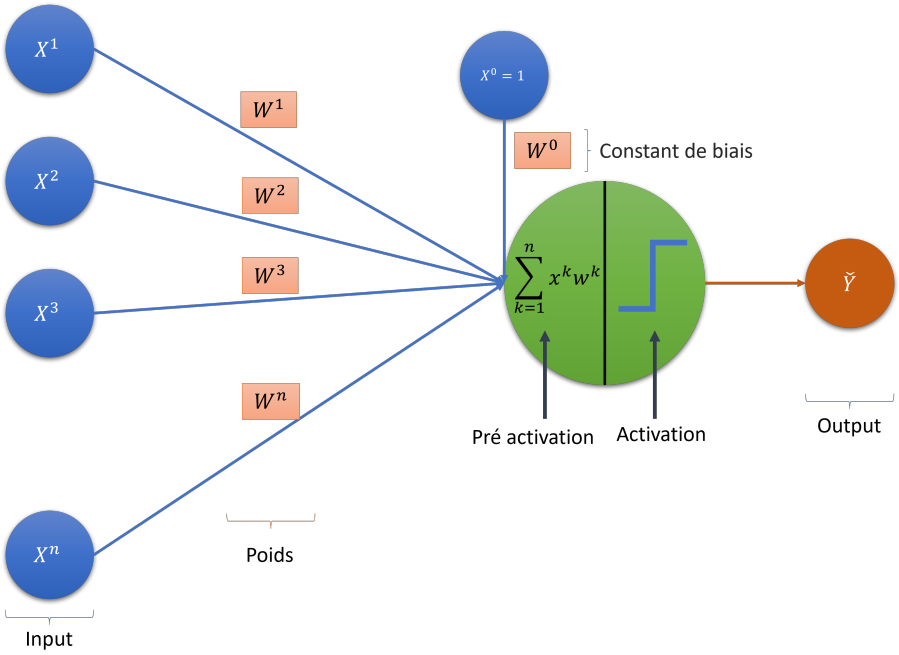

FIGURE 2.6 – L'architecture détailler d'un perceptron

Au cours de l'apprentissage du perceptron, l'algorithme d'apprentissage (ou algorithme de rétro-propagation) tente d'ajuster les poids w^i afin de pouvoir produire des sorties correctes pour des entrées qui n'ont pas été observées auparavant.

2. **Fonction d'activation :** la fonction d'activation (ou fonction de transfert) est une fonction mathématique dont le rôle est de rendre les perceptrons capables de traiter des problèmes non linéaires, c-à-d. qu'ils peuvent fonctionner dans des situations où les données ne sont pas séparables avec une ligne droite (voire les figures 2.7 et 2.8). Une autre utilisation des fonctions d'activation est de transformer la sortie de la pré-activation entre des valeurs requises telles que $[0, 1]$ ou $[-1, 1]$.

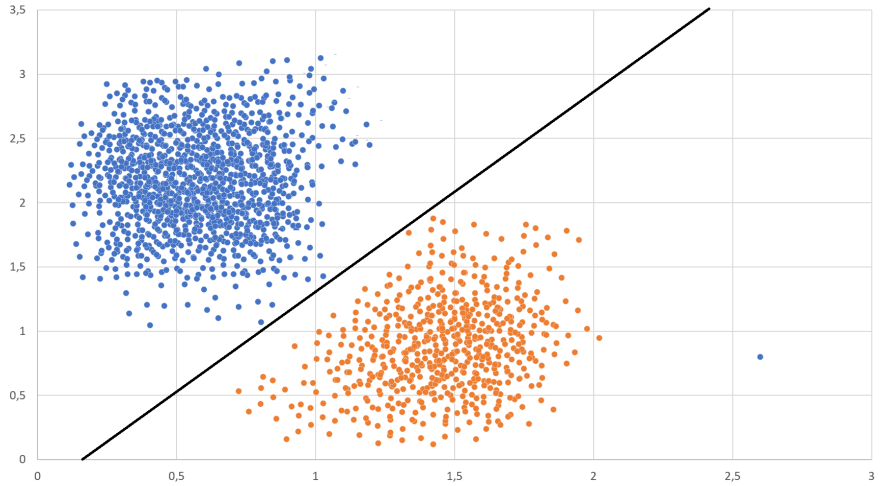

FIGURE 2.7 – exemple d'un problème linaire, c-à-d. qu'il existe une ligne droite qui sépare les deux distributions de données. Dans ce problème, nous pouvons trouver une ligne droite qui sépare les deux distributions sans l'utilisation d'une fonction d'activation non linaire

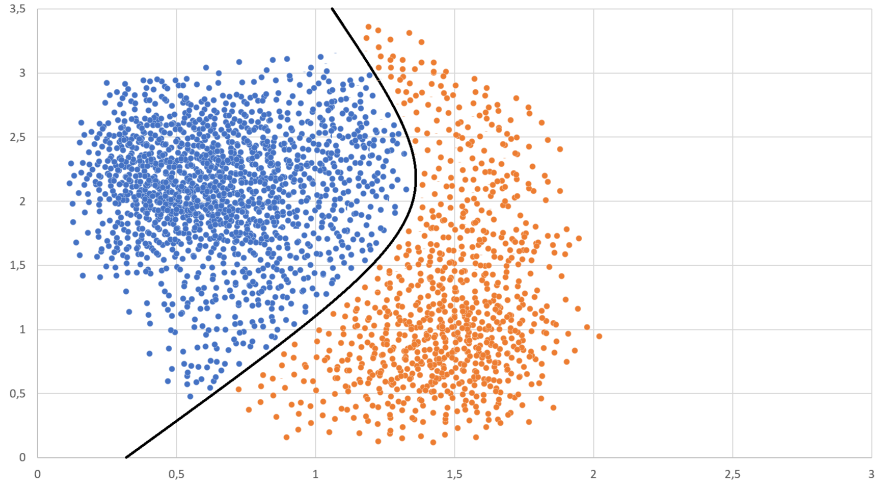

FIGURE 2.8 – exemple d'un problème non linaire, c-à-d. que les données ne sont pas séparables par une ligne droite. Dans ce problème, sans l'utilisation d'une fonction d'activation non linaire nous ne pouvons pas trouver une séparation non linaire

Plusieurs fonctions d'activation ont été proposées, telles que la tangente hyperbolique (tanh), Logistique (sigmoid), Unité de rectification linéaire (ReLU) et la fonction identité (linière). La figure 2.9 présente ces fonctions.

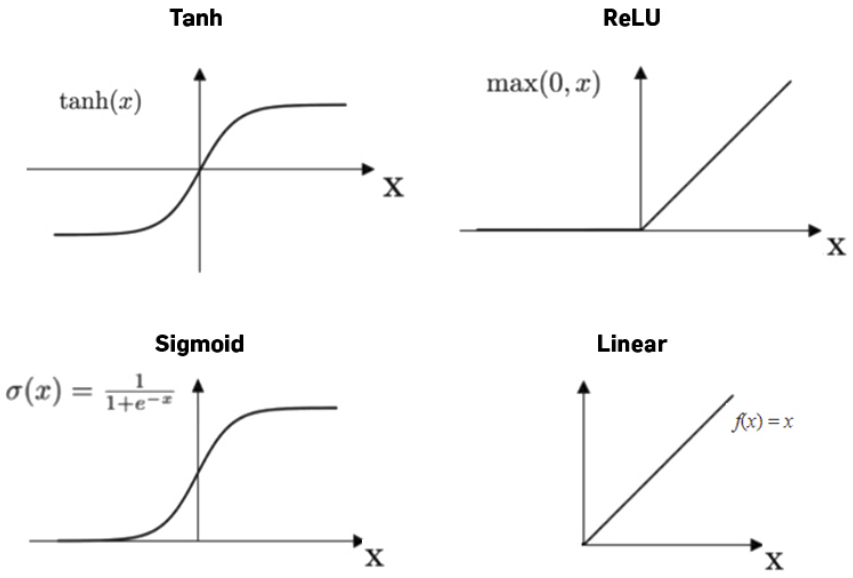

FIGURE 2.9 – Exemple des fonction d'activation [6]

3. **Fonction de perte :** la fonction de perte est l'un des composants des réseaux de neurones. La perte n'est rien d'autre qu'une erreur de prédiction où la méthode pour la calculer s'appelle la fonction de perte ou fonction d'erreur. La fonction de perte est utilisée par l'algorithme d'apprentissage (de rétro-propagation) pour l'ajustement des poids et par conséquent cette fonction sera minimisée pendant l'entrainement.

   Il existe plusieurs fonctions de perte, telles que la fonction d'erreur quadratique moyenne (MSE pour Mean Squared Error) et l'entropie croisée binaire (BCE pour Binary Crossentropy).

   La perte MSE, donnée par la formule (2.2), est utilisée pour des problèmes de régression, cette perte est calculée en prenant la moyenne des différences au carré entre les valeurs cibles et prévues.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.2}$$

   — n est le nombre des instances en entrée.

   — y_i est la sortie désire (prévues) pour la $i$ème instance.

   — $\hat{y}_i$ est la sortie prédit par le réseau de neurone pour la $i$ème instance.

   La perte BCE, donnée par la formule (2.3) est utilisée pour les tâches de classification binaire, si nous utilisons la perte BCE, nous n'avons besoin que d'un nœud de sortie pour classer les données en deux classes. La valeur de sortie doit passer par la fonction d'activation sigmoïde et la plage de sortie doit être entre $[0 - 1]$.

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}[y_i * log(\hat{y}_i) + (1 - y_i) * log(1 - \hat{y}_i)] \tag{2.3}$$

— n est le nombre des instances en entrée.

