RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLEB BLIDA 1

FACULTE DES SCIENCES

DEPARTEMENT D'INFORMATIQUE

MEMOIRE de fin d'étude Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématique et Informatique
Filière : Informatique
Spécialité : Système d'Informatique et Réseaux

Établi par : Cherouat manel
Chanane abir

Sujet :

Classification Des Services Web en utilisant L'apprentissage Automatique

Membres de jury :

Mr. FERFERA.S	Président
Mme. HADJ HANNI	Examinateur
Mme. Zahra Fatma Zohra	Promotrice

Année universitaire : 2021/2022

Résumé

Comme toute technologie, les services Web suscitent beaucoup d'intérêt et promettent beaucoup d'avantages. Ils fournissent des fonctionnalités accessibles via des protocoles réseau standard, et pour organiser un ensemble de services web selon des critères de classe et de similitude fonctionnelle, nécessitant le processus de classification des services Web. Elle peut permettre de faciliter, d'optimiser, d'automatiser l'efficacité et l'efficience des processus de découverte.

Ce travail consiste à classifier automatiquement les services web en plusieurs catégories par un modèle d'apprentissage automatique profond. En effet, Nous avons utilisé Word2Vec qui est un ensemble de modèles d'intégration pour convertir le mot en un vecteur, Nous avons également utilisé les Auto-encodeurs adaptés à la classification.

Le modèle de classification des web services proposé était testé sur le jeu de données disponible et les résultats ont montré son efficacité par rapport certaines solutions proposées dans la littérature.

Mots clés: Service web**,** Classification, Apprentissage profond, Auto-encodeur, Word2Vec, Apprentissage automatique.

Abstract

Like any other technologies, web services attracted a lot of interest and promise many benefits. They provide functionality accessible through standard network protocols, and to organize a set of web services according to class and functional similarity criteria, it requires the web services classification process. It can facilitate, optimize and automate the effectiveness and efficiency of discovery processes.

This work consists in automatically classifying Web services into several categories by a deep machine learning model. Indeed, we used Word2Vec that is a set of integration models to convert the word into a vector; we also used Auto-encoders adapted to the classification.

The proposed web services classification model was tested on the available dataset and the results showed its efficiency compared to some solutions proposed in the literature.

Key words: Web service, Classification, Deep learning, Auto-encoder, Word2Vec, Machine learning.

الملخص

مثل كل تكنولوجيا متقدمة ، تولد خدمات الويب الكثير من الاهتمام وتفتح المجال للكثير من الفوائد. وهي توفر وظائف يمكن الوصول إليها من خلال بروتوكولات الشبكة القياسية, ولتنظم هذه الخدمات وفقًا لمعايير التشابه الطبقي والوظيفي، تتطلب هذه العملية تصنيف خدمات الويب. التي تسمح بتسهيل وتحسين وأتمتة فعالية وكفاءة عمليات الاكتشاف.

نحاول من خلال هذا العمل التركيز على كيفية تصنيف خدمات الويب تلقائيًا إلى عدة فئات من خلال نموذج التعلم الآلي العميق. في الواقع، استخدمنا Word2Vec و هو مجموعة من نماذج التكامل لتحويل الكلمة إلى شعاع ، استخدمنا أيضًا مشفرات تلقائية مناسبة للتصنيف.

نموذج التصنيف المقترح تم اختباره على مجموعة البيانات DataSet المتاحة , و النتائج المتوصل اليها بينت فعاليته مقارنة مع بعض النتائج المقترحة في الدراسات السابقة.

-الكلمات المفتاحية: خدمة الويب، التصنيف، التعلم العميق، التشفير التلقائي، ، التعلم الآلي.

Remerciements

Tout d'abord, nous tenons à remercier **ALLAH le tout puissant,** de nous avoir donné la santé, la volonté et la patience pour mener à terme notre diplôme et pouvoir réaliser ce travail. Au seuil de ce travail, qu'il nous soit permis de témoigner de notre profonde et sincère gratitude envers tous ceux qui ont contribué de loin ou de près à notre formation.

Nous tenons à remercier en premier lieu notre promotrice **Dr. Zahra Fatma Zohra** pour sonattention, son orientation, son aide pendant la réalisation de ce travail et pour être source d'information et de communication sans hésiter à aucun moment de consacrer une part de son temps précieux.

Nous présentons également notre gratitude à tous les professeurs, docteurs chefs de départements, intervenants et assistants de l'université de SAAD DAHLEB DE BLIDA ainsi que ceux de la faculté de Science, et singulièrement les chargés de **la spécialité Système Informatique et Réseau** qui par leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé nos réflexions.

Nous souhaitons adresser nos remerciements les plus sincères à **M. Ramzi**, qui nous a aidé et nous a beaucoup dirigé coté programmation.

Nos sincères remerciements à tous les membres jury qui nous ont fait l'honneur de réviser ce travail.

Dédicace

À l'aide de Dieu tout-puissant, qui m'a tracé le chemin de ma vie, J'ai pu réaliser ce travail, que je dédie :

A ma chère mère Souhila

Aucune dédicace ne saurait exprimer mon respect mon amour éternel et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être, je vous remercie pour tout le soutien et l'amour que vous me portez depuis mon enfance je t'aime plus que tout, que dieu toujours tu gardes à mes coté mon ange gardien.

A l'âme de mon cher père Abdelghani

Qu'Allah t'accorde sa miséricorde et vous réserve une place en son vaste paradis.

J'espère que tu es fier de moi.

A mes chers frères et mes chères sœurs

Abdelraouf, Nabila , Hanaa et Romaissa que dieu les garde toujours pour moi et à mes côtés.

A mon fiancé Mohcen

L'homme de ma vie, mon précieux offre du dieu pour le soutien moral et pour ses encouragements que dieu le garde toujours pour moi et à mes côtés.

A mon neveu et **ma nièce** Adem et Meriem

A ma deuxième maman ma tante Dalila

A mon oncle Mouhamed

A mes chères tantes et mes chères cousins et cousines

A mes chères Grand mères

A mes chères amies

À toutes les personnes qui ont contribué de près ou de loin

A toute la Famille **CHANANE** et **BENABDLAZIZ**

Abir

Dédicace

Je dédie ce travail

A ma chère mère qu'elle a été toujours à mes côtés et aucune dédicace ne peut exprimer mes sincères sentiments

A ma chère sœur Fatima et ses enfants Yasser et Faress

A mes chers frères Khaled et Ali

A toutes mes chères Tantes précisément ma tante Fouzia que dieu lui fasse miséricorde

A mon cher grand père

A l'âme de ma chère grand-mère que dieu lui fasse miséricorde

A mes jolies cousines Ahlem, Nouha, Nada

A mes chers cousins Oussama, Anis, Walid

A tous mes chers oncles

A ma sœur d'enfance Manel

A mon âme sœur Ilhem

A mes amies proches Lynda, Lamiss, Hanene, Amina, Romi.

A toute ma famille

Pour leurs soutiens morale et encouragement.

A moi-même.

A tous ceux qui ont permis la réalisation de ce travail.

Manel

Table des matières

LISTE DES FIGURES

Liste des Tableaux

INTRODUCTION GÉNÉRAL

Motivation et problématique

L'industrie du logiciel a largement adopté les architectures orientées services en utilisant les technologies des services Web. Les services Web ont été largement utilisés dans le commerce électronique, la banque, l'assurance, la bourse et d'autres applications en ligne. Un service Web est un logiciel accessible sur le Web qui peut être publié, localisé et invoqué à l'aide de protocoles Web standard. Il utilise un message SOAP qui contient des services fonctionnels WSDL. Les informations des registres sont utilisées pour publier et découvrir les services dans le modèle SOA pour les utilisateurs de services via les services UDDI.

La détermination automatique de la catégorie d'un service Web, parmi plusieurs catégories prédéfinies, est un problème important. Elle joue un rôle crucial dans la découverte, la sélection, la composition, l'annotation sémantique et la correspondance des services. Les travaux sur la classification visent à proposer des solutions pour l’automatisation de ce processus. La plupart des méthodes de classification proposées sont basées sur la similitude entre les services.

L'apprentissage automatique est un domaine de recherche très important en intelligence artificielle. Et ce dernier est un processus d'imitation de l'intelligence humaine qui repose sur la création et l'application d'algorithmes exécutés dans un environnement informatique dynamique. Ce domaine offre plusieurs méthodes et modèles classification.

Objectif

L'objectif de ce travail est le développement d'un classifieur automatique de descriptions de services Web qui exploite les liens entre la catégorie d'un service Web et les informations couramment trouvées dans la description standard. Ce classificateur est basé sur des algorithmes d'apprentissage automatique, en particulier des modèles d'apprentissage profond.

Organisation du mémoire

La structure de ce document est organisée comme suit :

- **Chapitre 01** Nous commençons par présenter l'intelligence artificielle et l'apprentissage automatique et nous passons à la présentation de l'apprentissage profond avec ses bases, puis nous passons aux différentes techniques d'apprentissage profond utilisées dans la classification de textes.
- **Chapitre 02** Dans le deuxième chapitre nous présentons les services web avec les principes de fonctionnement et leur architecture et techniques, ainsi des travaux de classification.
- **Chapitre 03** Dans le troisième chapitre, nous présentons notre nouveau modèle et nous plongeons en profondeur en expliquant chacune de ses parties que nous avons utilisées pour le construire.
- **Chapitre 04** Dans le dernier chapitre, nous présentons les différents outils utilisés pour construire notre modèle. A la fin, nous partageons les résultats que nous avons obtenus.

Chapitre I : APPRENTISSAGE AUTOMATIQUE ET APPRENTISSAGE PROFOND

Introduction

Depuis 1956, l'intelligence artificielle (IA), en tant que domaine scientifique, représente un ensemble de théories et de techniques. Des programmes informatiques sophistiqués ont été développés capables de simuler certaines caractéristiques de l'intelligence humaine (raisonnement, apprentissage, etc.).

L'apprentissage automatique et l'apprentissage profond font partie à l'intelligence artificielle, dans ce premier chapitre nous allons expliquer ses deux approches importantes en détaillons sur la méthode qui nous avons utilisé comme classifieur.

Commençons par une figure qui présente la relation entre l'intelligence artificielle, l'apprentissage automatique et l'apprentissage profond (figure 1).

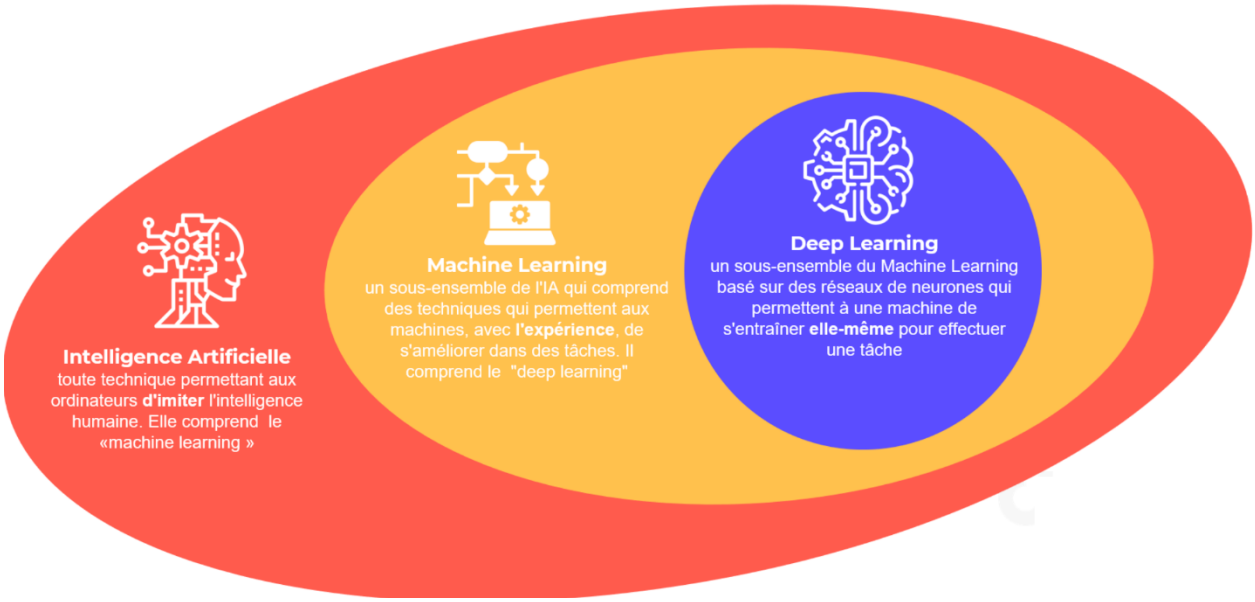


Figure 1 : l'IA, ML, DL en bref [1].

1. Apprentissage automatique

L'apprentissage automatique (machine learning) est un domaine de recherche très important en intelligence artificielle, qui permet aux ordinateurs d'apprendre sans être explicitement

programmés. Cependant, pour apprendre et grandir, les ordinateurs ont besoin de données à analyser et à former.

L'une des techniques d'apprentissage automatique les plus importantes est l'apprentissage en profondeur, qui permet de modéliser les méthodes d'apprentissage automatique de manière très abstraite et d'acquérir des données via une architecture articulée de différentes transformations non linéaires.

1.1 Définition

Depuis leur évolution, les humains ont utilisé de nombreux types d'outils pour accomplir diverses tâches. La créativité du cerveau humain a conduit à l'invention de différentes machines pour faciliter la vie en leur permettant de répondre à divers besoins, Le Machine Learning « ou l'apprentissage automatique » est domaine d'étude un de l'intelligence Artificiel, selon Arthur Samuel, est défini ce domaine comme le domaine d'étude qui donne aux ordinateurs la capacité d'apprendre sans être explicitement programmés. Cette technologie très puissante a permis le développement de tous les systèmes dits « intelligents ». [2]

1.2 Notion de base

La machine learning « l'apprentissage automatique » repose sur les quatre notions fondamentales suivants:

- Dataset : se traduit par jeu ou collection de données, Il s'agit d'un ensemble de données cohérent pouvant se présenter sous différents formats : données chiffrées, textuelles, vidéo, image ou encore son. Le dataset est une brique maîtresse du machine learning. Il va servir à apprendre à un modèle à réaliser une tâche ou faire une prédiction [3].
- Modèle: représente une fonction mathématique créer à partir des données de dataset, on appelle les coefficients de cet fonction *paramètres de modèle* [4].
- Fonction cout: le résultat lorsque on teste le modèle avec le dataset peut donner des erreurs, on appelle l'ensemble de ces derniers la fonction cout [4].

- Algorithme de minimisation (apprentissage): pour le but d'avoir un bon modèle qui donne moins des erreurs on utilise un algorithme de minimisation qui cherche de trouver les paramètres du modèle qui minimisent la fonction cout [4].

1.3 Différent types d'apprentissage

Le domaine de l'apprentissage automatique est souvent divisé en sous-domaines selon les types de problèmes abordés. Alors il existe plusieurs manières de faire apprendre un algorithme a une machine. Ces manières on peut les classer approximativement comme suit: apprentissage supervisé, apprentissage non_supervisé et apprentissage par renforcement (figure2) [4].

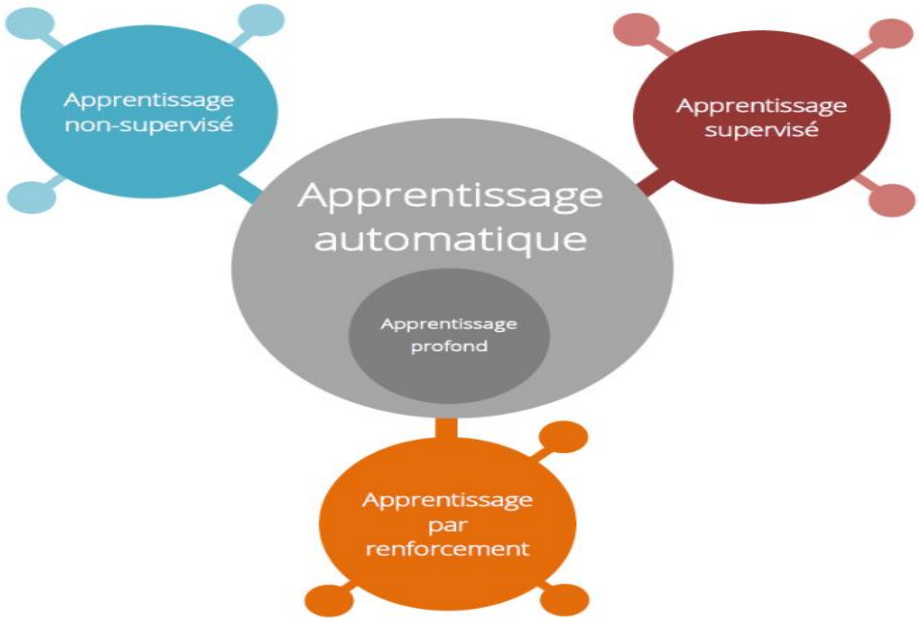


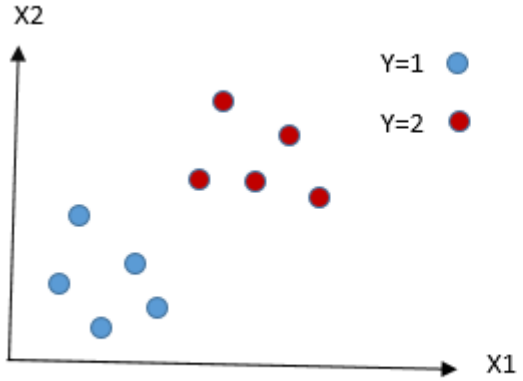
Figure 2: types d'apprentissage automatique [5].

1.3.1 Apprentissage supervisé

Cet apprentissage est le plus rependue pour la machine learning, il consiste à entrainer le programme a des exemples connus pour qu'il puissent les étudier pour créer un modèle (donner X et Y et demander à la machine de trouver F(X)=Y [4].

On parle d'apprentissage supervisé lorsque l'on fournit à une machine beaucoup d'exemples (x, y) dans le but de lui faire apprendre la relation qui relie x à y, l'ensemble de ces exemples sont compilé dans un tableau (dataset).

- La variable **y** (la classe): c'est la valeur qu'on cherche à prédire.
- La variable **x** (le facteur): un facteur (des facteurs) influence la valeur de **y**.



Y	X1	X2
1	$X1_a$	$X2_a$
1	$X1_b$	$X2_b$
2	$X1_c$	$X2_c$
2	$X1_d$	$X2_d$
.....	.....	.....

Figure 3: Dataset de l'apprentissage supervisé [4]

Avec l'apprentissage supervisé on peut développer des méthodes pour résoudre 2 type de problèmes [4] :

- **Problèmes de régression**: dans ce type de problème on cherche à prédire la valeur d'une variable continue (variable qui peut prendre une infinité de valeurs).

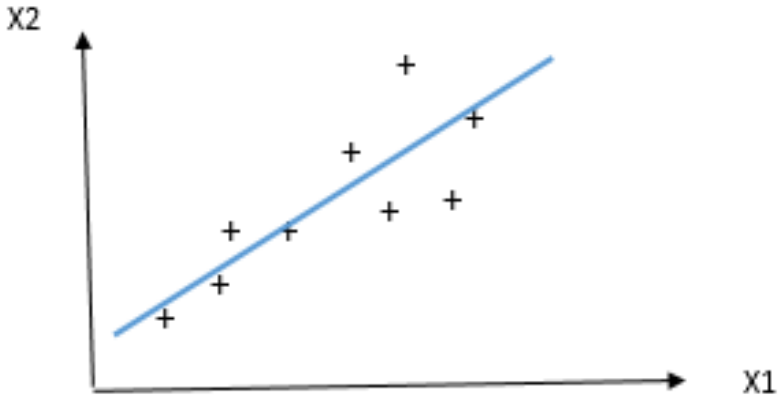


Figure 4: Exemple montre le problème de régression [4]

- **Problèmes de classification**: dans ce type de problème on cherche à classer un objet dans différentes classes, alors qu'on cherche à prédire la valeur d'une variable discrète (variable qui peut prendre un nombre fini de valeurs).

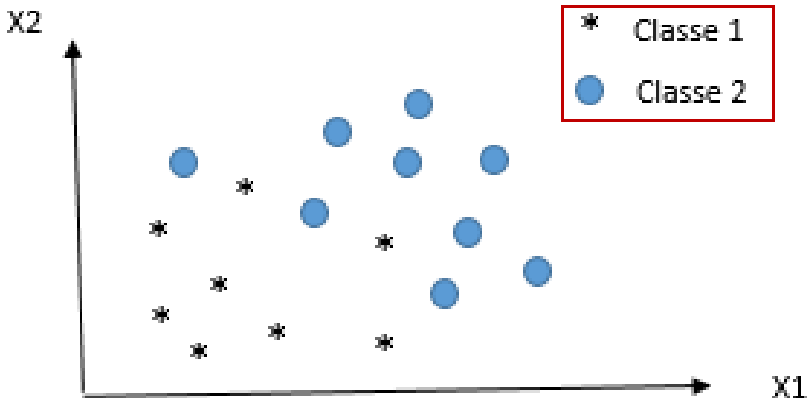


Figure 5 : Exemple montre le problème de classification [4]

1.3.1.1 Principale algorithmes d'apprentissage supervisé

Les algorithmes supervisés sont basés sur des données contenant des paires entrée-sortie. Le couple est connu dans le sens où la sortie est définie a priori. La valeur de sortie peut être une information fournie par un expert. Ces algorithmes tentent de définir une représentation compacte des associations entrée-sortie par le biais de fonctions de prédiction.

- **Algorithmes pour les problèmes de classification**
  - **Régression logistique**: sont très pratiques pour effectuer une classification binaire. En entrée, ils reçoivent des variables prédictives qualitatives et/ou ordinales et mesurent ensuite la probabilité de la valeur de sortie à l'aide de la fonction sigmoïde. [6]
  - **Naive Bayes** : utilise une approche similaire pour prédire la probabilité de différentes classes en fonction de divers attributs. Cet algorithme est principalement utilisé pour la classification de texte et les problèmes multi-classes. [7]

$$\mathbf{P(A \mid B)= \frac{P(B \mid A)P(A)}{P(B)}}$$

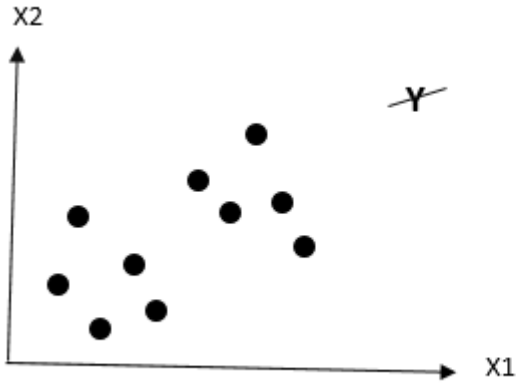
- **Machine à vecteurs de support (SVM) :** sont des algorithmes qui séparent les données en classes, pour que le modèle peut ensuite les appliquer aux nouvelles données. Les SVM sont très utilisées dans la finance [8] .
- **Autres algorithmes :**, Arbre de décision.

- **Algorithme pour les problèmes de régression**
  - **Régression linéaire :** Il s'agit d'effectuer des corrélations simples entre deux variables dans un jeu de données, ce type d'algorithmes est souvent utilisé pour prévoir des ventes ou des risques [9] .

1.3.2 Apprentissage non_supervisé

Cet apprentissage est utilisé lorsque nous n'avons pas des échantillons disponibles, pour cela le programme reçoit les facteurs (X) et il doit découvrir les structures et créer les classes de ce qu'ils appartiennent pour ranger ces facteurs.

Le dataset de cette technique contient seulement les variables X {x1, x2, …. , xn}.



X1	X2
$X1_a$	$X2_a$
$X1_b$	$X2_b$
$X1_c$	$X2_c$
$X1_d$	$X2_d$
.....	.....

Figure 6 : dataset de l'apprentissage non supervisé [4].

Alors pour pouvoir segmenter le dataset, détecter les anomalies ou réduire la dimension des données en compilant les dimensions ensembles, nous avons besoins d'utiliser un algorithme (ou des algorithmes) [4]

1.3.2.1 Principale algorithmes d'apprentissage non_supervisé

Les algorithmes les plus pertinent en entreprise sont [9] :

- **Algorithme apriori** : Cet algorithme cherche les affinités entre deux éléments d'un jeu de données afin d'identifier s'il y a une corrélation négative ou positive entre eux, est très utilisé par les équipes commerciales.
- **K-means :** s'appuie sur une méthode itérative pour trier des points de données en groupes basés sur des caractéristiques similaires, cette technique s'avère également efficace dans le contexte d'analyse de performances informatiques.

1.3.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé est une classe de techniques d'apprentissage automatique qui utilise un ensemble de données étiquetées et non étiquetées. Il se situe ainsi entre l'apprentissage supervisé qui n'utilise que des données étiquetées et l'apprentissage non supervisé qui n'utilise que des données non étiquetées. Il a été démontré que l'utilisation de données non étiquetées, en combinaison avec des données étiquetées, permet d'améliorer significativement la qualité de l'apprentissage [10].

1.3.3.1 Principale algorithmes d'apprentissage semi-supervisé

Les algorithmes les plus pertinent en entreprise sont [9]:

- **Réseaux antagonistes génératifs** : sont des modèles qui imitent la distribution de données, deux réseaux sont placés en compétition afin de déterminer la meilleure solution à un problème.
- **Autres algorithmes** : classificateur bayésien naïf.

1.3.4 Apprentissage par renforcement

Cette dernière méthode d'apprentissage de Machine Learning est de plus en plus utilisée. Elle consiste à laisser l'agent (l'algorithme) apprendre de ses expériences parmi une liste

d'actions grâce à un système de récompense ou de pénalité. L'agent apprend quelle stratégie (ou choix d'actions) maximise le cumul de récompenses (figure7).

Ce type d'apprentissage est souvent utilisé dans le cadre de la robotique, de la théorie des jeux et des véhicules autonomes [11]

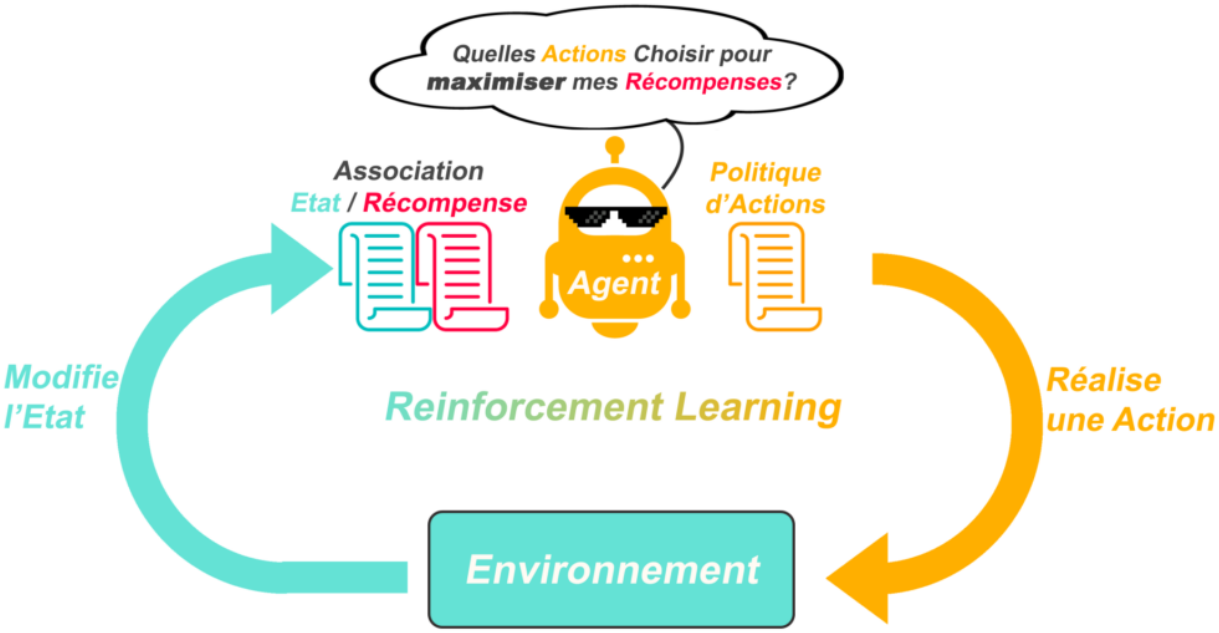


Figure 7: l'apprentissage par renforcement [11]

1.3.4.1 Principale algorithmes d'apprentissage par renforcement

Les algorithmes les plus pertinents en entreprise sont [9] :

- **Q-Learning** : ce type d'algorithmes cherche à trouver la meilleure méthode (une politique optimale) pour atteindre un objectif défini en cherchant à obtenir un maximum de récompenses.
- **Algorithme basé sur un modèle** : à l'inverse du Q-Learning, les algorithmes basés sur un modèle ont une liberté limitée pour créer des états et des actions. Cela leur apporte néanmoins une efficacité statistique supérieure.

2. Apprentissage profond

2.1 Définition

L'apprentissage profond (deep learning) est un ensemble d'algorithmes d'apprentissage automatique qui tentent d'apprendre à plusieurs niveaux, correspondant à différents niveaux d'abstraction. Il utilise généralement des réseaux neuronaux artificiels. Les niveaux de ces modèles statistiques appris correspondent à des niveaux distincts de concepts, où les concepts de niveau supérieur sont définis à partir des concepts de niveau inférieur, et où les mêmes concepts de niveau inférieur peuvent aider à définir de nombreux concepts de niveau supérieur [12].