— y_i est la sortie désirer pour la i ème instance, $y \in 0, 1$.

— $\hat{y}_i$ est une probabilité produite par le réseau de neurones pour la i ème instance, $\hat{y} \in [0, 1]$

Le réseau de neurones essaye de minimiser la fonction de perte pendant l'entraînement, pour la perte BCE par exemple si $y_i = 1$ l'algorithme maximise $log(\hat{y}_i)$ (minimise $-log(\hat{y}_i)$) dans ce cas $\hat{y}_i$ sera maximisé à 1 sinon si $y_i = 0$ le réseau de neurones maximiser $log(1-\hat{y}_i)$ (minimise $-log(1 - \hat{y}_i)$) et par conséquence $\hat{y}_i$ sera minimisé à 0.

4. **Algorithme de rétro-propagation :**

l'algorithme de rétro-propagation (backpropagation) est un algorithme d'optimisation permettant d'ajuster les paramètres (les poids) d'un réseau de neurones dans le but de réduire le taux d'erreur lors de la prédiction d'une sortie. La rétro-propagation calcule le gradient de l'erreur pour chaque neurone (perceptron) où le gradient est une dérivée directionnelle. Par la suite, l'algorithme propage ce gradient de neurones du sortie vers le neurone d'entrée. Cette technique permet de corriger les erreurs selon l'importance des poids (paramètres) qui ont participé à la réalisation de ces erreurs. Autrement dit, les poids qui contribuent à engendrer une erreur importante se verront modifiés de manière plus significative que les poids qui ont engendré une erreur faible.

Il existe plusieurs algorithmes d'optimisation basés sur la rétro-propagation de gradient, tels que la descente du gradient, la descente du gradient stochastique SDG, la racine carrée de l'erreur quadratique moyenne RMSprop.

Pour comprendre comment fonctionne la rétro-propagation, nous expliquons le processus d'optimisation (d'ajustement) par rétro-propagation avec le plus simple algorithme qu'est la descente du gradient.

Soit un réseau de neurones qui se compose de deux neurones mis en séquence (voir la figure 2.10), avec une couche d'entrée de dimension égale à 2 $\{x_1, x_2\}$ et un ensemble de paramètres (poids) $\{w_1^1, w_2^1, b^1\}$ pour les premiers neurones et $\{w_1^2, b^2\}$ pour le deuxième neurone. Nous exploitons une fonction d'activation a pour les deux neurones et nous choisissons une fonction de perte L.

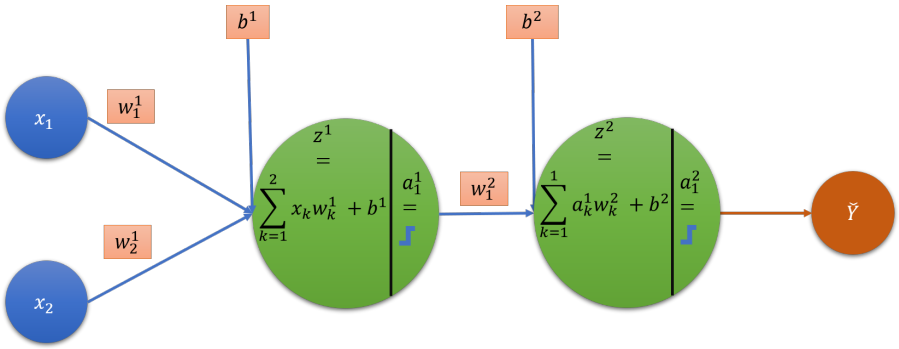

FIGURE 2.10 – Réseau de neurones avec deux perceptrons mis en séquence

La descente de gradient tente d'ajuster les poids jusqu'à que l'erreur soit minimale. La première étape est l'ajustement des poids de neurone de sortie (le deuxième neurone). L'algorithme calcule la dériver directionnel de la fonction de perte L par a port au poids w_1^1 cette dérivée est appelée gradient ∇w_1^1 et faite le même travail avec le poids b^1 pour calculer le gradient ∇b^1. Après avoir calculé les gradients, l'algorithme mit à jour les poids w_1^1 et b^1 il soustraire les gradients calculés multiplié par un taux d'apprentissage γ.

$$\begin{cases} \nabla w_1^2 = \dfrac{\partial L}{\partial w_1^2} \\ \nabla b^2 = \dfrac{\partial L}{\partial b^2} \end{cases} \tag{2.4}$$

$$\begin{cases} w_1^2 = w_1^2 - \gamma * \nabla w_1^2 \\ b^2 = b^2 - \gamma * \nabla b^2 \end{cases} \tag{2.5}$$

Pour généraliser le processus, si nous avons n poids $W = \{w_1, w_2, w_3, ..., w_n\}$, l'algorithme calcule le gradient ∇w_i pour chaque poids w_i et ensuite applique la mis à jour des poids w_i. ∇W est un vecteur de gradient où $\nabla W = \{\nabla w_1, \nabla w_2, \nabla w_3, ..., \nabla w_n\}$.

$$\begin{cases} \nabla W = \dfrac{\partial L}{\partial W} \\ \nabla b = \dfrac{\partial L}{\partial b} \end{cases} \tag{2.6}$$

La mise à jour de vecteur W se fait par la soustraction de vecteur ∇W multiplier par un scalaire représente le taux d'apprentissage γ.

$$\begin{cases} W = W - \gamma * \nabla W \\ b = b - \gamma * \nabla b \end{cases} \tag{2.7}$$

La perte est calculée en fonction de l'activation de neurone du sortie a et la valeur ciblée y, l'activation a est calculée en fonction du résultat de la pré-activation z où $z = W * a^{i-1} + b$, a^{i-1} et l'activation du neurone qui précède le $i$ème neurone. Donc nous n'avons pas le

vecteur de poids W pour calculer directement la dériver $\frac{\partial L}{\partial W}$ et nous n'avons pas b aussi pour calculer la dériver $\frac{\partial L}{\partial b}$, pour cela, il faut calculer le chinage de dériver suivant.

$$\begin{cases} \nabla W = \frac{\partial L}{\partial W} = \frac{\partial L}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial W} \\ \nabla b = \frac{\partial L}{\partial b} = \frac{\partial L}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial b} \end{cases} \tag{2.8}$$

Pour notre exemple, si nous prenons L comme entropie croisée binaire BCE et l'activation a comme sigmoid σ.

$$\begin{cases} \nabla w_1^2 = \frac{\partial BCE}{\partial w_1^2} = \frac{\partial BCE}{\partial \sigma^2} \frac{\partial \sigma^2}{\partial z^2} \frac{\partial z^2}{\partial w_1^2} \\ \nabla b^2 = \frac{\partial BCE}{\partial b^2} = \frac{\partial BCE}{\partial \sigma^2} \frac{\partial \sigma}{\partial z^2} \frac{\partial z^2}{\partial b^2} \end{cases} \tag{2.9}$$

Nous calculons les différents dérivées partial (2.10) et nous remplaçons les différents résultats dans l'équation $\frac{\partial L}{\partial w_1^2}$ et $\frac{\partial L}{\partial b^2}$ (2.11).

$$\begin{cases} \frac{\partial BCE}{\partial \sigma^2} = \frac{-y}{\sigma^2} + \frac{1-y}{1-\sigma^2} \\ \frac{\partial \sigma}{\partial z} = \sigma^2(1-\sigma^2) \\ \frac{\partial z}{\partial w_1^2} = \sigma^1 \\ \frac{\partial z}{\partial b^2} = 1 \end{cases} \tag{2.10}$$

$$\begin{cases} \nabla w_1^2 = (\sigma^2 - y)\sigma^1 \\ \nabla b^2 = (\sigma^2 - y) \end{cases} \tag{2.11}$$

Après l'ajustement, des poids de neurones de sortie ont été accomplis, il faut ajuster les poids des premiers neurones. Dans cette étape l'algorithme contenu à propager le gradient vers l'arrière. Elle calcule le gradient ∇w_1^1, ∇w_2^1 et ∇b^1 pour w_1^1, w_2^1 et b^1 respectivement.

$$\begin{cases} \nabla w_1^1 = \frac{\partial L}{\partial w_1^1} \\ \nabla w_2^1 = \frac{\partial L}{\partial w_2^1} \\ \nabla b^1 = \frac{\partial L}{\partial b^1} \end{cases} \tag{2.12}$$

$$\begin{cases} w_1^1 = w_1^1 - \gamma * \nabla w_1^1 \\ w_2^1 = w_2^1 - \gamma * \nabla w_2^1 \\ b^1 = b^1 - \gamma * \nabla b^1 \end{cases} \tag{2.13}$$

Pour calculer les différents gradients, il faut calculer le chinage de dériver suivant pour les trois paramétrés.

$$\begin{cases} \nabla w_1^1 = \dfrac{\partial BCE}{\partial w_1^1} = \dfrac{\partial BCE}{\partial \sigma^2}\ \dfrac{\partial \sigma^2}{\partial z^2}\ \dfrac{\partial z^2}{\partial \sigma^1}\ \dfrac{\partial \sigma^1}{\partial z^1}\ \dfrac{\partial z^1}{\partial w_1^1} \\ \nabla w_2^1 = \dfrac{\partial BCE}{\partial w_2^1} = \dfrac{\partial BCE}{\partial \sigma^2}\ \dfrac{\partial \sigma^2}{\partial z^2}\ \dfrac{\partial z^2}{\partial \sigma^1}\ \dfrac{\partial \sigma^1}{\partial z^1}\ \dfrac{\partial z^1}{\partial w_2^1} \\ \nabla b^1 = \dfrac{\partial BCE}{\partial b^1} = \dfrac{\partial BCE}{\partial \sigma^2}\ \dfrac{\partial \sigma^2}{\partial z^2}\ \dfrac{\partial z^2}{\partial \sigma^1}\ \dfrac{\partial \sigma^1}{\partial z^1}\ \dfrac{\partial z^1}{\partial b^1} \end{cases} \tag{2.14}$$

$\dfrac{\partial BCE}{\partial \sigma^2}\ \dfrac{\partial \sigma^2}{\partial z^2}$ est calculé dans la premier phase lors de la mis à jour des poids de neurone de sortie, donc il faut calculer que le reste des dérivées.