2.2 Domaine d'application d'apprentissage profond

Le deep Learning se cache aujourd'hui dans beaucoup de nos technologies et permet ces technologies on a [13]:

- Reconnaissance d'image.
- Traduction automatique.
- Voiture autonome.
- Diagnostic médical.
- Recommandations personnalisées.
- modération automatique des réseaux sociaux,
- Prédiction financière et trading automatisé,
- identification de pièces défectueuses,
- détection de malwares ou de fraudes,
- exploration spatiale,
- Robots intelligents.

Avec plusieurs d'autres domaines d'application existe.

2.3 Réseaux de neurones

La technique de réseaux de neurones (profond) représente le modèle utilisé et qu'il faut développer en deep learning, il repose au même principe avec les modèles de machine learning qu'ils sont :

- Données.
- Modèle.
- Algorithme d'optimisation.
- Minimisation des érreurs.

Le modèle dans cette technique représente un réseau de fonctions connecté les uns aux autre (et non pas une simple fonction), et plus qu'il contient des fonctions profondes plus que la machine est capable d'apprendre à réaliser des taches complexes.

2.2.1 Définition

Les réseaux de neurones sont l'unité fonctionnelle de Deep Learning et sont connus pour imiter le comportement du cerveau humain pour résoudre des problèmes complexes axés sur les données, les données d'entrée sont traitées à travers différentes couches de neurones artificiels empilés pour produire la sortie souhaitée [14].

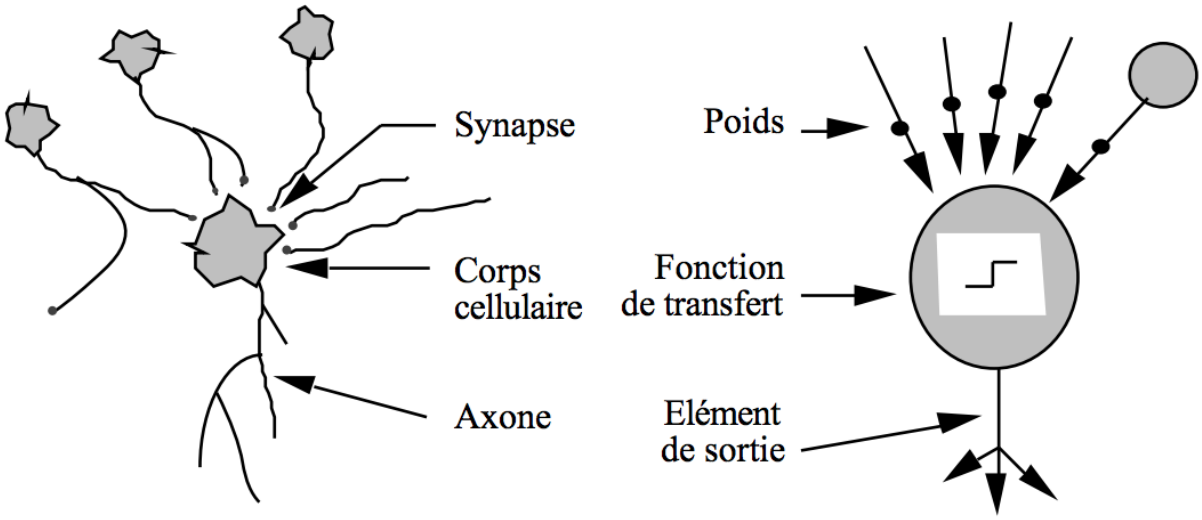


Figure 8:comparaison d'un neurone réel avec un neurone artificiel [15].

2.2.2 Composants clés de l'architecture de réseau neuronal

L'architecture du réseau de neurones est constituée d'unités individuelles appelées neurones qui imitent le comportement biologique du cerveau, voici les différents composants d'un neurone (figure 9) [14]:

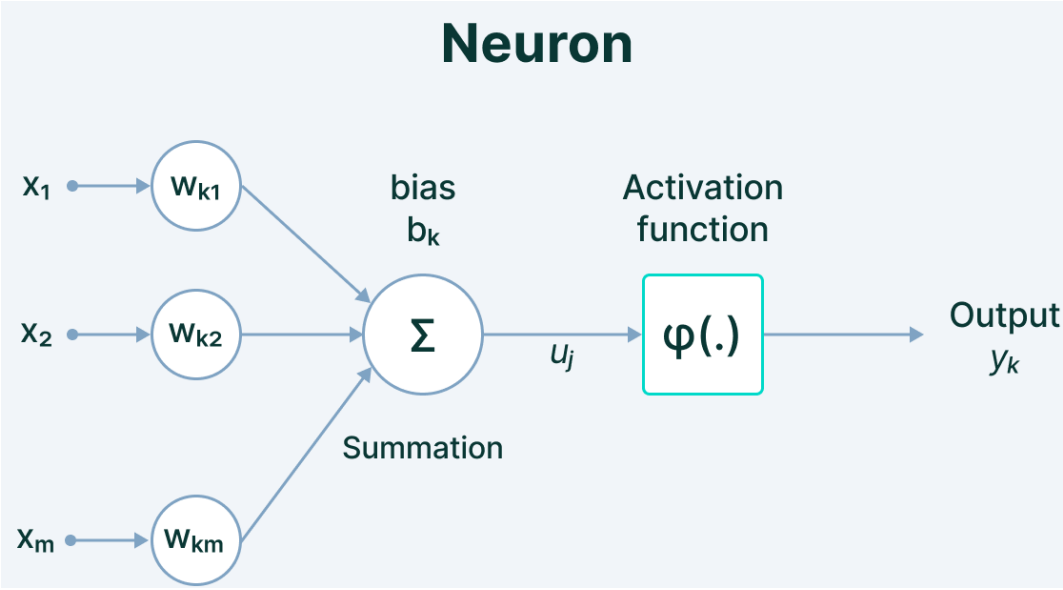


Figure 9: neurone dans un réseau de neurones artificiels [16].

- **Entrée(input) :** C'est l'ensemble des fonctionnalités qui sont introduites dans le modèle pour le processus d'apprentissage.
- **Poids(W) :** Sa fonction principale est de donner de l'importance aux caractéristiques qui contribuent le plus à l'apprentissage.
- **Fonction de transfère :** Le travail de la fonction de transfert consiste à combiner plusieurs entrées en une seule valeur de sortie afin que la fonction d'activation puisse être appliquée. Cela se fait par une simple sommation de toutes les entrées de la fonction de transfert.
- **Fonction d'activation :** Elle introduit la non-linéarité dans le fonctionnement des perceptrons pour envisager une linéarité variable avec les entrées. Sans cela, la sortie ne serait qu'une combinaison linéaire de valeurs d'entrée et ne pourrait pas introduire de non-linéarité dans le réseau.
- **Biais :** Le rôle du biais est de décaler la valeur produite par la fonction d'activation. Son rôle est similaire au rôle d'une constante dans une fonction linéaire.

Lorsque plusieurs neurones sont empilés ensemble dans une rangée, ils constituent une couche, et plusieurs couches empilées les unes à côté des autres sont appelées un réseau neuronal multicouche (figure 10).

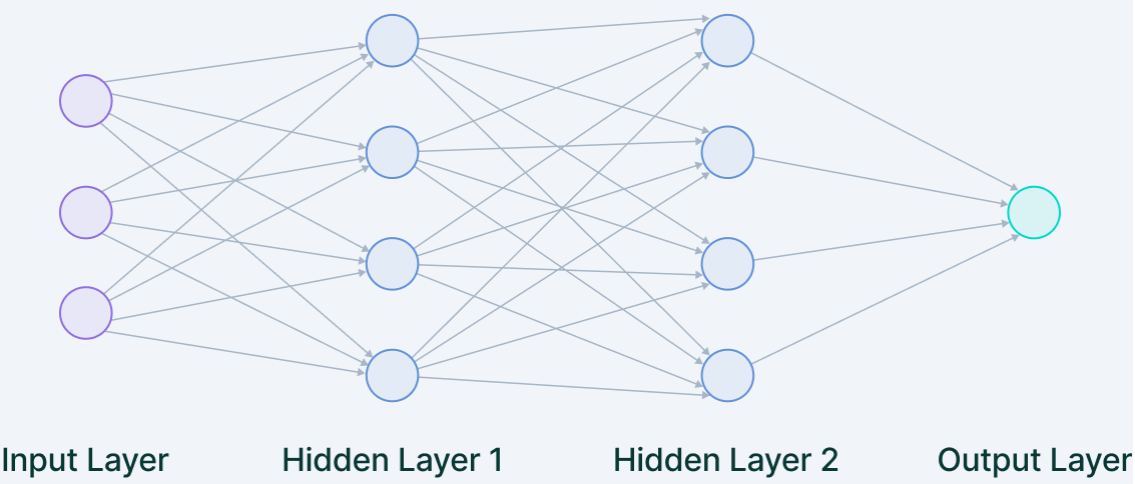


Figure 10: Réseau neuronal multicouche [14]:

- **Couche d'entrée (Input Layer) :** Les données que nous transmettons au modèle sont chargées dans la couche d'entrée à partir de sources externes comme un fichier CSV.
- **Calques masqués (Hidden Layer) :** Les couches cachées sont ce qui fait de l'apprentissage en profondeur ce qu'il est aujourd'hui, ce sont des couches intermédiaires qui effectuent tous les calculs et extraient les caractéristiques des données. Il peut y avoir plusieurs couches cachées interconnectées qui tiennent compte de la recherche de différentes fonctionnalités cachées dans les données.
- **Couche de sortie (output Layer)** : La couche de sortie prend l'entrée des couches cachées précédentes et arrive à une prédiction finale basée sur les apprentissages du modèle. C'est la couche la plus importante où nous obtenons le résultat final.

2.2.3 Types de réseaux de neurones:

2.3.3.1 Réseaux de neurones récurrents RNN

Les réseaux récurrent (ou RNN pour Recurrent Neural Networks) sont des réseaux de neurones dans lesquels l'information peut se propager dans les deux sens, y compris des

couches profondes aux premières couches. En cela, ils sont plus proches du vrai fonctionnement du système nerveux, qui n'est pas à sens unique [17].

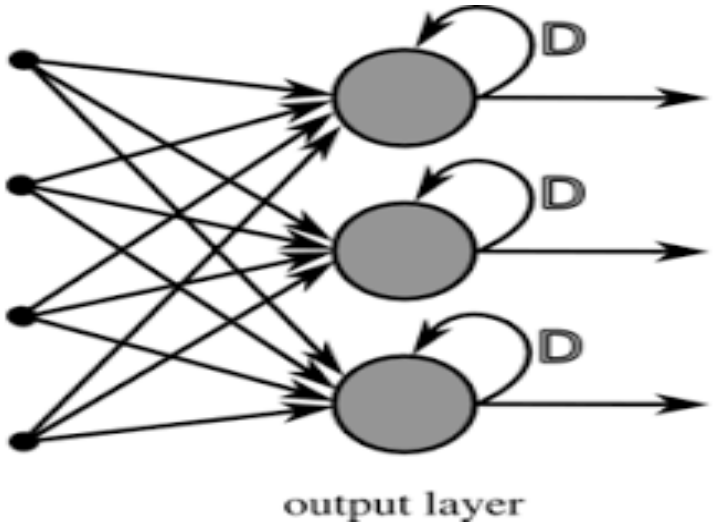


Figure 11: Réseau de neurones récurrent [18].

2.3.3.2 Réseaux de neurones convolutifs CNN

Le réseau de neurones à convolution (également connu sous le nom de ConvNet ou CNN) est un type de réseau de neurones à anticipation utilisé dans des tâches telles que l'analyse d'images, le traitement du langage naturel et d'autres problèmes complexes de classification d'images [14].

Le nom « convolution » est dérivé d'une opération mathématique impliquant la convolution de différentes fonctions. La conception d'un CNN comporte 4 étapes principales [19] :

- **Convolution :** le signal d'entrée est reçu à ce stade
- **Sous-échantillonnage** : les entrées reçues de la couche de convolution sont lissées pour réduire la sensibilité des filtres au bruit ou à toute autre variation.
- **Activation** : cette couche contrôle la façon dont le signal circule d'une couche à l'autre, semblable aux neurones de notre cerveau.
- **Entièrement connecté** : dans cette étape, toutes les couches du réseau sont connectées avec chaque neurone d'une couche précédente aux neurones de la couche suivante Voici un aperçu approfondi de l'architecture CNN et de son fonctionnement, comme l'explique le célèbre chercheur en IA Giancarlo Zaccone. [19]

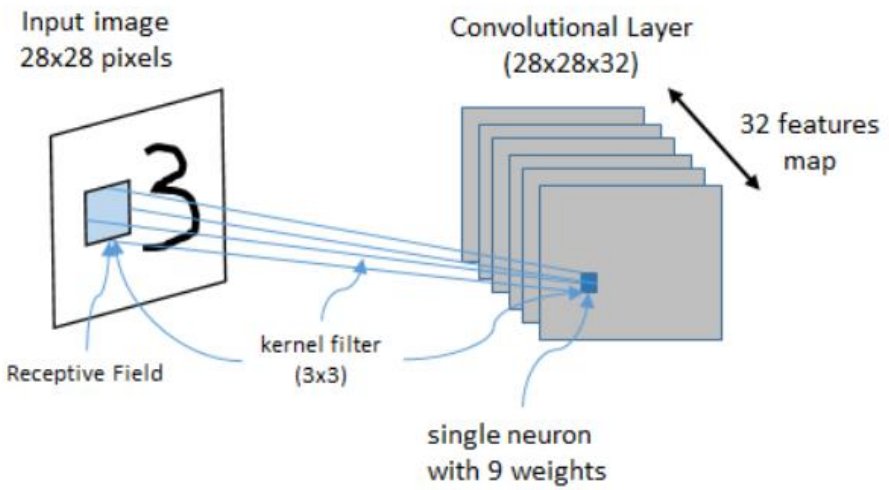


Figure 12: Un exemple de CNN en action [19]

2.3.3.3 Auto-encodeur

Les auto-encodeurs sont une technique d'apprentissage non supervisée dans laquelle nous tirons parti des réseaux de neurones pour la tâche d'apprentissage de la représentation**.** Plus précisément, nous concevrons une architecture de réseau neuronal telle que nous imposons un goulot d'étranglement dans le réseau qui force une représentation compressée des connaissances de l'entrée d'origine [20].

- **L'architecture de l'auto-encodeur** : Les encodeurs automatiques se composent de 3 parties [21] :
  - **Encodeur :** Un module qui comprime les données d'entrée en une représentation codée qui est généralement inférieure de plusieurs ordres de grandeur aux données d'entrée. Est un ensemble de blocs convolutifs suivis de modules de regroupement qui compressent l'entrée du modèle dans une section compacte appelée goulot d'étranglement.
  - **Goulot d'étranglement « bottleneck » :** un module qui contient les représentations de connaissances compressées et qui est donc la partie la plus importante du réseau. Le goulot d'étranglement existe pour restreindre le flux d'informations vers le

décodeur depuis l'encodeur, ne permettant ainsi que le passage des informations les plus vitales.