Remarque 2.6.1 *L'utilisation d'une fonction d'activation qui donne des valeurs entre 0 et 1 en apprentissage en profondeur*[2]*, affecte des gradients très petits, ce qui stagne l'apprentissage, par exemple l(utilisation de la fonction sigmoid dans tous les neurones. Ce type de problème est appelé disparation de gradient.*

2.6.2 Réseau perceptron multicouches

Le réseau perceptron multicouches MLP (Multi Layer Perceptron) est un type d'architecture de réseau de neurones artificiels à propagation avant [3](feedforward). Dans ce type, les neurones sont organisés en couches, une couche d'entrée, une ou plusieurs couches intermédiaires appelées couches cachées et une couche de sortie, la figure 2.12, donne l'exemple d'un réseau perceptron multicouche prend une entrée de taille 4 et contenant deux couches cachées et une couche de sortie, chaque couche se compose d'un ou plusieurs neurones, pour la couche d'entrée, elle représente toujours une couche virtuelle associée aux entrées du réseau, elle ne contient aucun neurone tandis que les couches suivantes représentent des couches effectives de neurones.

2. L'utilisation d'un grand nombre de neurones en séquence
3. un réseau de neurones artificiel dans lequel les connexions entre les perceptrons ne forment pas un cycle

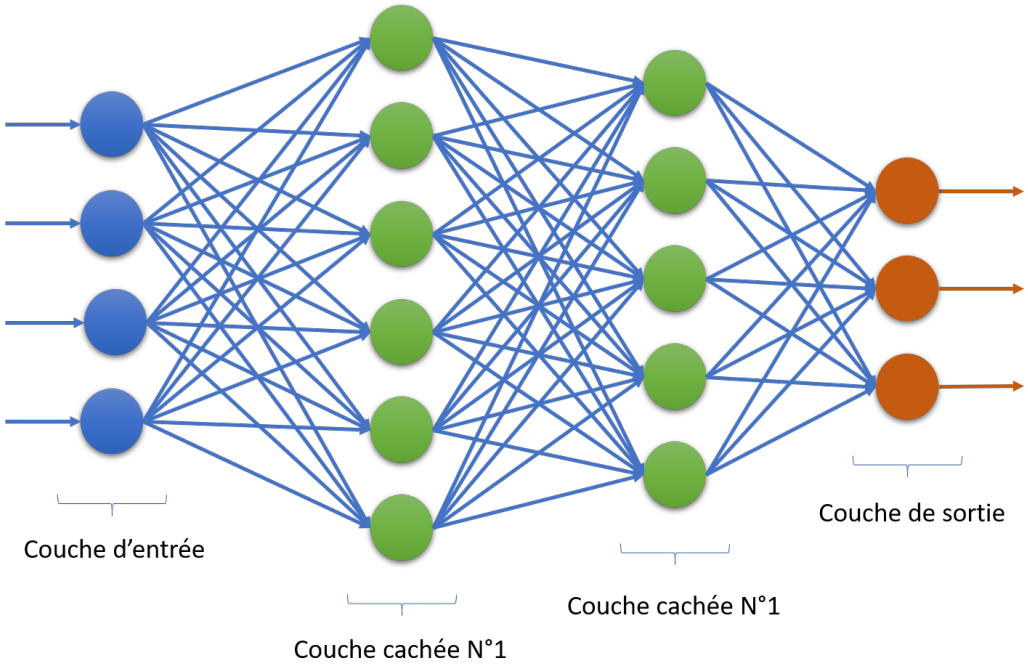

FIGURE 2.11 – Un exemple de réseau de neurones de type perceptron multicouches

Dans un réseau perceptron multicouches, nous pouvons définir une fonction d'activation pour chaque couche, où les neurones de la même couche prennent la même fonction d'activation. Les réseaux MLP utilisent la rétro-propagation dans leur formation.

2.6.3 L'incorporation de mots et L'architecture word2vec

2.6.3.1 Word Embedding

Le Word Embedding, connu également sous le nom d'incorporation de mots, est une technique permettant de convertir des mots textuels en une représentation numérique sous forme de vecteur. Chaque mot est associé à un vecteur spécifique qui vise à saisir différentes caractéristiques du mot par rapport à l'ensemble du texte, telles que les relations sémantiques, les définitions et le contexte. Une fois ces représentations numériques (vecteurs) obtenues, il est possible d'effectuer diverses opérations, telles que l'identification de similarités et la dissemblance entre les mots ou d'effectuer des opérations vectorielles telles que l'addition et la soustraction de mots. voir la figure suivant.

```
Roi - Homme + Femme = Reine
[5,3] - [2,1] + [3, 2] = [6,4]
```

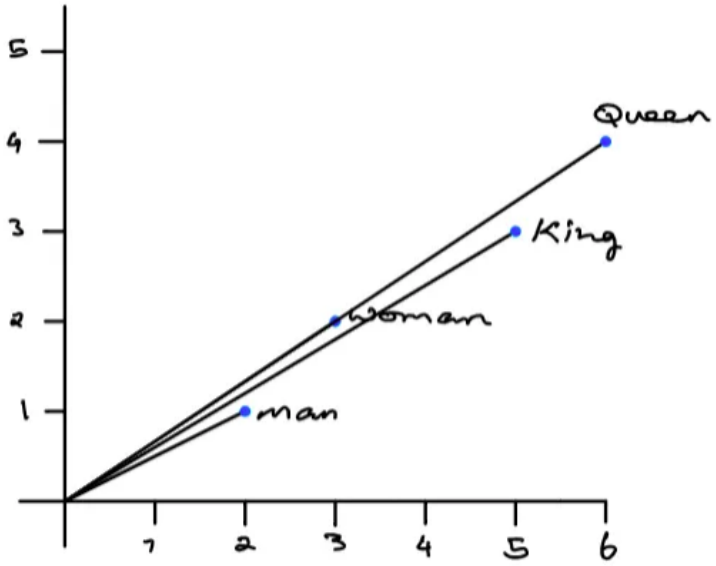

FIGURE 2.12 – Représentation des mots en vecteurs

Remarque 2.6.2 *L'encodage one hot n'est pas une technique de word embding car les vecteurs qui serons en sortie ne permettent pas d'identifier les carctéristiques des mots tel que les relations sémentiqu les déffinition et le context.*

2.6.3.2 La techenique Word2Vect

Word2Vec [15] est une technique de Word Embedding, elle permet de générer des représentations vectorielles (embeddings) pour les mots d'un corpus de texte. Ces modèle ont été développés par une équipe de recherche chez Google sous la direction de Tomas Mikolov. Word2Vec est un réseau de neurones artificiels à deux couches entraînés pour reconstruire le contexte linguistique des mots. Ce modèle peut être entraîné avec deux architectures principales : le modèle Skip-Gram et le modèle Continuous Bag-of-Words (CBOW). Le modèle Skip-Gram prédit les mots environnants à partir d'un mot cible, tandis que le modèle CBOW prédit le mot cible à partir de mots environnants. Les deux architectures sont couramment utilisées pour générer des embeddings de mots avec Word2Vec.

Dans les deux type d'architecteur, le réseau de neurones comporte deux couches. La couche cachée contient quelques centaines de neurones et constitue, à l'issue de la représentation, l'incorporation de mots (embedding) permettant de représenter un mot. La couche de sortie permet d'implémenter une tâche de classification binaire.

En raison de sa simplicité et de son temps d'entraînement plus rapide, le modèle CBOW est souvent préféré pour les corpus de texte plus petits. Dans ce travaille nous intérrésons au modèle CBOW. Le modèle Skip-Gram est un simple réseau de neurones avec une couche cachée entraînée afin de prédire la probabilité qu'un mot donné soit présent lorsqu'un mot d'entrée est présent.

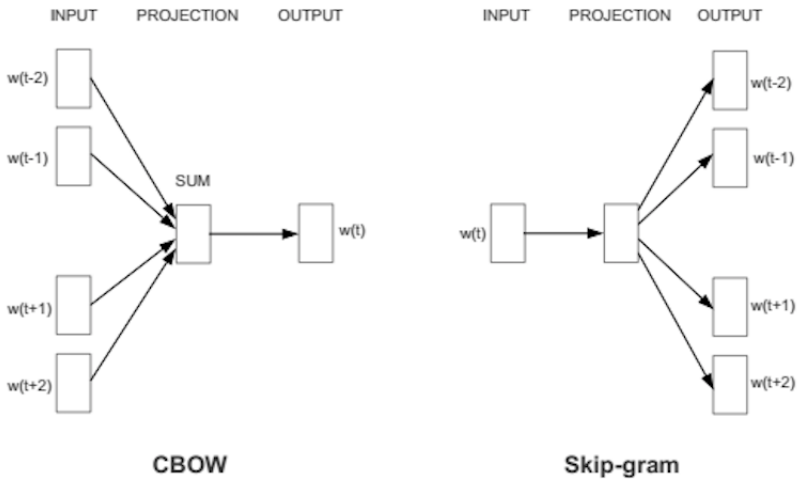

FIGURE 2.13 – L'architecture de modèle CBOW et le modèle Skip-Gram

Pour le reste de la discussion, nous nous limiterons au modèle Skip-gram car elle produit des résultats plus précis sur de grands ensembles de données [4].

Définition 2.6.3 *Taille de la fenêtre*

La taille de la fenêtre est un paramètre dans l'algorithme qui désigne le nombre de mots voisins [5] *d'un mot ciblé. Ces mots voisins sont considéré comme le contexte de mot ciblé, si la taille de la fenêtre est égale à n ceci signifie qur n mots derrière et n mots devant le mot ciblé seront considérés (Voir figure 2.16). Pratiquement, une taille de fenêtre peut être de 5 jusqu'à 10.*

Le Skip-gram est un algorithme qui prend en entrée un mot et produit en sortie un ensemble de mots avec une mesure de similarité pour chaque mot. Cette mesure de similarité définit les pourcentages ou les poids de compatibilité de contexte entre le mot en entrée et les mots en sortie.

Exemple 2.6.4 *Dans la phrase de la figure 2.16, le mot "back-alley" est notre mot actuel, et les mots "little", "dark", "behind" et "the" sont les mots que nous voudrions prédire avec ce mot (Voir figure 2.15).*

4. https ://skymind.ai/wiki/word2vec
5. Les mots voisins sont les mots qui sont derrière et devant le mot ciblé

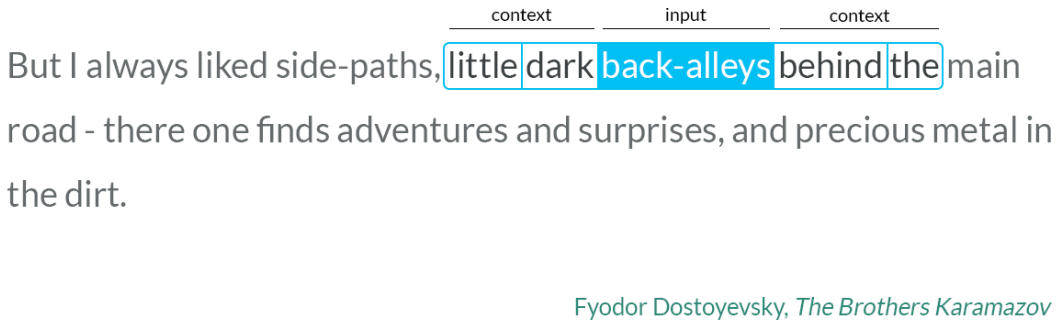

FIGURE 2.14 – Représentation les voisins (contexte) d'un mot cible (input) tel que la taille de la fenêtre est égale à 2

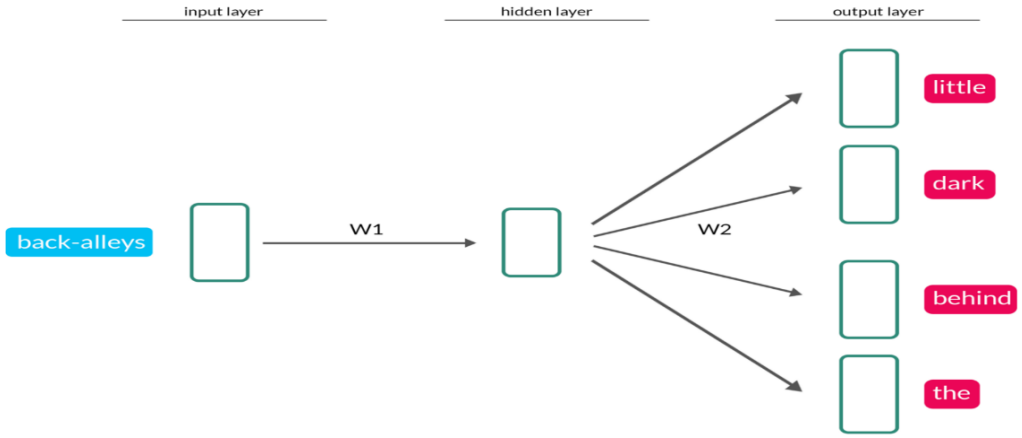

FIGURE 2.15 – Illustration de la prédiction de similarité de mot back-alley

- L'élaboration de modèle skip-gram passe par plusieurs étapes (ou sous-processus) :

(A) Première étape consiste à définir une taille de fenêtre, on suppose que la taille de la fenêtre est égale à n.

(B) La deuxième étape consiste à former des tuplet $(w, v_1, ..v_i, ..v_{2n})_{i\in\{1,..,Tailledevocabulaire\}}$, où w représente le mot ciblé et v_i correspond à ses voisins (Voir figure 2.16). Pendant la phase d'apprentissage, le mot w est utilisé en tant qu'entrée du réseau de neurones et les mots v_i sont utilisés comme sorties.

(C) Troisième étape a pour but de représenter chaque mot de vocabulaire en tant que vecteur encodé à un bit non nul (one-hot vector) de détention égale à la taille de vocabulaire, tel que nous placerons un 1 dans la position correspondant à la position de mot cible dans le corpus d'entraînement et des 0 dans toutes les autres positions (Voire tableau 2.1).

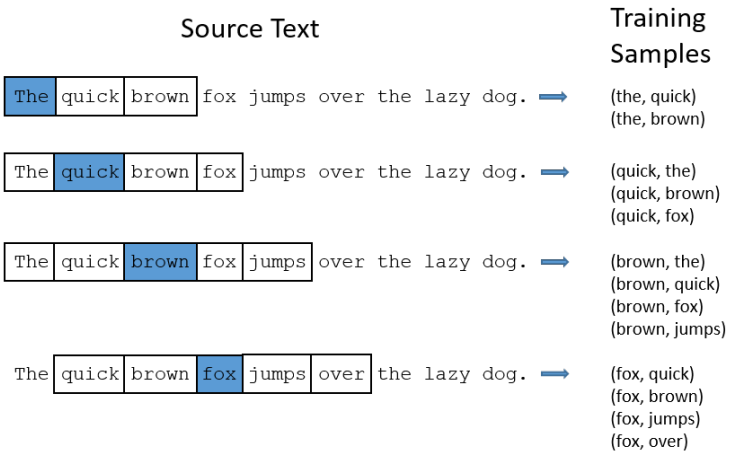

FIGURE 2.16 – Construction des couples (mot, voisin) avec la taille de la fenêtre $n = 2$

word	word one hot encoding	neighbor	word one hot encoding)
The	[1,0,0,0,0,0,0,...,0]	quick	[0,1,0,0,0,0,0,...,0]
The	[1,0,0,0,0,0,0,...,0]	brown	[0,0,1,0,0,0,0,...,0]
quick	[0,1,0,0,0,0,0,...,0]	The	[1,0,0,0,0,0,0,...,0]
quick	[0,1,0,0,0,0,0,...,0]	brown	[0,0,1,0,0,0,0,...,0]
quick	[0,1,0,0,0,0,0,...,0]	fox	[0,0,0,1,0,0,0,...,0]
brown	[0,0,1,0,0,0,0,...,0]	The	[1,0,0,0,0,0,0,...,0]
brown	[0,0,1,0,0,0,0,...,0]	quick	[0,1,0,0,0,0,0,...,0]
brown	[0,0,1,0,0,0,0,...,0]	fox	[0,0,0,1,0,0,0,...,0]
brown	[0,0,1,0,0,0,0,...,0]	jumps	[0,0,0,0,1,0,0,...,0]
fox	[0,0,0,1,0,0,0,...,0]	quick	[0,1,0,0,0,0,0,...,0]
fox	[0,0,0,1,0,0,0,...,0]	brown	[0,0,1,0,0,0,0,...,0]
fox	[0,0,0,1,0,0,0,...,0]	jumps	[0,0,0,0,0,1,0,...,0]
fox	[0,0,0,1,0,0,0,...,0]	over	[0,0,0,0,0,1,0,...,0]

TABLE 2.1 – Représentation des mots avec des vecteurs encodés à un bit non nul

(D) Quatrième étape est une phase d'apprentissage de modèle sur les paires de vecteurs construits dans l'étape précédent. La formation du réseau de neurones consiste à apprendre les valeurs des poids de la couche caaché qui permet l'incorporation de mots, les valeurs de poids sont représenter par une matrice **W**. La matrice de poids **W** est une matrice comportant autant de lignes que de mots dans notre vocabulaire, chaque ligne contenant les poids associés à un mot particulier. Par conséquent, étant donné que des mots similaires doivent générer des prédictions similaires, leurs lignes dans la matrice **W** doivent être similaires.

(E) La cinquième et dernière étape consiste en une étape de prédiction. Pour incorporer un mot spécifique, il est nécessaire d'extraire la matrice **W** à partir du réseau de neurones préalablement entraîné. Ensuite, il suffit de multiplier le vecteur encodé à un bit ()one-hot) du mot par la matrice **W** afin de générer le vecteur d'incorporation (Voir figure 2.17).

$$[0 \quad 0 \quad 0 \quad 1 \quad 0] \times \begin{bmatrix} 17 & 24 & 1 \\ 23 & 5 & 7 \\ 4 & 6 & 13 \\ 10 & 12 & 19 \\ 11 & 18 & 25 \end{bmatrix} = [10 \quad 12 \quad 19]$$

FIGURE 2.17 – Exemple de projection d'une ligne de la matrice à partir de vecteur encoder à un bit non nul

2.6.3.3 Le calcul du similarété

Définition 2.6.5 *Similarité*

La similarité est la mesure de la ressemblance de deux objets de données. La similarité généralement décrite comme une distance dont les dimensions représentent les caractéristiques des objets. Une petite distance indiquant un degré élevé de similitude et une grande distance indiquant un faible degré de similitude. La similarité est subjective et dépend fortement du domaine et de l'application [16].