- **Décodeur :** un module qui aide le réseau à « décompresser » les représentations des connaissances et reconstruit les données à partir de leur forme codée, la sortie est ensuite comparée à une vérité terrain. Le décodeur est un ensemble de blocs de suréchantillonnage et de convolution qui reconstruit la sortie du goulot d'étranglement.

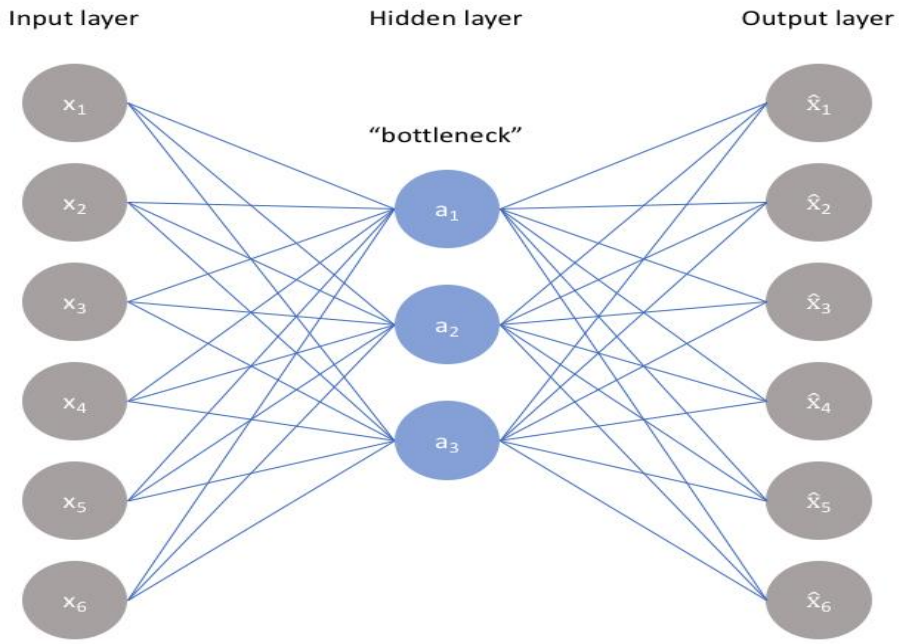


Figure 13: architecture de l'auto-encodeur [21].

- **Bases d'entrainement des auto-encodeurs**: nous devons définir 4 hyperparamètres avant de former un auto-encodeur [21] :
  - **Taille de code :** la taille du code ou la taille du goulot d'étranglement est l'hyperparamètre le plus important utilisé pour régler l'auto-encodeur. La taille du goulot d'étranglement détermine la quantité de données à compresser. Cela peut également servir de terme de régularisation.
  - **Nombre de couches :** comme tous les réseaux de neurones, un hyperparamètre important pour régler les auto-encodeurs est la profondeur de l'encodeur et du décodeur. Alors qu'une profondeur plus élevée augmente la complexité du modèle, une profondeur plus faible est plus rapide à traiter.

- **Nombre de nœuds par couche :** Le nombre de nœuds par couche définit les poids que nous utilisons par couche. En règle générale, le nombre de nœuds diminue avec chaque couche suivante dans l'auto-encodeur à mesure que l'entrée de chacune de ces couches devient plus petite à travers les couches.
- **Perte de reconstruction :** la fonction de perte que nous utilisons pour former l'auto-encodeur dépend fortement du type d'entrée et de sortie auquel nous voulons que l'auto-encodeur s'adapte.

- **Types d'auto-encodeur** : il existe plusieurs types de la méthode auto-encodeur comme : l'auto-encodeur empilé, l'auto-encodeur clairsemé, l'auto-encodeur variationnel
  - **Auto-encodeur empilé :** est un réseau neuronal composé de plusieurs couches d'auto-codeurs épars où la sortie de chaque couche cachée est connectée à l'entrée de la couche cachée suivante.

3. Conclusion

Nous avons présenté au cours de ce chapitre le concept de l'apprentissage automatique ainsi que les différents algorithmes utilisés dans chaque type d'apprentissage. Ensuite, Nous avons introduit le deep learning où on a parlé sur ses domaines d'application tout en insistant sur le fonctionnement des réseaux de neurones et la méthode auto-encodeur utilisés dans notre travail.

Chapitre II : CLASSIFICATION DES SERVICES WEB

Introduction

Les services Web sont une conséquence naturelle de l'évolution du Web. Depuis ses débuts comme moyen de partager et de distribuer des informations à l'échelle mondiale, le Web a progressivement élargi sa portée pour permettre des formes d'interaction plus sophistiquées entre les clients et les serveurs des navigateurs.

La classification automatisée des services web joue un rôle crucial dans la découverte, la sélection et la composition des services, dans ce chapitre nous allons générer tous ce qui est important sur les services web, ensuit nous allons discuter sur les travaux qu'ils sont réalisés pour le même but que notre projet « classification des services web » et faire une comparaison entre aux, s'appuyant sur un certain nombre d'articles.

1. Généralité sur les services web

1.1 Définition

L'affinement de la définition des services Web est fournie par le consortium World Wide Web (W3C), et plus particulièrement le groupe impliqué dans l'activité des services Web: "Une application logicielle identifiée par un URI, dont les interfaces et les liaisons peuvent être définies, décrites et découvertes en tant qu'artefacts XML. Un service Web prend en charge les interactions directes avec d'autres agents logiciels au moyen de messages XML échangés via des protocoles Internet".

La définition du W3C est assez précise et donne également des indications sur la manière dont les services Web devraient fonctionner. La définition souligne que les services Web doivent pouvoir être "définis, décrits et découverts", ce qui clarifie le sens du mot "accessible" et rend plus concrète la notion d'"interfaces orientées Internet et fondées sur des normes". Il indique également que les services Web devraient être des "services" similaires à ceux des intergiciels classiques. Non seulement ils doivent être "opérationnels", mais ils doivent être décrits et annoncés de manière à ce qu'il soit possible d'écrire des clients qui se lient et interagissent avec eux. En d'autres termes, les services Web sont des composants qui peuvent être intégrés dans des applications distribuées plus complexes [22].

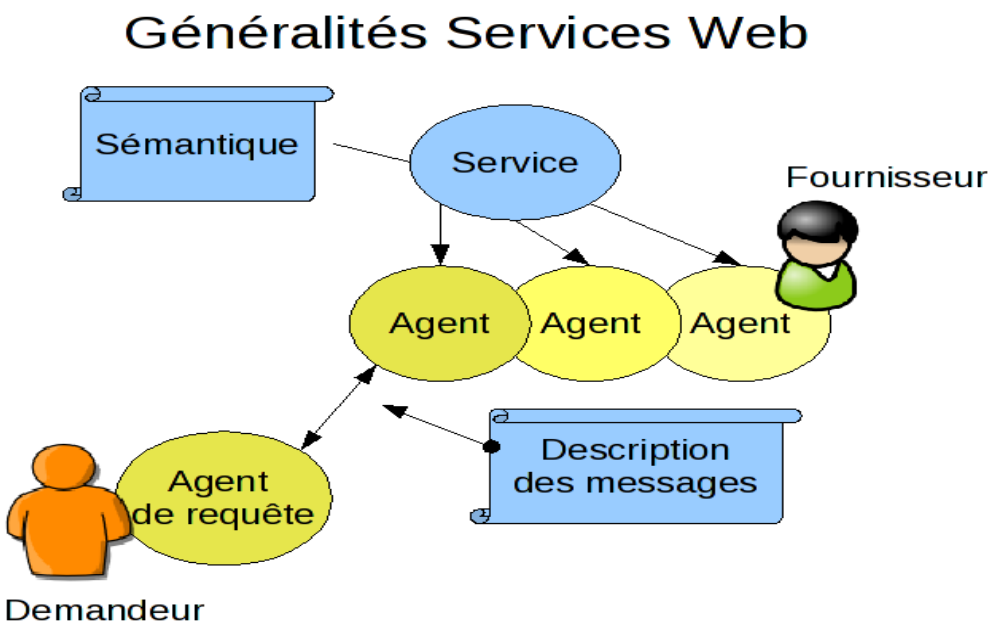


Figure 14: Très grandes généralités sur un service web [22].

1.2 Architecture Orientée Services SOA

1.2.1 Définition

L'architecture orientée services (SOA) est une évolution de la technologie qui devient un facilitateur et, à terme, un transformateur de l'activité.

La définition suivante a été élaborée et raffinée par *James Bean* en travaillant sur diverses initiatives de la SOA au cours des dernières années « Une architecture orientée services (SOA) est une combinaison de consommateurs et de services qui collaborent, est soutenue par un ensemble géré de capacités, est guidée par des principes et est régie par des normes de soutien » [23]

L'objectif essentiel d'une SOA est de permettre une interopérabilité générale entre les technologies existantes et une extensibilité aux objectifs et architectures futurs. La SOA réduit les obstacles à l'interopérabilité en convertissant des systèmes monolithiques et statiques en composants modulaires et flexibles, qu'elle représente sous forme de services pouvant être demandés par le biais d'un protocole standard de l'industrie (figure15) [24].

Figure 15: Architecture Orientée Services SOA [25].

1.2.2 Le concept de SOA

L'architecture orientée services (SOA) repose sur des principes qui permettent d'adopter une approche flexible pour la réalisation de systèmes distribués qui interagissent au-delà des frontières des domaines.

Le concept de l'AOS repose sur la notion de services. Les services regroupent les fonctionnalités d'une application et les rendent disponibles par le biais d'interfaces. Étant donné que la fonctionnalité d'un composant est spécifiée par son interface et que les détails de sa mise en œuvre sont ainsi masqués, on dit qu'il est faiblement couplé à d'autres systèmes qui peuvent accéder à ses services. Cette abstraction sépare la description des services de l'environnement d'exécution.

La description des services peut se faire à différents niveaux d'abstraction et est souvent appelée métadonnées de service. Les métadonnées décrivent les services en termes de qualité de services, d'exigences de sécurité, de fonctionnalités commerciales, etc. d'une manière lisible par une machine (par exemple, XML). Les métadonnées permettent aux services d'être découverts par d'autres systèmes sur un réseau et facilitent un concept connu sous le nom de liaison dynamique, où la mise en œuvre exacte est déterminée au moment de l'exécution (figure16) [26].

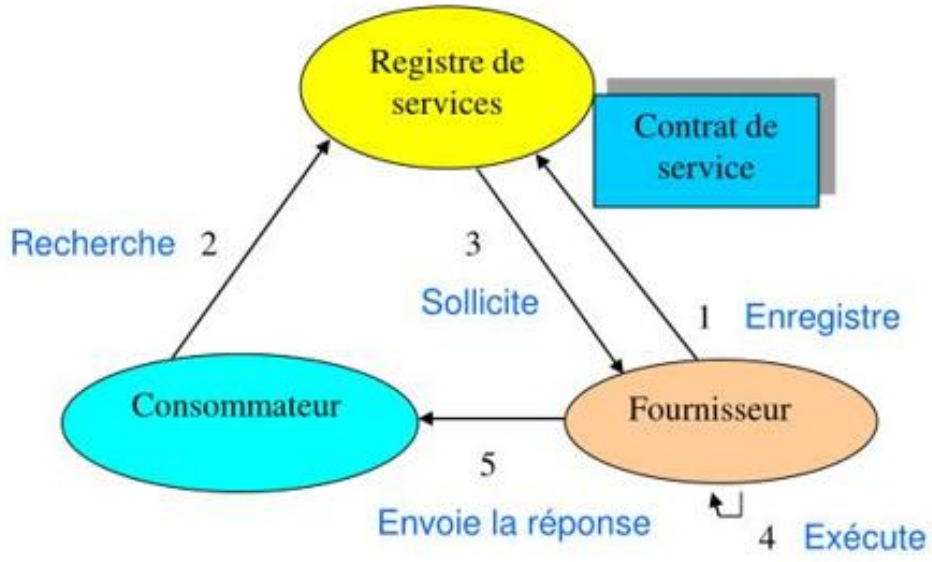


Figure 16: concept SOA : définition primitive [26].

1.3 Principes de fonctionnement des services web

Vous trouverez ci-dessous l'explication de son fonctionnement :

- Le service Web fonctionne comme un modèle de demande-réponse qui permet la communication entre différentes applications en utilisant des jauges ouvertes, par exemple, HTML, XML, WSDL et SOAP.
- Un élément demandera à son partenaire de l'aider à devenir un fournisseur de services spécialisés spécifique. Sur demande, la coopérative spécialisée réagira par un message de réaction. Il y a donc deux messages inclus : un message de demande (XML) et un message de réponse (XML). Nous pouvons fabriquer une administration web en Java par rapport à Solaris qui est ouverte à partir de votre programme Visual Basic que les pics soudains de demande pour Windows.
- Une administration utilise XML pour étiqueter les informations, SOAP pour transférer un message et enfin, WSDL pour décrire l'accessibilité des administrations.

1.4 Architecture des services web

Chaque structure a besoin d'une sorte d'architecture pour s'assurer que l'ensemble de la structure fonctionne comme souhaité, de même pour les services Web, L'architecture des services Web se compose de trois rôles distincts, comme indiqué ci-dessous [27]:

- **Fournisseur (Provider):** Le fournisseur crée le service web et le met à la disposition des applications clientes qui veulent l'utiliser.
- **Demandeur (Requestor):** Un demandeur n'est rien d'autre que l'application cliente qui a besoin de contacter un service web. L'application cliente peut être une application Net, Java ou toute autre application basée sur un langage qui recherche une fonctionnalité quelconque via un service web.
- **Courtier (Broker):** Le courtier n'est rien d'autre que l'application qui fournit un accès à l'UDDI qui permet à l'application cliente de localiser le service web.

Le diagramme ci-dessous, figure (17), montre comment le fournisseur de services, le demandeur de services et le registre de services interagissent les uns avec les autres [27].