Pour évaluer la similarité entre deux mots, il est possible d'obtenir leurs vecteurs d'incorporation et de calculer la distance entre ces deux vecteurs. Différentes fonctions mathématiques peuvent être utilisées pour calculer la distance entre les vecteurs, telles que la similarité cosinus, distance de Jaccard et coefficient de Dice. Le résultat fourni par les fonctions est un nombre qui représente la distance entre les deux vecteur.

$$\text{similarité cosinus}(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B} == \sum_{i=1}^{n} A_i \cdot B_i}{\|\vec{A}\| \|\vec{B}\|}$$

2.7 Conclusion

Au cours de ce chapitre, après avoir fourni une introduction qui clarifie le sujet général du chapitre, nous avons décrit les définitions de base de l'apprentissage automatique, les différents types de problèmes qu'il aborde et les trois techniques d'apprentissage automatique à savoir l'apprentissage supervisé, non supervisé et par renforcement. Par la suite, nous avons évoqué les algorithmes d'apprentissage automatique les plus populaires et les méthodes d'évaluation utilisées pour mesurer les performances des modèles d'apprentissage générés. Dans la dernière

section de ce chapitre, nous avons décrit en détail l'algorithme des réseaux de neurones artificiels ainsi que la technique de similarité basée sur l'architecture de réseau de neurones Word2vec. Grâce à cela, nous disposons de toutes les informations nécessaires pour la mise en œuvre de notre système qui sera abordée dans le chapitre suivant.

Chapitre 3

Implémentation tests et résultats

Définition 3.0.1 *Curriculum vitae*

Le curriculum vitae (en abrégé CV) est un document détaillant le parcours scolaire et/ou professionnel et autres compétences acquises d'un individu, son rôle se situe davantage au niveau de la recherche d'un emploi [17].

Dans ce chapitre nous présentons système de recommandation qui mettre en relation les candidats (CVs) avec les missions correspondantes des postes de travail. Le système repose sur une architecture de réseau neuronal artificiel.

Dans la première phase, le système reçoit le CV du candidat au format PDF (non structuré) et effectue un parsing pour convertir le PDF en texte brut. Ensuite, il segmente le CV en un ensemble de paragraphes, puis applique un prétraitement spécifique à ces paragraphes. En utilisant un réseau de neurones (classificateur), il attribue des étiquettes décrivant le contexte de chaque paragraphe. Le résultat obtenu est un CV semi-structuré. Le même processus est appliqué à l'offre d'emploi pour obtenir une offre d'emploi semi-structurée.

Une fois que le format semi-structuré est obtenu, le système passe au calcul de la similarité entre les paragraphes du CV et de l'offre d'emploi qui partagent le même contexte. Pour cela, il utilise un autre réseau de neurones appelé word2vec. Enfin, le score final est calculé en prenant la moyenne des résultats de similarité entre les paragraphes.

L'architecture générale de notre système de recommandations est donnée par la figure 3.3.

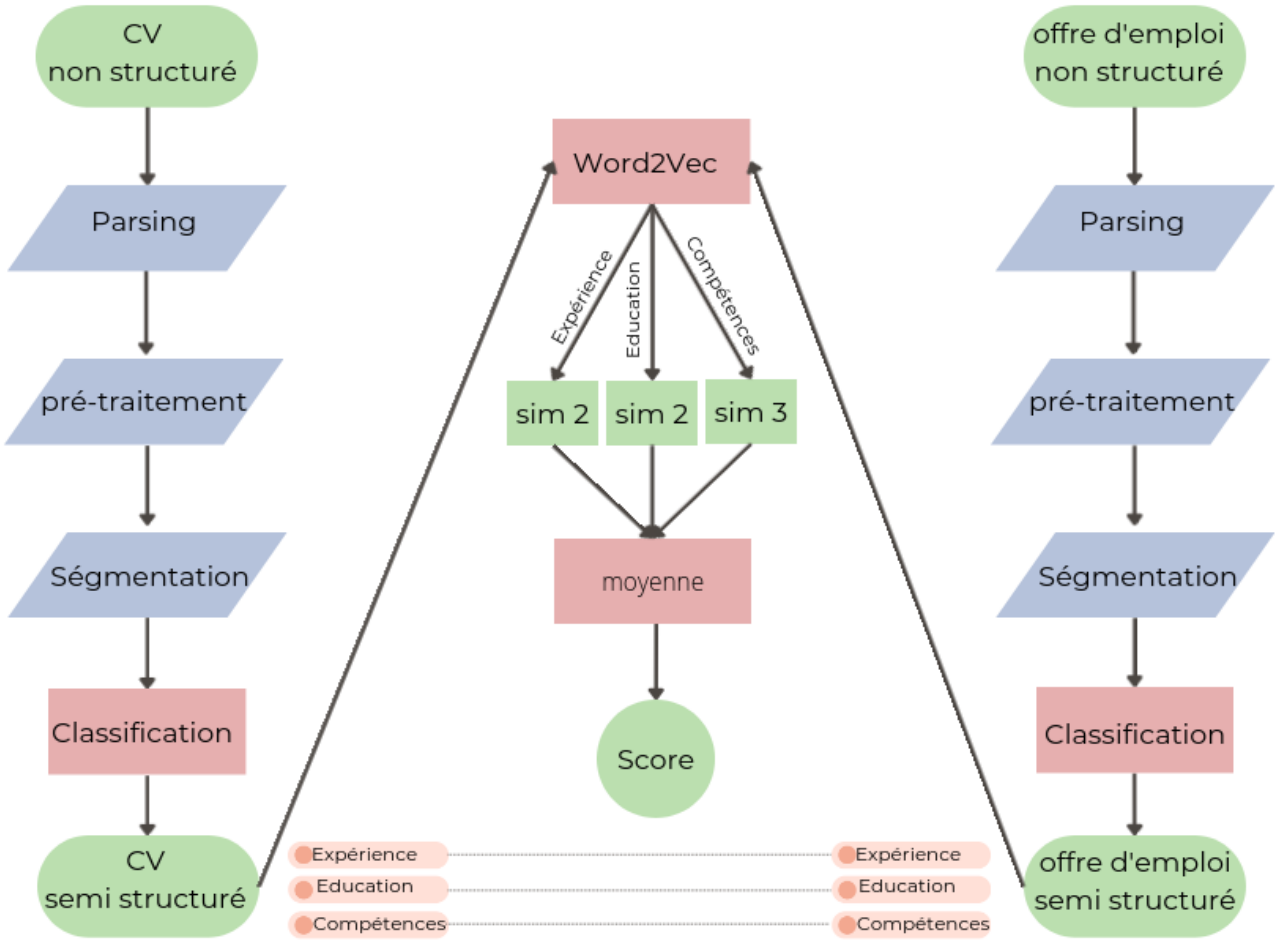

FIGURE 3.1 – Architecture de SEEKJOBS

3.1 Le parsing

Les CV des demandeurs d'emploi et les offres d'emploi sont généralement au format PDF, mais notre système nécessite une analyse du contenu texte brut pour un traitement adéquat. Pour extraire le texte des fichiers PDF, nous avons utilisé la bibliothèque Python PDFminer. Donc, cette étape vise uniquement à convertir les fichiers PDF en texte brut.

3.2 Le prétraitemet

Définition 3.2.1 *mots vide*

Les mots vides sont toute unité de langage qui n'a pas de valeur dans la compréhension d'un texte comme les prépositions, les articles et les pronoms. Ces mots ne sont pas pris en considération par un langage documentaire(artificiel) [1].

1. Les langages artificiels sont créés pour indexer des documents. C'est un procédé conventionnel de représentation des informations d'un document sous une forme condensée et normalisée.

Une fois le texte brut extrait lors de l'étape de parsing, il est nécessaire de lui appliquer un prétraitement spécifique. Ce prétraitement comprend la normalisation en minuscules, la suppression des chiffres et des mots vides (stop words).

3.3 La ségmentation

Une fois le prétraitement terminé, nous procédons à l'étape de segmentation. Cette étape consiste à prendre en entrée le texte nettoyé et à retourne en sortie une liste de paragraphes en utilisant deux sauts de ligne "\n\n" pour séparer le texte d'entrée en un enssemble de paragraphes.

Remarque 3.3.1 *Dans ce travail, nous utilisons le terme "segment" pour désigner un paragraphe non classé.*

3.4 Entraînement de modèles de réseaux de neurones

Pour mettre en œuvre notre système, il est nécessaire de créer deux modèles de réseau de neurones. Le premier modèle est un classificateur qui attribue une étiquette (context) à chaque segment (paragraphe), tandis que le second modèle est utilisé pour calculer le score de similarité entre les différents segments.

3.4.1 La collection de données

Définition 3.4.1 *web scrapping*

C'est une technique utilisée pour extraire des données des sites web. Il s'agit d'un processus automatisé dans lequel une application traite le code HTML d'une page web afin d'extraire des données

Avant de procéder à l'entraînement de nos modèles de réseaux neuronaux, nous devons disposer d'un ensemble de données. Pour cela, nous avons utilisé la technique du web scraping afin de collecter un enssemble de CV à partir du site web Indeed [2].

En examinant les CVs dans le site web, nous constatons que le titre de chaque paragraphe décrit précisément le contexte de celui-ci, tel que nous pouvons utiliser ces titres comme étiquettes pour chaque paragraphe. le format de CV dans le site resume.indeed est présenté dans la figure suivante.

2. https ://resumes.indeed.com/

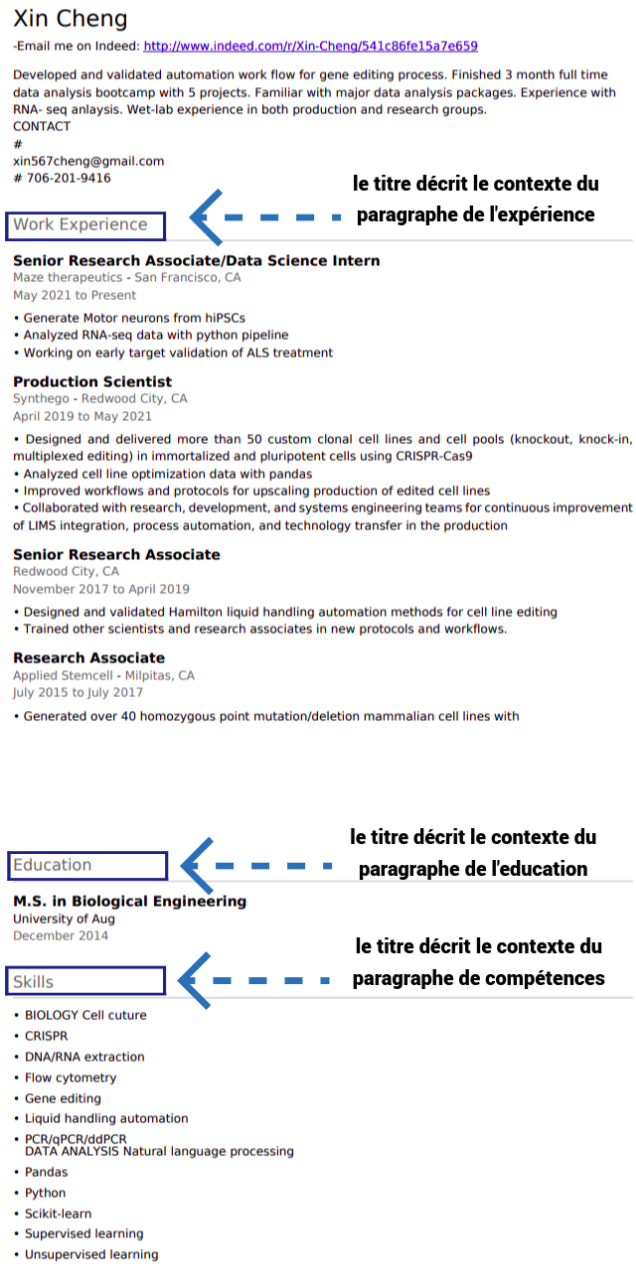

FIGURE 3.2 – la form d'un CV dans le site resume.indeed

Lors de la collecte des CVs, les informations sont sauvegardées séparément dans des fichiers JSON. Chaque fichier JSON représente un CV et utilise des clés pour identifier les différentes sections du CV, tandis que les valeurs associées à ces clés représentent le contenu de chaque section (paragraphe). La figure suivante illustre un exemple de CV au format JSON :

```
{
    "Work Experience" : "Senior Research Associate/Data Science Intern Maze therapeutics -
                        San Francisco, CA May 2021 to Present Generate Motor neurons from hiPSCs Analyzed RNA-seq data with python pipeline Working
                        on early target validation of ALS treatment Production Scientist Synthego - Redwood City, CA April 2019 to May 2021 Designed
                        and delivered more than 50 custom clonal cell lines and cell pools (knockout, knock-in, multiplexed editing) in immortalized
                        and pluripotent cells using CRISPR-Cas9 Analyzed cell line optimization data with pandas Improved workflows and protocols for
                        upscaling production of edited cell lines Collaborated with research, development, and systems engineering teams for continuous
                        improvement of LIMS integration, process automation, and technology transfer in the production enior Research Associate Redwood
                        City, CA November 2017 to April 2019 Designed and validated Hamilton liquid handling automation methods for cell line editing
                        Trained other scientists and research associates in new protocols and workflows. Research Associate Applied Stemcell - Milpitas,
                         CA July 2015 to July 2017 Generated over 40 homozygous point mutation/deletion mammalian cell lines with",

    "Education" :  "M.S. in Biological Engineering University of Aug December 2014",

    "Skills" : "BIOLOGY Cell cuture CRISPR DNA/RNA extraction Flow cytometry Gene editing Liquid handling automation PCR/qPCR/ddPCR
                DATA ANALYSIS Natural language processing Pandas Python Scikit-learn Supervised learning Unsupervised learning"
}
```


FIGURE 3.3 – CV au format JSON

Dans cet exemple, les clés telles que "Work Experience", "Education", et "Skills" représentent les différentes sections du CV. Les valeurs associées à ces clés (étiquettes) contiennent les informations spécifiques à chaque section. Cela permet de stocker les informations de chaque CV de manière structurée dans des fichiers JSON,

À la fin de la collecte, nous avons converti l'ensemble des fichiers JSON en un seul fichier CSV, ce qui simplifie leur lecture, leur traitement et leur analyse ultérieure. Un aperçu du contenu du fichier CSV est présenté dans la figure suivante.

	Unnamed: 0	text	classe
0	152	B.S. in Computer ScienceUniversity of Oregon E...	Education
1	555	Carmel, IN	Location
2	163	Guard CardPresentSecurity	Certifications
3	65	Software EngineerSpring BootOctober 2015 to Pr...	Experience
4	664	Security Officer	Title
5	1526	Master of Data Science in Machine Learning Fea...	Education
6	406	SkillsJanitorial, Houskeeping, Cashier, custom...	Skills
7	595	Software EngineerFast Enterprises - Olympia, W...	Experience
8	885	Network Administrator	Title
9	408	Pharmacy TechnicianSeptember 2014 to PresentCe...	Certifications
10	1236	Network AdministratorPermianLide - Odessa, TXM...	Experience
11	1323	Belton, TX	Location
12	363	Driver's License	Certifications
13	545	Software Engineer - Technical LeadMarlabs Inc ...	Experience
14	474	KEY QUALIFICATIONS• Information Systems Securi...	Skills

FIGURE 3.4 – Aperçu du contenu du fichier CSV

Sur une collection de CV de plus de 3000, voici quelques statistiques tirées de notre jeu de données.

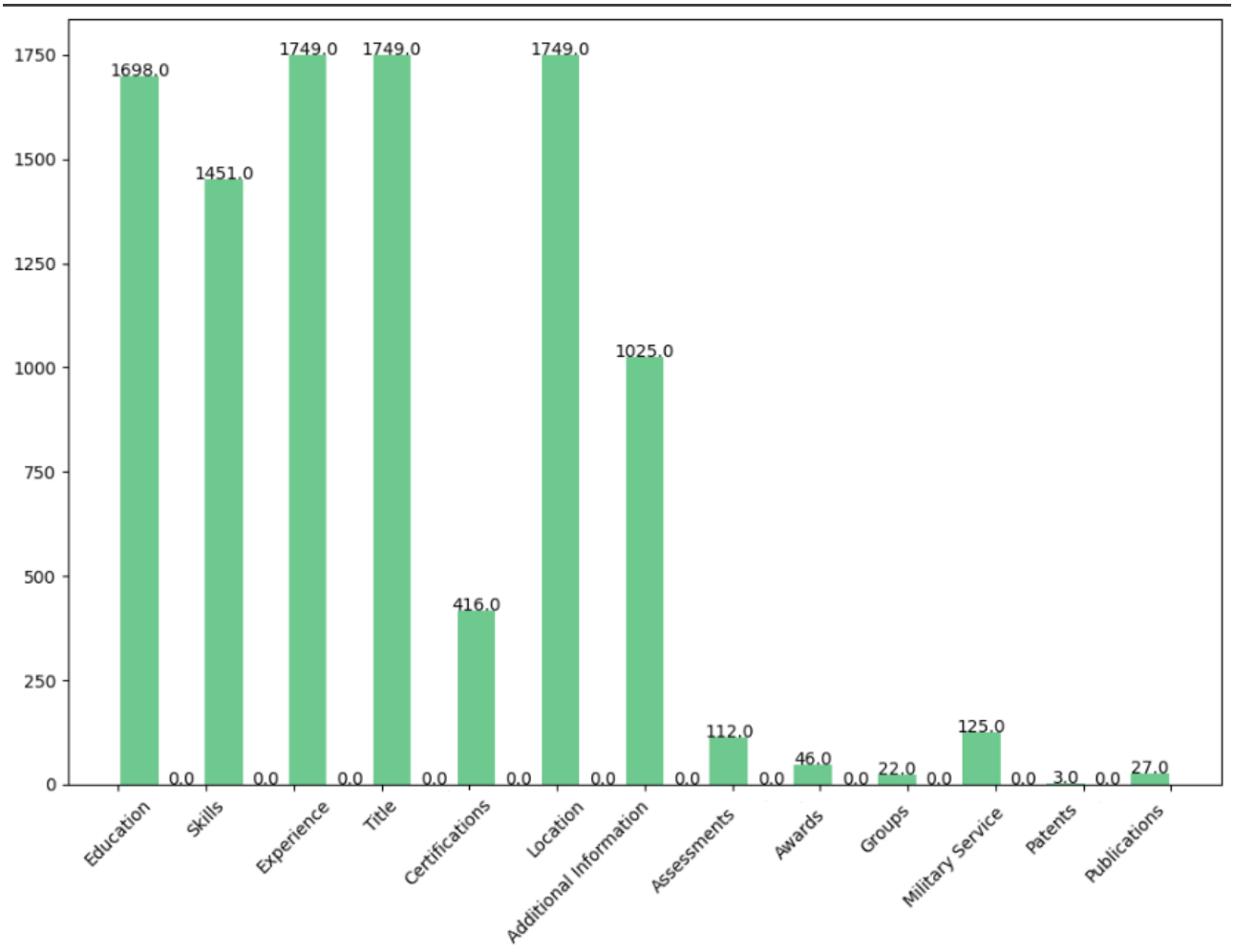

FIGURE 3.5 – Histogram represents the number of text instance according to the label

Afin de simplifier le processus d'entrainement, nous nous intéressons dans ce travail aux trois labels "Education" (formation), "Experience" (expérience) et "Skills" (compétences).