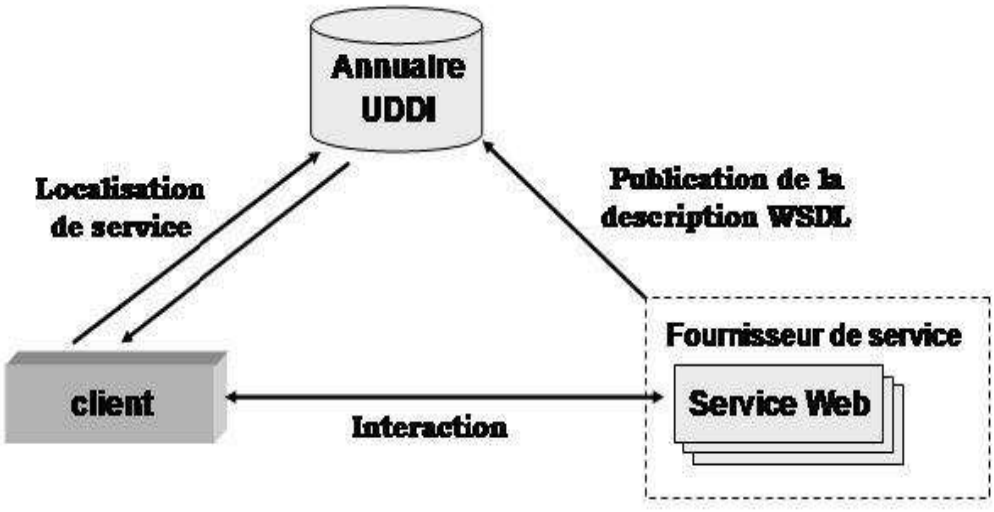


Figure 17: Architecture des services web [28]

Où:

- Publication de la description WSDL: un fournisseur informe le courtier (registre de services) de l'existence du service web en utilisant l'interface de publication du courtier pour rendre le service accessible aux clients.
- Localisation de services : Le demandeur consulte le courtier pour localiser un service web publié.
- Interaction : avec les informations qu'il a obtenues du courtier (registre de services) sur le service web, le demandeur est en mesure de lier, ou d'invoquer, le service web.

1.5 Technologie des services web

1.5.1 Le protocole SOAP (Simple Object Access Protocol)

a. Définition

SOAP est un standard développé par le W3C (World Wide Web Consortium). C'est un protocole de communication basé sur XML qui permet aux applications d'échanger des informations via HTTP. Par conséquent, il permet l'accès aux services Web et l'interopérabilité entre les applications Web.

Son but est de définir la structure générale des messages échangés entre les composants du service Web, et non la structure du contenu. En ce sens, il s'agit d'un protocole sans restriction qui permet aux composants de service Web de définir comment ils formateront le contenu des messages [29]

1.5.2 Les Services Web REST

a. Définition

REST (Representational State Transfer) est un style d'architecture logicielle, pas un standard. Donc il n'existe pas de spécification de REST. Les services Web conformes au style architectural REST, également appelés services Web RESTful, établissent l'interopérabilité entre les ordinateurs sur Internet. Les services Web REST permettent aux systèmes demandeurs de

manipuler les ressources Web via leurs représentations textuelles via un ensemble unifié et prédéfini d'opérations sans état.

REST support plusieurs langages de communication et peuvent renvoyer des messages dans différents formats : HTML, XML, texte brut et JSON. Le format JSON (JavaScript Object Notation) est le plus utilisé pour les messages, car en plus d'être léger, il est lisible par tous les langages de programmation [30].

1.5.3 Différences entre SOA (basé sur SOAP) et REST

L'architecture REST et SOA (basé sur SOAP) sont utilisées pour fournir des services web. Contrairement à ce que l'acronyme SOAP laisse entendre (Simple Object Access Protocol), REST est souvent utilisé lorsque la simplicité de mise en œuvre est recherchée. [31]

De nombreux systèmes d'anciennes générations reposent encore sur le protocole SOAP. REST est arrivé plus tardivement et est souvent considéré comme une solution plus rapide pour des scénarios basés sur le web.

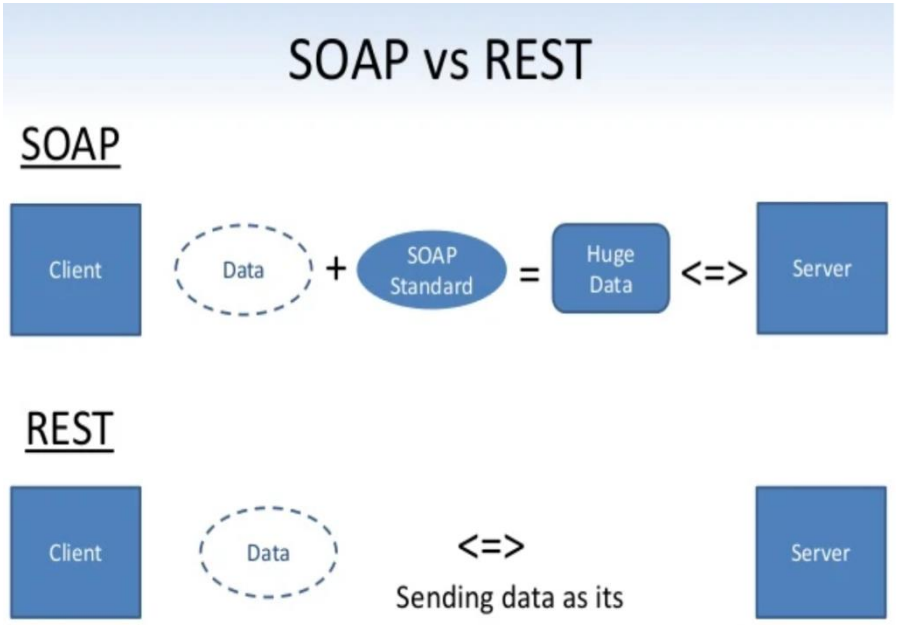


Figure 18: Comparaison entre SOAP et REST [32]

1.5.4 Langage WSDL

a. Définition

WSDL (Web Services Description Language) est un langage de description fondé sur XML, Il permet de décrire de façon précise les services web et indique comment les utiliser en incluant des détails tels que les protocoles, les ports utilisés, les opérations pouvant être effectuées, les formats des messages d'entrée et de sortie et les exceptions pouvant être renvoyées [33].

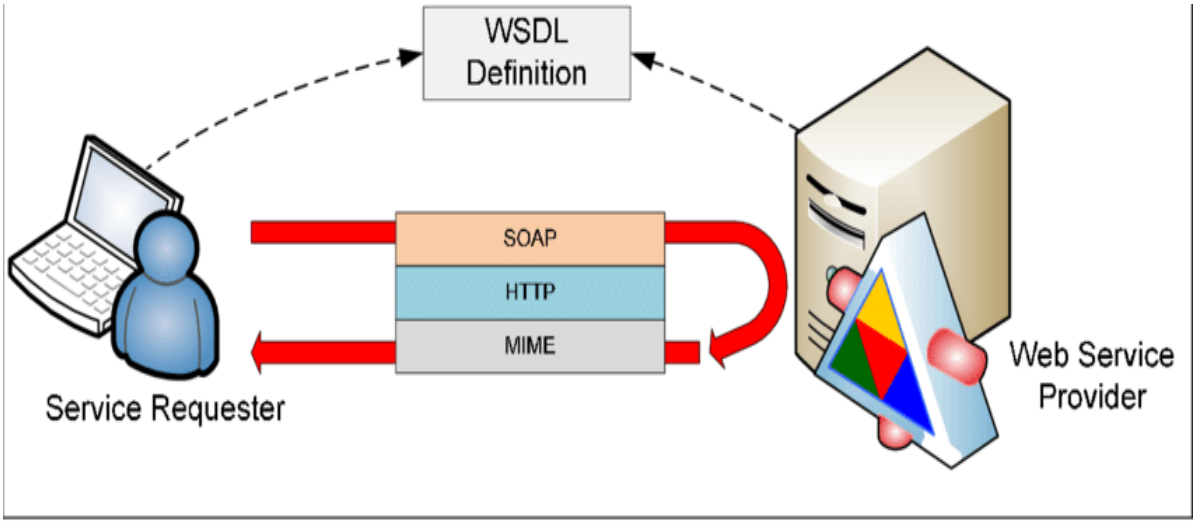


Figure 19: Architecture des services Web [34]

1.5.5 L'annuaire UDDI

a. Définition

UDDI (Universal Description Discovery and Integration) définit une méthode d'identification et de publication d'informations sur les services Web. L'objectif de cette norme est de fournir la fonctionnalité d'un catalogue de services Web afin que les services Web de plusieurs entreprises puissent être découverts et consommés via un mécanisme centralisé qui peut les regrouper [35] UDDI a deux fonctions:

- Il s'agit d'un protocole SOAP qui définit la manière dont les clients communiquent avec le registre UDDI.
- Il s'agit d'un ensemble de registres globalement répliqués.

b. Structure de l'annuaire UDDI

Chaque entrée du répertoire UDDI est constituée de trois parties [35]:

- **Les pages blanches (White Paper) :** ce composant contient la liste des informations nécessaires pour identifier l'organisation (telles que son nom, son adresse physique)
- **Les pages jaunes (Yellow Paper) :** ce composant contient la liste des web services, qui y sont décrit via le standard WSDL. Donc les pages jaunes d'UDDI détaillent la description de l'organisation.
- **Les pages vertes (Green Paper) :** ce composant décrit l'interface vers le service avec suffisamment de détail pour qu'il soit possible d'écrire une application permettant d'utiliser le service Web.

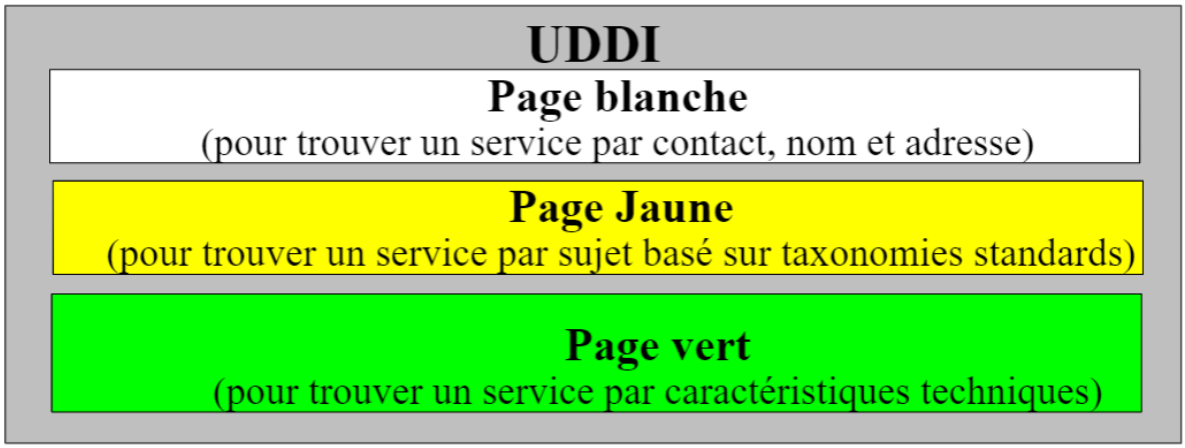


Figure 20: La Structure d'UDDI [35].

2. Classification des web services

2.1 Définition

La classification des services web est l'acte de groupement des services web similaires en différents groupes. La similitude entre ensemble de services web dépend de différents critères, ce qui conduit à différentes méthodes de classification [36]. La Classification améliore de la vitesse du processus de découverte du service Web, et augmente la précision de la découverte du bon service pour un besoin spécifique

2.2 Travaux de classification

La classification des services web se fait en deux types des méthodes :

- Les méthodes d'apprentissage automatique (méthodes classiques).
- Les méthodes d'apprentissage profond.

Le schéma ci-dessous présente les travaux des méthodes dans les deux types de classifications dont nous allons parlerons dans cette section (figure 21).

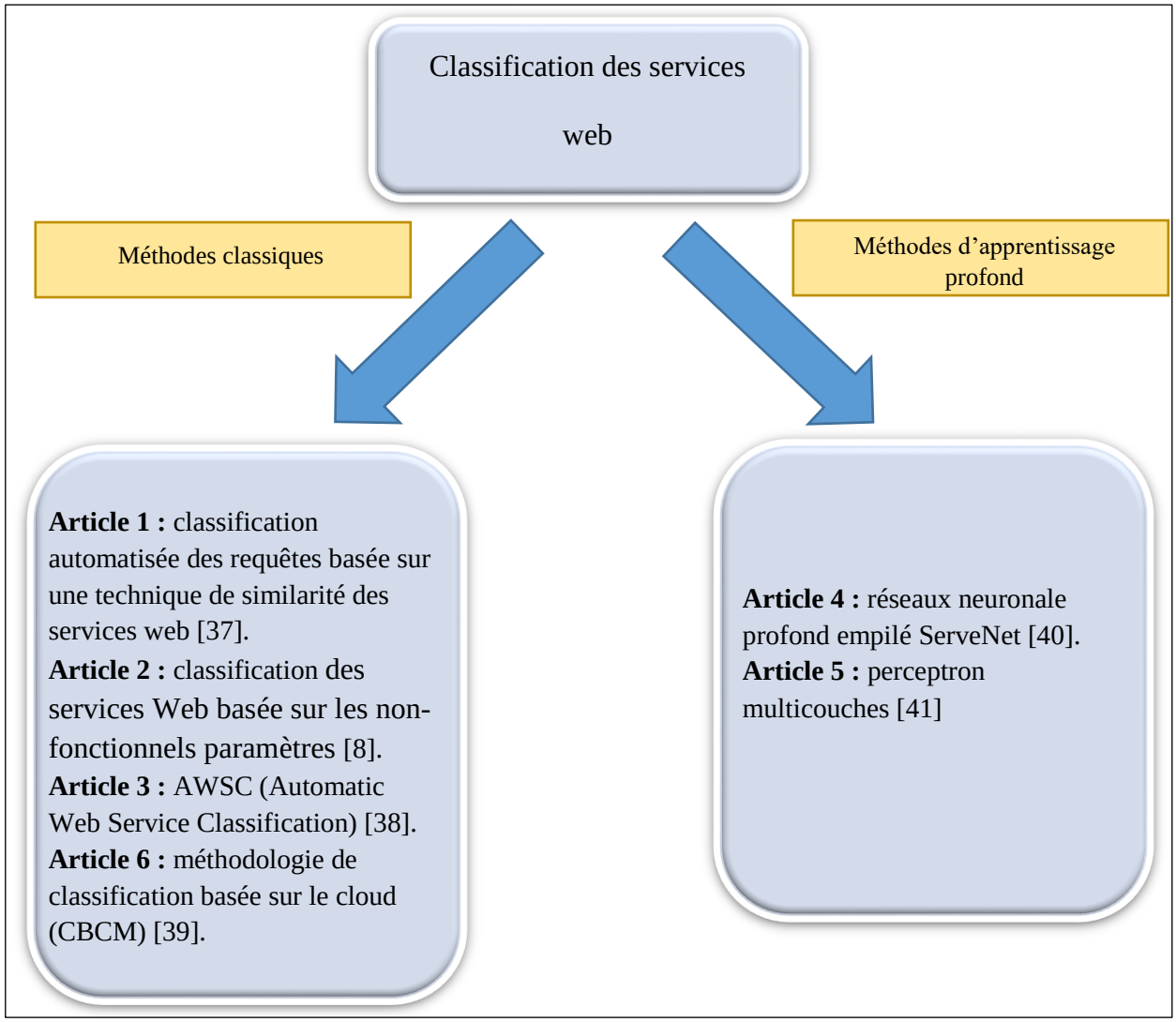


Figure 21: travaux des types de classification.

2.3.1 Les méthodes classiques

2.3.1.1 Classification automatisée des requêtes basée sur une technique de similarité des services Web

B. Saravana Balaji S. Balakrishnan K. Venkatachalam V. Jeyakrishnan, ont utilisé la technologie du web sémantique pour déterminer les similitudes entre les requêtes d'entrée et la base de données stockée.

C'est une nouvelle architecture pour la correspondance de similarité dans les termes donnés pendant la demande de service qui est utilisée pendant le processus de récupération du service. Dans cette recherche, il a été prouvé l'amélioration dans le stockage basé sur l'index, la correspondance de similarité et la comparaison de requêtes pour quelques domaines. Pour la meilleure performance, nous avons implémenté les algorithmes K-Means et KNN pour le processus de recherche de services de qualité.

Cette architecture fonctionne mieux pour la découverte des services [37].

- **Résultats de l'expérience**

Les résultats de l'expérience sont présentés dans le tableau 1.

Tableau 1:Résultat de l'expérience

Précision	Recall
91%	89%

2.3.1.2 Classification des services Web basée sur les non-fonctionnels paramètres (utilisant un classificateur basé sur le vote)

Neerja Negi, Poonam Chaudhary, Satish Chandra Ces ont proposé l'introduction rapide de services Web modernes dans environnement commercial dynamique influence la qualité du service et la satisfaction du client.

Il est donc nécessaire de se concentrer sur l'aspect fonctionnel et non fonctionnel des services Web. Dans cette méthode pour classer avec précision les services web, ils ont utilisé un classificateur basé sur le vote.

Pour la classification de service web sont proposé une méthode qui est composé de quatre classificateurs linéaires et non linéaires : j48, Naive Bayes, SVM et Random Forest. Cette méthode surmonte la limitation de l'individu technique [8].

- **Résultats de l'expérience**

Les résultats de l'expérience sont présentés dans le tableau 2.

.

Tableau 2: Résultat de l'expérience

Critère	SVM	J48	Naive Bayes	Random Forest	Vote Based classifier
Instance correct	86.53%	99.72%	81.31%	99.72%	99.72%

Les résultats montrent que le classificateur basé sur le vote est plus performant que les autres [8].

2.3.1.3 AWSC (Automatic Web Service Classification)

Cette méthode a été présenté par les chercheurs *(Marco Crasso, Alejandro Zunino et Marcelo Campo)* ont proposé que AWSC c'est un classificateur automatique de descriptions de services Web. AWSC exploite les liens entre la catégorie d'un service Web et les informations que l'on trouve couramment dans la description standard.

En outre, AWSC fait le lien entre les différents styles de description des services en combinant des techniques d'exploration de texte et d'apprentissage automatique. Les évaluations expérimentales montrent que cette combinaison aide notre système de classification à améliorer sa précision. Rocchio avec TF-IDF est une technique de classification appropriée pour les services Web et AWSC obtient de meilleurs résultats avec cette technique [38].

- **Résultats de l'expérience**

Les résultats de l'expérience sont présentés dans le tableau 3.

Tableau 3: Résultat de l'expérience

Classificateur	Accuracy
K-NN	39.59%
Naive Bayes	79.38%
Rocchio	85.08%

2.3.1.4 Méthodologie de classification basée sur le cloud (CBCM)

Cette section présente l'approche de CBCM, l'idée principale du travail proposé est de construire un algorithme de classification sémantique appelé « Méthodologie de classification basée sur le cloud » et d'obtenir des résultats plus précis dans la découverte du service web [39].

- **Architecture CBCM**

Elle comprend trois modules [39]:

- Le module de préparation des concepts (CPM) : est le premier module de la méthodologie CBCM. L'objectif de ce module est de convertir la forme originale du service web en une liste de concepts.
- Le module de création d'arbres (TCM) : formalise la liste des concepts sélectionnés dans une structure arborescente.
- Le module CEAM : fait des calculs et renvoie la meilleure correspondance à la demande de l'utilisateur.

- **Résultats de l'expérience**

Ils sont comparés le classificateur CBCM avec les classificateurs les plus récents normes de performance en utilisant le dataset TC V 3.0 [39].
Les résultats de l'expérience sont présentés dans le tableau 4.

Tableau 4: Résultat de l'expérience [39]

Méthode	Precision	Accuracy	Recall	Error
CBCM	97.9	96.4	98.3	3.6
SVM	87.7	85.3	89.5	15.0
SWSC	93.8	90.3	95.7	9.7
ITMT	96.2	94.5	97.9	5.5
ISeM	95.5	92.5	96.5	7.5

Comme montre le tableau ci-dessus, la méthode de CBCM a les meilleures mesures de performances par rapport les autres méthodes.

2.3.2 Méthodes d'apprentissage profond

2.3.2.1 Réseaux neuronale profond empilé ServeNet

La solution proposée par *Yilong Yang, Wei Ke, Weiru Wang et Yongxin Zhao* est le réseau neuronal profond empilé ServeNet basé sur CNN et le modèle de séquence pour la classification des services web sans ingénierie des caractéristiques [40].

- **Définition de modèle**

Le modèle ServNet adopte un CNN 2-D avec LSTM bid pour l'extraction des caractéristiques, ainsi que le CNN 2-D peut aussi extraire les petites régions à l'intérieur des mots, et LSTM bid peut aussi extraire les caractéristiques séquentielles du passé et du futur, pour meilleur performances sur les textes contenant des informations contextuelles [40].

- **Architecture ServeNet**

Le réseau neuronal profond ServeNet composé de trois couches [40]:

- Couche d'incorporation : incorpore la description du service dans des vecteurs numériques.
- Couche d'extraction des caractéristiques : la sortie de cette couche est une représentation de bas niveau de la description de service, pour découvrir automatiquement les caractéristiques locales et globales.
- Couche de taches : contient un réseau neuronal de type feed-forward entièrement connecté.

- **Résultat de l'expérience**

Ils sont comparé les résultats des 10 méthodes d'apprentissage automatique et d'apprentissage profond sur le même jeu de données (Service.csv) notamment Naive-Bayes, Random Forest (RF), SVM (LDA-Linear-SVM et LDA-RBF-SVM), AdaBoost, CNN, Recurrent-CNN, LSTM, BI-LSTM et C-LSTM [40].

Les résultats de l'expérience sont présentés dans le tableau 5.

Tableau 5: Résultat de l'expérience [40].

Méthode	Accuracy%
CNN	27.60
AdaBoost	34.93
LDA-Linear-SVM	33.28
LDA-RBF-SVM	39.79
Naive-Bayes	47.74
LSTM	51.18
RF	54.29
Recurrent-CNN	60.02
C-LSTM	59.24
BI-LSTM	60.45
ServNet	63.31

Enfin, leur proposition ServeNet utilise des CNN 2-D avec BI-LSTM, il peut atteindre la plus haute précision 63,31% parmi tous les repères (Accuracy est une mesure de performance nous allons la définie en 4ème chapitre) y compris toute les méthodes classiques.

2.3.2.2 Perceptron multicouches

Le perceptron multicouche (MLP), également connu sous le nom de réseau neuronal à rétropropagation, est un réseau neuronal artificiel multicouche.
Dans cette étude ils sont formé le perceptrons multicouches (MLP) avec deux méthodes

- Avec l'algorithme de propagation inverse MLP-BPP.
- Avec l'algorithme génétique MLP-GA.

Les deux protocoles ont le même temps d'exécution en théorie [41].

- **Résultats de l'expérience**

Ils sont comparé les résultats des classifieurs de MLP-BPP et MLP-GA avec le résultat de classifieur Naïve Bayes sur le même jeu de données (QWS) [41].

Les résultats de l'expérience sont présentés dans le tableau 6.

Tableau 6:Résultat de l'expérience [41]**.**

/	**Naïve Bayes**	**MLP - BPP**	**MLP-GA**
Accuracy	77.81	96.99	97.81
Precision-Moy	0.7783	0.9701	0.9782
Recall-Moy	0.7795	0.97	0.9781
RMSE	0.2912	0.1328	0.1062

Le tableau 6 montre que la méthode proposée, MLP avec GA, améliore la mesure Accuracy de 22,77% par rapport à Naïve Bayes de la méthode classique.

Et MLP avec GA, a amélioré la précision moyenne de 22,76 % par rapport à Naïve Bayes. De même, MLP BPP a amélioré la précision moyenne de 21,94% par rapport à Naïve Bayes.

Et MLP avec GA, a amélioré le rappel moyen de 22,59% par rapport à Naïve Bayes. Et MLP BPP a amélioré le rappel moyen de 21,77% par rapport à Naïve Bayes (nous allons définie ces mesures de performances dans le 4ème chapitre) [41].

2.3 Discussion

Dans cette section on a présenté les travaux sur la classification des services web dans les méthodes classiques (apprentissage automatique) et dans les méthodes de l'apprentissage profond, les résultats obtenus à partir de ces expériences montre qu'il y a une convergence du taux de réussite entre les deux types de méthodes.

Finalement dans le but de trouver une autre solution pour la classification des services web et pour améliorer le taux de précision par rapport à les travaux ci-dessus on va présenter une nouvelle méthode avec un nouveau modèle, appelé l'auto-encodeur empilé.

3. Conclusion

Nous avons présenté au cours de ce chapitre les services web avec ses principes de fonctionnement et son architecture ainsi que l'architecture orientée services SOA, ensuite on a défini les technologies les plus liées aux services web qui sont : les protocoles SOAP et REST, langage WSDL et l'annuaire UDDI. Et dans la deuxième partie nous avons présenté les déférentes méthodes utilisées dans la classification des web services et pour mieux analyser notre thème on a procédé à une comparaison entre ces méthodes.

Chapitre III : CONCEPTION ET MODELISATION DE LA SOLUTION

Introduction

Dans ce chapitre nous allons parler de notı avec sa conception et modélisation, nous allons entamer le chapitre en présentant notre ıre globale, après, nous allons passer en revue chaque composants de la solution en expliquant leur logique de fonctionnement, commençant par l'extraction des caractéristiques avec ses étapes de prétraitement de texte et la vectorisation en utilisant la méthode Word2Vect en définissent sa structure, passant au partie de classification de texte par la méthode auto-encodeur en expliquant le fonctionnement du modèle neuronal.

1. Architecture globale de classifieur proposé

L'architecture globale de notre solution est divisée en deux partie essentielle :

- Extraction des caractéristiques : qu'elle est présentée en deux étapes :
  - Prétraitement de texte.
  - Vectorisation en utilisons la méthode Word2Vect.
- Classification de texte en utilisons la méthode auto-encodeur.

Nous avons illustré tous les parties et les étapes de cette architecture par la figure présenté ci-dessous (figure 24) :

Où :

- Partie d'extraction des caractéristiques.
- Partie de classification de texte.

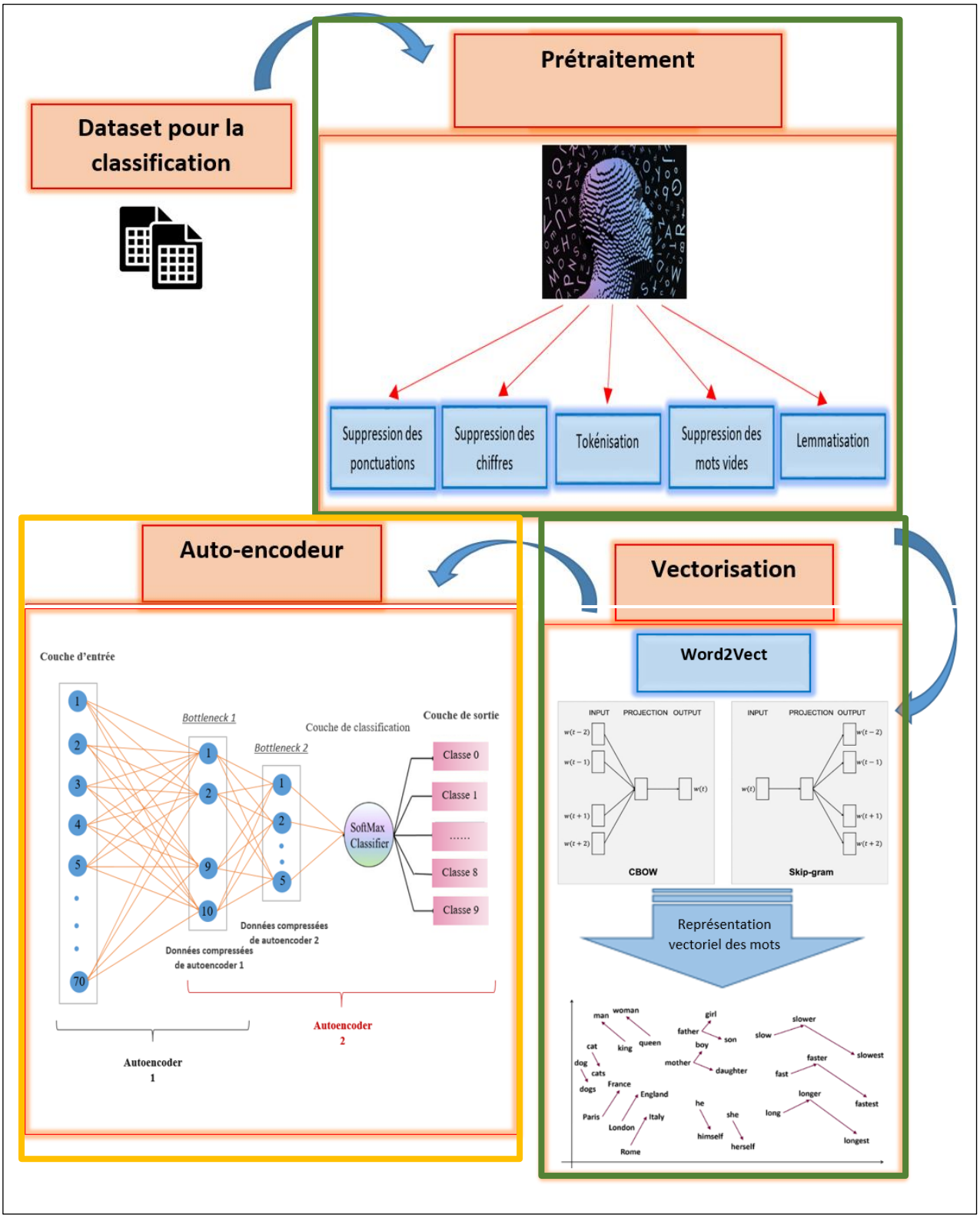


Figure 22: Architecture globale de notre solution.

2. Description de l'architecture

2.1 Dataset pour la classification

Dans cette section, nous présentons le jeu de données de services qui nous avons utilisé pour réaliser notre classifieur.