3.4.2 prétraitement de données

"*Data is the new oil. It's valuable, but if unrefined it cannot really be used... It must be broken down and analyzed for it to have value.*"— Clive Humby

Cette étape consiste à représenter le format textuel des données collectés dans l'étape précédente en une forme plus simple. Pour réaliser cette étape, il faut éliminer en premier tous les mots vides, les chiffres et les mots insigniants, ansi convertir tous les mots en minuscules.

La prochaine étape consiste à indexer les documents, c'est-à-dire à transformer le document sous forme de texte en un vecteur de mots, cette représentation est appelée modèle d'espace vectoriel. L'ensemble de documents et l'ensemble de mots sont représenté sous la forme d'une matrice où chaque case de la matrice contient le poids w_{ij} du mot m_j dans le document d_i (Voir figure 3.6). Il existe plusieurs manières de déterminer le poids comme la pondération booléenne, la pondération fréquentielle des mots, tf-idf, etc. Dans notre travail, Nous avons appliqué une

pondération booléenne pour représenter les vecteurs de mots. Cette pondération attribue la valeur 0 si un mot existe dans un document et 1 s'il est absent.

$$\begin{pmatrix} & m_1 & m_2 & ... & m_N \\ d_1 & w_{11} & w_{12} & ... & w_{1N} \\ d_2 & w_{21} & w_{22} & ... & w_{2N} \\ : & : & : & & : \\ d_M & w_{M1} & w_{M2} & ... & w_{MN} \end{pmatrix}$$

FIGURE 3.6 – La représentation vectorielle des documents

Après avoir effectué ce traitement, nous avons obtenu une matrice de dimensions $5923 * 42826$. Le nombre de lignes de la matrice correspond au nombre d'exemples ou d'instances de données collectées, tandis que le nombre de colonnes correspond au nombre de mots distincts présents dans notre dataset.

En suite, il est nécessaire de coder toutes les étiquettes numériquement, ce qui permettra d'entraîner le réseau de neurones.

1. 'experience' : 001
2. 'education' : 010
3. 'skills' : 100

À la fin, l'ennsemble de 5923 exemples de textes étiquetés est divisé en un ensemble d'apprentissage et un ensemble de test.

1. enssemble d'apprentissage : 4000
2. enssemble de test : 1923

3.4.3 Le modèle classificateur

Le modèle classificateur est un simple réseau de neurones multicouche dont l'objectif est de prédire le contexte d'un paragraphe donné en entrée, c-t-d que ce modèle permet d'étiqueté le paragraphe soit comme "Education" (formation), "Experience" (expérience) ou "Skills" (compétences). l'architecteur de réseau de neurone est présenté dans la figure 3.7.

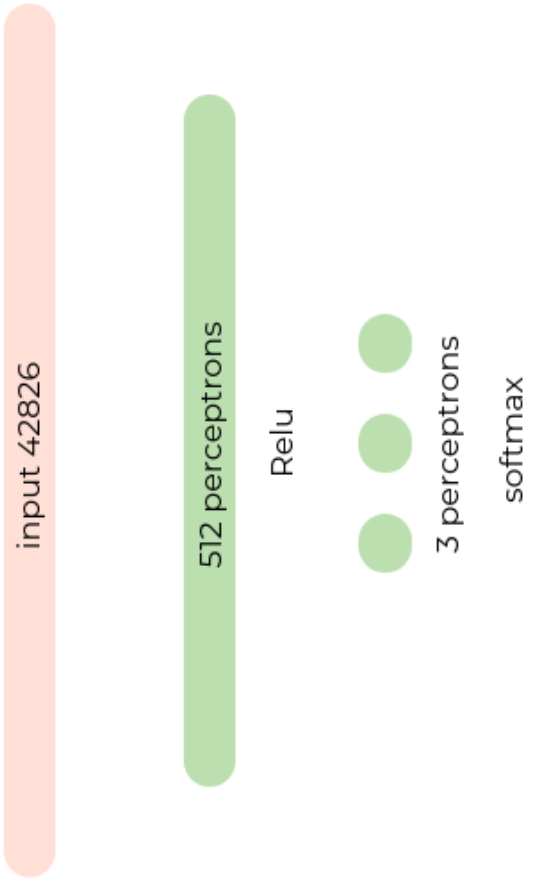

FIGURE 3.7 – Aperçu du contenu du fichier CSV

Le modèle que nous avons mis en place comprend une architecture en plusieurs couches. La première couche est la couche d'entrée, dont la taille est de 42826. Cette taille correspond au nombre de mots présents dans notre jeu de données. Ensuite, nous avons une seule couche cachée qui est composée de 512 perceptrons. Chaque perceptron utilise une fonction d'activation ReLU. Enfin, la couche de sortie est composée de 3 perceptrons, chacun utilisant une activation softmax. Cette configuration permet à chaque perceptron de prédire une probabilité pour une classe spécifique : "Education" (formation), "Experience" (expérience) ou "Skills" (compétences). La fonction softmax assure que les probabilités prédites par les perceptrons se situent entre 0 et 1 et que leur somme est égale à 1.

Pour l'entraînement, nous optons pour la fonction de perte Categorical Cross-Entropy étant donné que notre problème de classification implique trois classes différentes (classification multi-class).

Les résultats de l'accuracy et l'erreu de perte obtenu après l'netrainement du réseau de neurone pour les données d'entrainement et les donnés de test sont représenté dans le tableau suivant.

données : métrique	accuracy	Categorical Cross-Entropy
Sonnées d'ntrainement	99.9%	0.00034
Données de test	96.8%	0.2902

TABLE 3.1 – Les résultats du test du réseau de neurones classificateur

Text	classe réelle	prédiction
'in Criminal JusticeTexas State University January 1995 to January 1999'	Education	Education (99%)
'.NET (4 years), .NET Framework 4.0 (Less than 1 year), C# (4 years), databases (4 years), HTML (4 years)'	Skills	Skills (100%)
'Software EngineerTATA Consultancy servicesJanuary 2006 to August 2015from Aug 10th, 2015 to Jan 6 th, 2017. Working as a Software Engineer in Concentrix from Jan 23rd, 2017 to till date.'	Experience	Experience (99.99%)

TABLE 3.2 – Les résultats du prédiction du réseau de neurones classificateur

Le tableau ci-dessous 3.1 affiche les résultats d'accuracy et de perte obtenus après l'entraînement du réseau de neurones, à la fois pour les données d'entraînement et les données de test. Le tableau 3.2 présente des exemples de prédictions réalisées par notre modèle classificateur.

3.4.4 Le modèle Word2vec

Nous avons entraîné notre modèle Word2Vec en utilisant la colonne "texte" qui contient les différents paragraphes de notre ensemble de données. Pour cette étape, nous n'avons pas besoin de la colonne "classe". Nous avons utilisé le framework Gensim pour créer et entraîner notre modèle Word2Vec, en choisissant une taille de fenêtre de 5.

Pour évaluer notre modèle Word2Vec, nous pouvons effectuer des expérimentations telles que calculer le score de similarité entre des mots et vérifier si les résultats sont cohérents. Nous pouvons également afficher les mots les plus similaires à un mot spécifique. les résultatu d'evaluation est donnée pas le tableau 3.3.

Une fois que notre modèle Word2Vec a été entraîné et testé, nous pouvons extraire la matrice de poids de sa première couche cachée. Cette matrice de poids peut ensuite être utilisée pour incorporer les mots, c'est-à-dire de récupérer leurs vecteurs de caractéristiques (de sémantiques). Pour calculer la similarité entre deux paragraphes, nous devons d'abord déduire

Text	python	javascript	cybersecurity	malware
python	1.0	0.88	0.55	0.50
javascript	0.88	1.0	0.54	0.59
cybersecurity	0.55	0.54	1.0	0.83
malware	0.50	0.59	0.83	1.0

TABLE 3.3 – Les résultats du prédiction du réseau de neurones classificateur

leur vecteur d'incorporation en utilisant les vecteurs d'incorporation de leurs mots. Le vecteur d'incorporation du paragraphe est obtenu en calculant la moyenne des vecteurs d'incorporation des mots qui le composent.

Exemple 3.4.2 *Pour obtenir le vecteur de caractéristiques d'un paragraphe, noté $\vec{D}$, à partir des vecteurs de caractéristiques de ses mots, notés $A_1, A_2, ..., A_n$, où la taille de chaque vecteur est égale à K, nous pouvons utiliser comme méthode la moyenne des vecteurs de mots.*

$$\vec{D} = moyenne(\vec{A_1}, ..., \vec{A_n}) = \{\frac{\sum_{i=1}^{n} A_{ij}}{K}, \forall 1 <= j <= K\}$$

Le résultat obtenu est un vecteur de caractéristiques $\vec{D}$ de dimension K,

Une fois que nous avons obtenu le vecteur d'incorporation pour chaque paragraphe, nous pouvons calculer leur score de similarité en utilisant la fonction de similarité cosinus.

$$\text{similarité cosinus}(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B} == \sum_{i=1}^{n} A_i \cdot B_i}{\|\vec{A}\| \|\vec{B}\|}$$

3.4.5 Calcul de score de similarité entre un CV et un offre d'emploi

Après avoir terminé les étapes de parsing, de prétraitement et de segmentation, nous utilisons le modèle classificateur pour attribuer chaque segment (paragraphe) à l'une des trois catégories suivantes : "Education" (formation), "Experience" (expérience) et "Skills" (compétences). En résultat, nous obtenons une liste de paragraphes classifiés. Ensuite, nous regroupons ces paragraphes en fonction de leurs classes, formant ainsi trois groupes de texte distincts : un pour l'éducation, un pour l'expérience et un autre pour les compétences. Ce processus s'applique aussi bien aux CV qu'aux offres d'emploi. Par conséquent, en réalité, nous obtenons deux groupes de texte pour chaque catégorie, soit un total de six groupes.