Il a été crée par *Yilong Yang, Wei Ke, Weiru Wang et Yongxin Zhao* en passant par les étapes suivantes [40] :

- Collecte des services : Les services web sont collectés à partir de la plateforme de partage d'API (ProgrammableWeb).
- Analyse des catégories de services.
- Analyse de la description du service.

Finalement ils construisent le fichier « Service » qui est un fichier de type csv, qui contient deux champs, le premier champ sur le nom de *Service Description* qui présente plusieurs paragraphes descriptent 50 types des services web, le deuxième champ est sur le nom de *Service Classification* qui contient les 50 types (classes) des services web qui sont représentent dans le champ de *Service Description.*

Cet ens1emble de données est contient 10184 lignes et 2 colonnes de taille 4 MO.

	A	B
1	Service Desciption	Service Classification
2	It has been split into multiple APIs, including the Twitter Ads API, Twitter Search Twe	Social
3	The Flickr API can be used to retrieve photos from the Flickr photo sharing service us	Photos
4	What was formerly the ECSeCommerce Servicehas been renamed the Product Adver	eCommerce
5	The Last.fm API gives users the ability to build programs using Last.fm data, whether	Music
6	Bing Maps API and Interactive SDK features an AJAX Map Control. Use BM to build ma	Mapping
7	The Foursquare API provides location based experiences with diverse information al	Social
8	The Google AdSense API is ideal for developers whose users create their own conter	Advertising
9	Geonames is a geographical database with web services that let users extract useful	Reference
10	Bing Web Search API provides an experience similar to Bing.com/search by returning	Search
11	Instagram is a photo sharing iPhone app and service. Users take photos and can share	Photos
12	A State of Trance is a radio show, hosted by Armin van Buuren, which plays trance an	Music
13	Use the Google Analytics Management API to leverage Google's extensive website a	Analytics
14	The Google Earth Engine API allows developers to run algorithms on georeferenced i	Mapping
15	Eventful is the world's largest collection of events, taking place in local markets thro	Events
16	The Commission Detail Service is a REST-based API that enables advertisers and publ	eCommerce
17	Geocoder.us is a public service providing free geocoding of addresses and intersecti	Mapping
18	PayPal offers online payment solutions and has more than 153 million customers wo	Payments
19	The MusicBrainz database contains a huge amount of music metadata, which is maint	Music
20	The service provides online tools for business organizations to manage travel and ex	Enterprise

Figure 23: notre dataset.

2.2 Extraction des caractéristiques

2.2.1 Prétraitement de texte

Le- traitement du langage naturel (NLP) est une branche de la science des données qui traite des données textuelles.

Avant d'utiliser nos données textuelles de notre dataset pour la création de la solution de la classification des Web Services, le traitement de données est important.

Pour préparer les données textuelles pour la construction du modèle, nous effectuons un prétraitement du texte. C'est la toute première étape de projet.

Nous avons utilisé la bibliothèque NLTK, ou Natural Language Toolkit, qui est une suite de bibliothèques logicielles et de programmes. Elle est conçue pour le traitement naturel symbolique et statistique du langage anglais en langage Python. C'est l'une des bibliothèques de traitement naturel du langage(NLP) les plus puissantes.

Les étapes qui sont utilisées dans notre prétraitement de texte sont :

- **Suppression des ponctuations :** Dans cette étape, toutes les ponctuations du texte sont supprimées, telles que '!"#$%&'()*+,-./:; ?@ [\]^_`{|}~'.

  Prenons l'exemple suivant:

  Texte = « the Zazzle create-a-product API can be used to automatically generate over 200 different types of customized products: t-shirts»

  Après la suppression des ponctuations :

  Texte = « the Zazzle create a product API can be used to automatically generate over 200 different types of customized products t shirts»

- **Suppression des chiffres :** Dans cette étape, tous les chiffres du texte sont supprimés.

  Prenons l'exemple suivant:

  Texte = « the Zazzle create a product API can be used to automatically generate over 200 different types of customized products t shirts».

  Après la suppression des chiffres :

  Texte = « the Zazzle create a product API can be used to automatically generate over different types of customized products t shirts».

- **Tokénisation :** la tokénisation dans la bibliothèque NLTK divise les chaînes en listes de sous-chaînes et les transformer en données très simple stockée dans une liste nominative.

  Prenons l'exemple suivant:

  Texte = « the Zazzle create a product API can be used to automatically generate over different types of customized products t shirts».

  Après la tokénisation:

  Texte = ['the', 'Zazzle', 'create', 'a', 'product', 'API', 'can', 'be', 'used', 'to', 'automatically', 'generate', 'over', 'different', 'types', 'of', 'customized', 'products', 't', 'shirts']

- **Suppression des mots vides :** Les mots vides sont les mots couramment utilisés et sont supprimés du texte car ils n'ajoutent aucune valeur à l'analyse, La bibliothèque NLTK

consiste en une liste de mots qui sont considérés comme des mots vides pour la langue anglaise. Certains d'entre eux sont : [je, moi, mon, moi-même, nous, notre, ...].

Prenons l'exemple suivant:

Texte = ['the', 'Zazzle', 'create', 'a', 'product', 'API', 'can', 'be', 'used', 'to', 'automatically', 'generate', 'over', 'different', 'types', 'of', 'customized', 'products', 't', 'shirts']

Après la suppression des mots vides :

Texte = ['Zazzle', 'create', 'product', 'API', 'can', 'used', 'automatically', 'generate', 'over', 'different', 'types', 'customized', 'products', 'shirts']

- **Lemmatisation :** Il enracine le mot mais veille à ce qu'il ne perde pas son sens. La lemmatisation a un dictionnaire prédéfini qui stocke le contexte des mots et vérifie le mot dans le dictionnaire tout en diminuant.

  Prenons l'exemple suivant :

  Texte = ['Zazzle', 'create', 'product', 'API', 'can', 'used', 'automatically', 'generate', 'over', 'different', 'types', 'customized', 'products', 'shirts']

  Après la lemmatisation :

  Texte = ['Zazzle', 'create', 'product', 'API', 'can', 'used', 'automat', 'generat', 'over', 'differe', 'types', 'customi', 'product', 'shirts']

2.2.2 Vectorisation

C'est la deuxième étape importante de la partie extraction de caractéristiques. Son but est de transformer les mots en vecteurs pour terminer l'extraction des caractéristiques et être prêt à les intégrer dans notre modèle d'auto-encodeur.

Dans cette étape, nous avons utilisé l'incorporation de mots (Word Embedding) avec la méthode Word2Vect pour la vectorisation des mots de nos données.

- **Incorporation des mots (Word Embedding) :**

L'incorporation de mots est l'une des représentations les plus populaires du vocabulaire des documents. Elle est capable de capturer le contexte d'un mot dans un document, la similarité sémantique et syntaxique, la relation avec d'autres mots, etc.

Nous avons utilisé la méthode Word2Vect de cette technique qui permet à chaque mot du nos données d'être représenté par un vecteur de nombres réels.

- **Word2Vect :**

Word2Vec est une méthode permettant de construire l'incorporation, elle peut être obtenue à l'aide de deux méthodes, les deux utilisent un réseau de neurones à 3 couches (1 couche d'entrée, 1 couche cachée, 1 couche de sortie): Skip Gram et Common Bag Of Words (CBOW).

a. Modèle CBOW : ce modèle est nourri par le contexte, les représentations distribuées du contexte (ou des mots environnants) sont combinées pour prédire le mot du milieu (prédit le mot cible), le résultat de la couche cachée est la nouvelle représentation du mot.

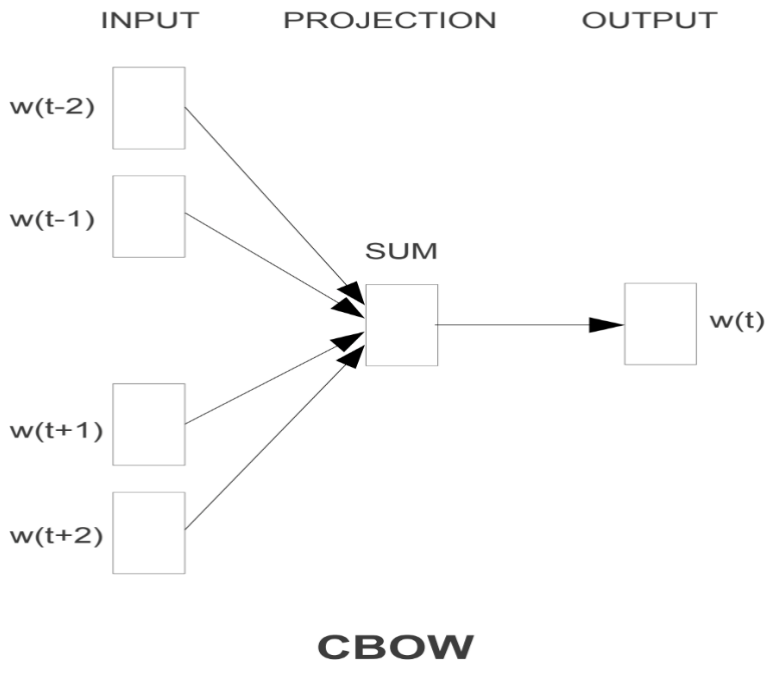


Figure 24: modèle de CBOW.

b. Modèle Skip Gram : ce modèle est nourri par le mot cible, la représentation distribuée du mot d'entrée est utilisée pour prédire le contexte, le résultat de la couche cachée est la nouvelle représentation du mot.

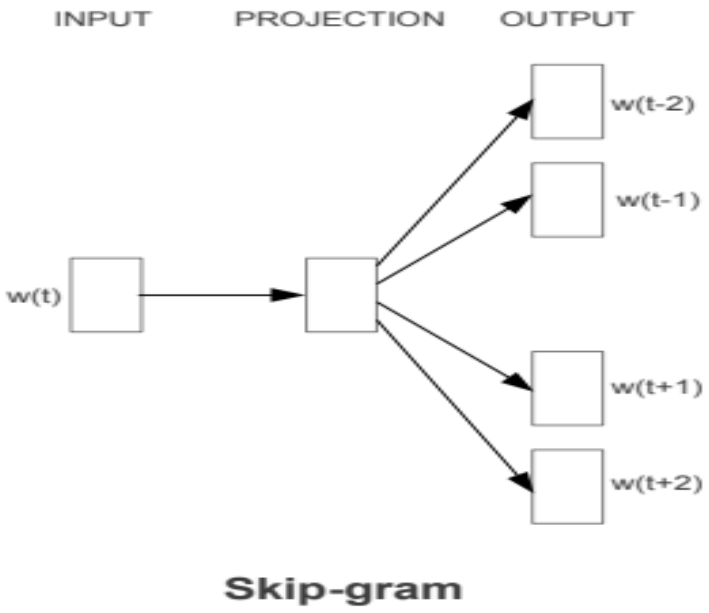


Figure 25: modèle de Skip Gram.

Le résultat qu'on veut chercher à partir de l'utilisation de la méthode Word2Vec est la transformation des mots de notre dataset en vecteur pour qu'on peut les utiliser comme données d'entrer dans la partie suivant de classification (figure 27) :

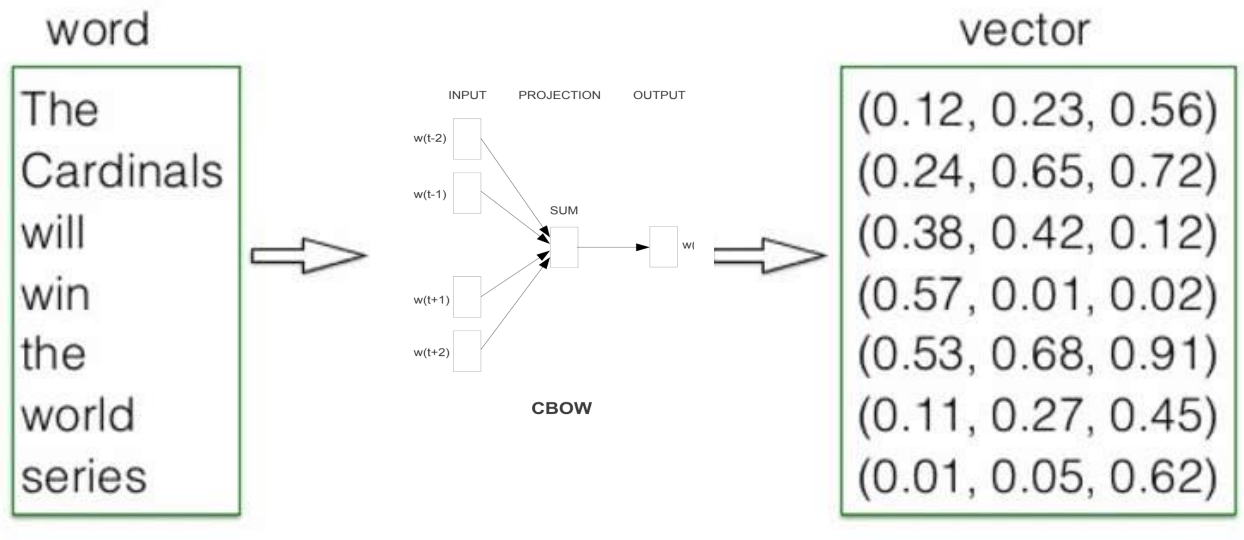


Figure 26: résultat de la méthode Word2Vec [42]

2.3 Classification des textes

Dans cette section nous présentons la méthode d'apprentissage auto-encodeur empilé qui nous avons utilisé pour notre solution de la classification des web services, pour le but d'apprendre une représentation compressée des caractéristiques d'entrée pour un problème de modélisation prédictive de classification de textes.

Nous avons utilisé 10184 descriptions pour les types de classification des services comme échantillons pour l'entrainement, où chaque description à 70 mots qui représente le nombre des caractéristiques d'entrée pour le modèle.

Nous utilisons deux auto-encodeurs, pour apprendre les caractéristiques de notre ensemble de données de manière non supervisée, puis nous utilisons la carte de caractéristiques de l'auto-encodeur pour former un classifier Softmax. Enfin, nous réapprenons le réseau empilé de manière supervisée.

Tableau 7: les caractiristiques des couches d'encodeur et decodeur de 1er auto-encodeur.

Couche	Taille de sortie	Fonction d'activation
Entrée(input)	70	/
Encodeur	30	Relu
Encodeur	10	Relu
Décodeur	30	Relu
Décodeur	70	Sigmoid

Le tableau ci-dessus représente les couches d'encodeur et décodeur qui nous avons utilisé dans notre apprentissage de 1er auto-encodeur et ses paramètres de définition pour chaque couche.

Le partie encodeur apprend à interpréter l'entrée et à la compresser pour construire une représentation compressée qui s'appelle le goulot d'étranglement « bottleneck ».
Le partie décodeur prend la sortie de l'encodeur (la couche de bottleneck) et tente de recréer la même entrée.