Après cela, nous procédons au calcul de la similarité entre les deux groupes de texte pour chaque classe $(text_CV(class_i), text_offre_emploi(class_i))$ où $i \in Education, Experience, Skills$, en utilisant la méthode décrite dans la section Word2Vec.

Enfin, pour obtenir le score de similarité entre un CV et une offre d'emploi, nous prenons la moyenne des trois scores obtenus pour les trois classes respectives.

3.5 Mise Correspondance entre les CV et les offres d'emploi

Pour effectuer la mise en correspondance entre les candidats (CV) et les missions (offres d'emploi), il suffit d'appliquer la méthode décrite précédemment à chaque paire de CV et d'offre d'emploi afin de calculer leur score de similarité. Ensuite, pour chaque CV_i, nous sélectionnons le couple (CV_i, offre d'emploi) ayant le score le plus élevé parmi tous les tuples (CV_i, offre d'emploi, score) et vice-versa pour un $OFFRE\ EMPLOI_j$.

3.6 Outils de réalisation

Cette section décrit les outils et langages de programmation que nous avons employés pour développer notre système. Nous aborderons également l'outil utilisé pour la conception du notre application web POC (Proof of Concept).

3.6.1 Outils et langages de programmation utilisé :

3.6.1.1 Python :

Python est un langage de programmation de haut niveau, interprété, et orienté objet. Il a été créé par Guido van Rossum et a été publié pour la première fois en 1991. Python est conçu pour être facile à lire, avec une syntaxe claire et simple [**?**].

Pourquoi Python ?

- **La simplicité** : la structure simple de code python.
- **Multi-plateforme** :python fonctionne sur tous les systèmes d'exploitation.
- **Multi-fonction :** python est principalement utilisé dans trois domaines(développement web , les sciences de donnée , scripting).
- **Populaire** : grâce a sa communauté ce qui garantie la disponibilité de la documentation et des réponses sur les différentes problèmes rencontrés lors du développement.

3.6.1.2 Selenium :

Un framework de test et automatisation. Il permet d'interagir avec différents navigateurs web de même que le ferait un utilisateur de l'application.

3.6.1.3 BeautifulSoup :

C'est une bibliothèque Python permettant d'extraire des données de fichiers HTML et XML.

3.6.1.4 NLTK :

Natural Language Toolkit est une bibliothèque Python permettant un traitement automatique du langage naturel (humain), développée par Steven Bird et Edward Loper du département d'informatique de l'Université de Pennsylvanie.

3.6.1.5 Kease et Tensorflow

"Keras" et "TensorFlow" sont deux bibliothèques populaires d'apprentissage automatique en Python. Keras est une interface haut niveau qui facilite la création, la configuration et l'entraînement de réseaux de neurones. Il offre des outils et des fonctionnalités avancés pour développer des modèles d'apprentissage profond et utilise TensorFlow comme moteur de calcul sous-jacent.

3.6.2 Gensim

Gensim est une bibliothèque Python populaire dédiée à la modélisation de sujets et au traitement du langage naturel (NLP). Elle offre une solution simple et efficace pour traiter de grandes collections de données textuelles, effectuer des tâches avancées de NLP et extraire des informations pertinentes. Gensim offre une implémentation efficace de l'algorithme Word2Vec où vous pouvez facilement entraîner le dmodèle à partir de grands dataset de texte.

3.6.2.1 django :

Django est un framework Python destiné au pour créer des applications web. Il est devenu très populaire et utilisé par des sociétés du monde entier, comme Instagram, Pinterest, et la NASA, c'est notamment grâce à sa philosophie(le modèle MVT), qui a su séduire de nombreux développeurs et chefs de projets. nous avons choisi de réaliser note système(POC) sous forme d'une application web pour les raisons suivantes :

- Pour que l'utilisateur n'aura pas besoin d'installer et configurer le logiciel sur sa machine
- Multi-platform, car le site Web est accessible partir de n'importe quel système d'exploitation.

3.7 Conclusion

Dans ce chapitre, nous avons détaillé l'ensemble du processus effectué dans ce travail, allant de la collecte des données jusqu'à la mise en correspondance des candidats et des missions. Nous avons décrit les étapes de collecte et de prétraitement des données, l'entraînement du modèle de classification ainsi que le modèle Word2Vec. Enfin, nous avons expliqué comment ces deux modèles de réseaux de neurones sont utilisés pour réaliser la mise en correspondance entre les candidats et les missions.

Conclusion générale

L'objectif de notre travail était de concevoir et de mettre en œuvre une architecture de réseau de neurones artificiels permettant de faire correspondre les candidats aux missions dans les boites d'intérim. Nous avons décidé de nous concentrer sur un domaine de travail spécifique, à savoir l'informatique.

Nous avons choisi une approche basée sur le contenu des CV et des offres d'emploi pour concevoir notre système.

Nous avons collecté un ensemble de données pour entraîner nos modèles de réseaux de neurones artificiels. Les données ont été collectées à partir du site web Indeed en utilisant une technique de web scraping.

Pour nettoyer les données textuelles présentes dans les CVs et les offres d'emploi, nous avons appliqué différentes techniques de traitement automatique du langage naturel (TALN).

Nous avons entraîné un réseau de neurones pour la classification de textes afin de mieux structurer les différents paragraphes (segments) des CVs et des offres d'emploi en trois catégories : éducation, compétences et expériences.

Nous avons entraîné un réseau de neurones en utilisant l'algorithme Word2Vec, ce qui nous permet d'associer un vecteur de caractéristiques sémantiques à chaque mot présent dans le jeu de données. Cela nous permet ensuite de déduire le vecteur de caractéristiques du CV et de l'offre d'emploi.

La similarité entre un CV et une offre d'emploi est calculée à l'aide d'une fonction de similarité cosinus qui utilise les vecteurs de caractéristiques correspondants.

Un site web (POC) a été implémenté en utilisant le framework Django python pour mettre en œuvre notre architecture de réseau de neurones.

Nous identifions les perspectives suivantes qui pourraient contribuer à l'amélioration de notre système

— Nous envisageons de généraliser notre modèle pour qu'il puisse fonctionner dans un contexte multilingue et d'établir un benchmark qui nous permettra d'effectuer des tests plus approfondis. À noter que notre travail a été réalisé en utilisant des données en langue anglaise.

— Nous prévoyons d'entraîner les deux modèles avec davantage de données provenant d'autres domaines professionnels tels que la médecine, la mécanique, etc. Cela permettra d'améliorer la généralisation et l'adaptabilité de nos modèles.

Bibliographie

[1] Kenza Harifi. Bien comprendre l'algorithme des k-plus proches voisins (fonctionnement et implémentation sur r et python). https ://medium.com/@kenzaharifi/bien-comprendre-lalgorithme-des-k-plus-proches-voisins-fonctionnement-et-impl

[2] techvidvan. Svm in r for data classification using e1071 package.

[3] Amar Budhiraja. Ml 101 : Linear regression tutorial. https ://medium.com/@amarbudhiraja/ml-101-linear-regression-tutorial-1e40e29f1934.

[4] Ke Gu, Guangtao Zhai, Weisi Lin, Xiaokang Yang, and Wenjun Zhang. No-reference image sharpness assessment in autoregressive parameter space. *IEEE Transactions on Image Processing*, 24(10) :3218–3231, 2015.

[5] bookdown. K-means clustering. https ://bookdown.org/tpinto_home/Unsupervised-learning/k-means-clustering.html.

[6] Artificial Intelligence Wiki. Activation function.

[7] Carine Khalil. *Les méthodes" agiles" de management de projets informatiques : une analyse" par la pratique"*. PhD thesis, Télécom ParisTech, 2011.

[8] digital insiders. [dÉfinition] qu'est-ce que l'apprentissage automatique ? https ://digitalinsiders.feelandclic.com/construire/definition-quest-machine-learning.

[9] expert system. Qu'est-ce que l'apprentissage automatique ? une définition. https ://www.expertsystem.com/machine-learning-definition/.

[10] Wikipedia contributors. Precision and recall — Wikipedia, the free encyclopedia, 2021. [Online ; accessed 31-May-2021].

[11] Metomo JOSEPH BERTRAND RAPHAËL. Machine learning : Introduction à l'apprentissage automatique. https ://www.supinfo.com/articles/single/6041-machine-learning-introduction-apprentissage-automatique, 2017.

[12] Shichao Zhang, Xuelong Li, Ming Zong, Xiaofeng Zhu, and Debo Cheng. Learning k for knn classification. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(3) :1–19, 2017.

[13] William S Noble. What is a support vector machine ? *Nature biotechnology*, 24(12) :1565–1567, 2006.

[14] Kishan Mehrotra, Chilukuri K Mohan, and Sanjay Ranka. *Elements of artificial neural networks*. MIT press, 1997.

[15] wikipedia. word2vec. https ://en.wikipedia.org/wiki/Word2vec.

[16] Dekang Lin et al. An information-theoretic definition of similarity. In *Icml*, volume 98, pages 296–304. Citeseer, 1998.

[17] wikipedia. Curriculum vitæ. https ://fr.wikipedia.org/wiki/Curriculum_vit