Le nombre d'entrée pour la couche d'entrée (input layer) est égale à 70 qui représente le premier nombre qui va se diminuer pour chaque couche suivante jusqu'à que nous arrivons à la dernière couche cachée codé qui représente la couche de goulot d'étranglement qu'elle a un nombre d'entrées égale à 10.

Et pour la première couche de partie décodeur elle reçoit comme entrée le nombre minimisé de résultat de couche goulot d'étranglement 10 et va extrait les caractéristiques pour obtenir les données de sortie, qui sont les mêmes que celles que nous avons entrées dans les couches codées 70.

Pour chaque couche on a utilisé comme type de fonction d'activation la fonction « relu » et pour la dernière couche de sortie on a utilisé la fonction « sigmoid ».

Pour la compilation de notre 1er auto-encodeur nous avons utilisé la fonction « mean_squared_error » cette fonction représente la fonction cout qui calculer les érreurs pour notre modèle, avec l'optimisateur « RMSprop » qui représente l'algorithme de minimisation qu'on a utilisé pour minimiser les erreurs de la fonction cout.

```
Model: "model_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 70)]              0

 dense_4 (Dense)             (None, 30)                2130

 dense_5 (Dense)             (None, 10)                310

 dense_6 (Dense)             (None, 30)                330

 dense_7 (Dense)             (None, 70)                2170

=================================================================
Total params: 4,940
Trainable params: 4,940
Non-trainable params: 0
```

Figure 27: le résumé pour notre entrainement de 1er auto-encodeur.

La figure ci-dessus représente l'affichage de résumé pour notre entrainement de 1er auto-encodeur qui nous aider de résumer toutes les couches utiliser avec ses caractéristiques.

Après que nous avons entrainé le 1er auto-encodeur, nous entrainons le 2ème auto-encodeur de la même manière. La principale différence est que vous utilisez les caractéristiques

compressées à partir du 1er auto-encodeur comme données d'apprentissage dans le second auto-encodeur. Alors, nous réduisons la taille de la représentation masquée à 5 (qu'elle représente le 2ème bottleneck), de sorte que l'encodeur du 2ème auto-encodeur apprenne une représentation encore plus petite des données d'entrée pour que nous pouvons former une couche finale pour classer ces caractéristiques compressées à 5 dimensions en 10 classes de services web (tableau 8).

Tableau 8: les caractéristiques des couches d'encodeur et décodeur de 2eme auto-encodeur.

Couche	Taille de sortie	Fonction d'activation
Entrée(input)	10	/
Encodeur	5	Relu
Décodeur	10	Softmax

Comme une dernière étape nous entrainons la dernière couche Softmax pour classer les vecteurs de caractéristiques à 5 dimensions. Contrairement aux auto-encodeurs, nous entrainons la couche Softmax de manière supervisée à l'aide d'étiquettes des données d'entraînement.

- **La fonction d'activation Softmax :** est une fonction mathématique qui convertit un vecteur de nombres en un vecteur de probabilités, chaque valeur dans la sortie de la fonction Softmax est interprétée comme la probabilité d'appartenance à chaque classe [43].

```
Model: "model_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 70)]              0

 dense (Dense)               (None, 30)                2130

 dense_1 (Dense)             (None, 10)                310

 flatten (Flatten)           (None, 10)                0

 dense_4 (Dense)             (None, 5)                 55

 dense_5 (Dense)             (None, 10)                60

=================================================================
Total params: 2,555
Trainable params: 115
Non-trainable params: 2,440
_________________________________________________________________
```

Figure 28: le résumé pour notre entrainement de 2ème auto-encodeur.

Comme cela a été expliqué, les encodeurs des auto-encodeurs ont été utilisés pour extraire les caractéristiques. Et nous pouvons empiler ces encodeurs avec la couche Softmax pour former un réseau empilé pour la classification.

3. Conclusion

Dans ce chapitre nous avons présenté plus en détail notre architecture globale de la solution et on a descripté cette dernière par l'explication de toutes les parties avec leurs étapes : partie d'extraction des caractéristiques et la partie de classification de texte.

Chapitre IV : TESTES ET RESULTATS

Introduction

Après toutes les notions théoriques et tout ce qui nous avions présentées dans les chapitres précédents, nous passons maintenant à l'étape d'implémentation et réalisation de notre solution. Pour mener à bien notre recherche et construire notre modèle de classification, nous avons utilisé des utiles pour le développement, dans ce chapitre nous avons détailler ces utiles ainsi que la partie d'implémentation.

1. Langage de programmation

Notre solution est programmer avec la langage Python, qui est le langage de programmation open source le plus employé par les informaticiens.

Ce langage s'est propulsé en tête de la gestion d'infrastructure, d'analyse de données ou dans le domaine du développement de logiciels. En effet, parmi ses qualités, Python permet notamment aux développeurs de se concentrer sur ce qu'ils font plutôt que sur la manière dont ils le font. Il a libéré les développeurs des contraintes de formes qui occupaient leur temps avec les langages plus anciens. Ainsi, développer du code avec Python est plus rapide qu'avec d'autres langages [44].

Les principales utilisations de Python par les développeurs sont :

- La programmation d'applications.
- La création de services web
- La génération de code
- La métaprogrammation.

Dans l'univers de la programmation informatique, pouvoir compter sur le soutien d'une communauté de développeurs est essentiel. Or, chaque langage fédère sa propre communauté. Celle du langage Python est particulièrement large, ce qui permet de recevoir de l'aide et de profiter de nombreux outils créés pour simplifier le processus de développement.

2. Environnement de développement

Notre environnement de développement de la solution est la Colaboratory, souvent raccourci en "Colab", est un produit de Google Research.

Colab permet à n'importe qui d'écrire et d'exécuter le code Python de son choix par le biais du navigateur.

C'est un environnement particulièrement adapté au machine learning, à l'analyse de données et à l'éducation.

En termes plus techniques, Colab est un service hébergé de notebooks Jupyter (outil open source permettant d'écrire du code informatique) qui ne nécessite aucune configuration et permet d'accéder sans frais à des ressources informatiques.

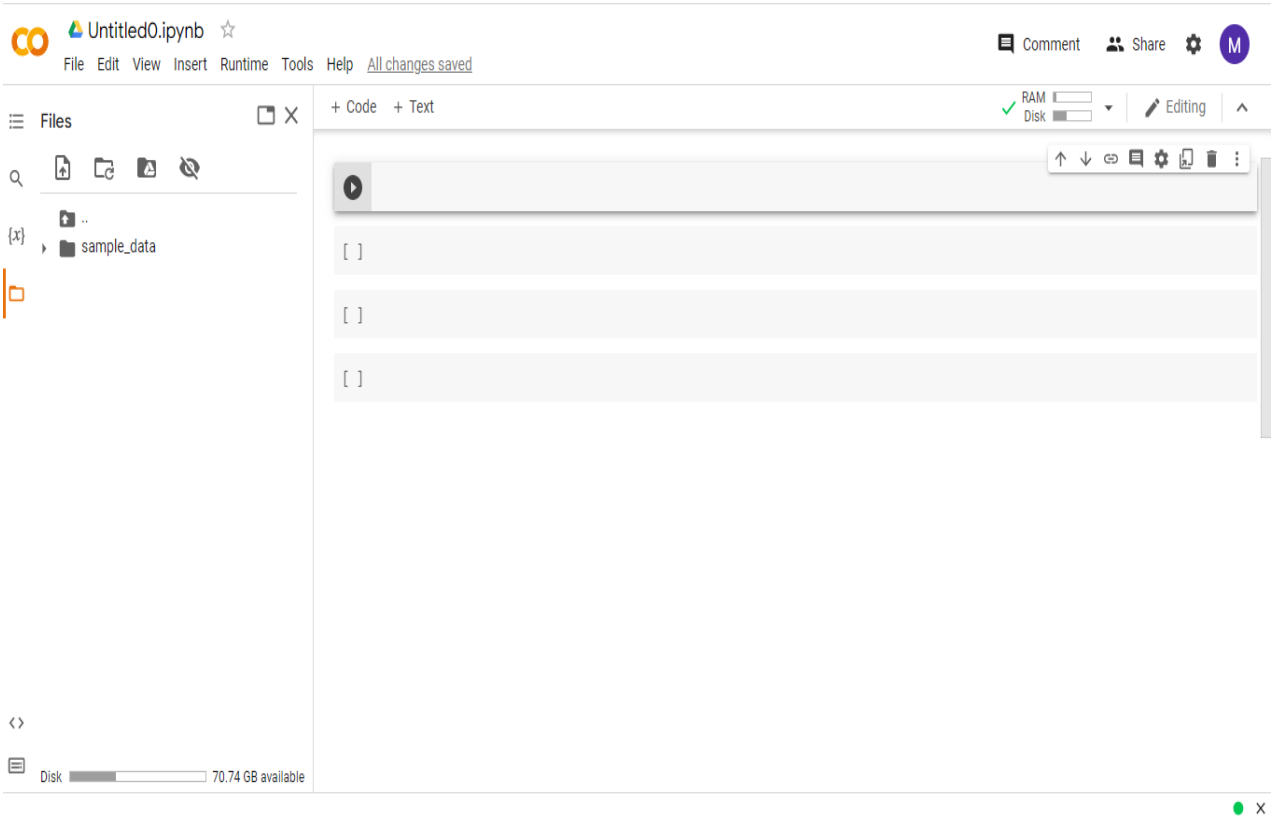


Figure 29: l'environnement Google Colab.

3. Utiles et librairies utilisés

Les bibliothèques que nous avons utilisées pour construire notre modèle d'apprentissage profond sont :

- **NLTK :** Le Natural Language Toolkit (NLTK) est une plateforme utilisée pour construire des programmes Python qui travaillent avec des données de langage humain pour les appliquer au traitement statistique du langage naturel (NLP) [45].

- **TensorFlow :** TensorFlow est un framework open source développé par des chercheurs de Google pour exécuter l'apprentissage automatique , l'apprentissage en profondeur et d'autres charges de travail d'analyse statistique et prédictive [46].

- **Keras :** Keras est une bibliothèque Python pour l'apprentissage profond qui peut s'exécuter au-dessus de TensorFlow et d'autre plateformes. Elle a été développée pour rendre la mise en œuvre de modèles d'apprentissage profond aussi rapide et facile que possible pour la recherche et le développement [47].

- **Gensim :** Gensim est une bibliothèque Python open-source gratuite permettant de représenter des documents sous forme de vecteurs sémantiques [48].

- **Scikit-learn :** est une bibliothèque clé pour le langage de programmation Python qui est généralement utilisé dans les projets d'apprentissage automatique. Scikit-learn se concentre sur les outils d'apprentissage automatique [49].

- **NumPy :** NumPy est le package fondamental pour le calcul scientifique en Python. Il s'agit d'une bibliothèque Python qui fournit un objet tableau multidimensionnel, divers objets dérivés (tels que des tableaux masqués et des matrices) [50].

4. Implémentation

Nous allons commencer le travail principal pour construire notre modèle en utilisant la combinaison des encodeurs de deux auto-encodeurs, après l'extraction des caractéristiques par les deux étapes importantes qui sont le prétraitement et la vectorisation en utilisons le dataset *Service.csv,* ensuit nous allons construire les 10 catégories (classes) des services web par l'extraction des 9 plus ocourants et le reste des catégories nous allons les pris comme 10 ème catégories pour faciliter l'entrainement et pris des meilleures résultats.

Nous allons diviser les données de notre dataset en trois parties : la partie d'entrainement, la partie de validation et la partie de test.

Pour l'entrainement, nous avons pris 80% des données, et pour le test, nous avons pris 20% des données, et enfin pour la validation, nous avons pris 10% des données d'entrainement, alors cela signifie que nous avons pris 72% des données pour l'entrainement et 8% pour la validation.

Enfin, nous avons obtenu :

- 7739 échantillons pour l'entrainement.
- 408 échantillons pour la validation.
- 2037 échantillons pour le test.

5.1 Traitement et validation

Après la création du modèle finale nous l'avons compilé en utilisant la fonction « *categorical_crossentropy »* comme type de la fonction coût pour calculer les erreurs dans ce modèle, avec la fonction « *Adam ()* » comme type de fonction d'optimisation pour le but d'optimiser la fonction coût (erreurs) de modèle.

Ensuit nous avons commencé à entraîner le modèle pendant 200 époques en utilisant la fonction « ***fit()*** » de Keras, cette fonction retourne un objet historique, en stockant sa résultat dans une variable, et nous pouvons l'utiliser plus tard pour tracer les courbes de la fonction de perte

entre l'entraînement et la validation, ce qui nous aidera à analyser visuellement les performances de notre modèle.

Enfin, nous sauvegardons notre modèle à l'aide de la fonction ***save_()*** pour l'utiliser en partie de test.

5.2 Test

Dans cette partie nous avons évaluer notre test par les données qui présente 20% (2037 échantillons) de données initiale avec les 10 classes.

Nous allons pris comme résultat (figure 32) :

```
Test loss: 1.7701585292816162
Test accuracy: 0.645557165145874
```

Figure 30: comparaison entre le nombre de précision et de perte de notre test.

Nous remarquons que le nombre de précision inférieur au nombre de perte, la cause de cette remarque est due au notre dataset qui est de taille très petite.

Nous allons estimer à partir de ce test 1344 classes correct cela signifie que notre modèle a pu estimer 1344 échantillons eux leurs classes exactes.

```
Found 1344 correct labels
```

Figure 31: nombre de classes correctement trouver.

Et nous allons estimer à partir de ce test 693 classes incorrect cela signifie que notre modèle n’a pas pu estimer 693 échantillons eux leurs classes exactes.

```
Found 693 incorrect labels
```

Figure 32: nombre de classes incorrectement trouver.

5.3 Code source

Dans cette partie nous allons présenter notre code source avec une petite explication de chaque étape de code.

La figure ci-dessous montre le code source de l'importation des bibliothèques ainsi les données qui sont téléchargées pour les utilisés dans notre travail (figure 35).

```
[ ]  import nltk
     import string
     from nltk.stem import WordNetLemmatizer
     nltk.download('stopwords')
     nltk.download('punkt')
     nltk.download('words')
     nltk.download('wordnet')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Unzipping corpora/words.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

```
[ ]  import tensorflow as tf
     from tensorflow import keras
     from keras.datasets import mnist
     from keras.layers import Input, Dense , Flatten
     from keras.models import Model
     import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import gzip
     from tensorflow.keras.optimizers import RMSprop
```

```
[ ]  stopwords = nltk.corpus.stopwords.words('english')
     words = set(nltk.corpus.words.words())
     lemmatizer = WordNetLemmatizer()
```

Figure 33: code d'importation des bibliothèques et des données.

La figure ci-dessous montre le code source de la fonction qu'elle on a créé pour le prétraitement de notre données bruts (figure 36).

```
def Preprocess_listofSentence(listofSentence):
 preprocess_list = []
 for sentence in listofSentence :
  sentence_w_punct = "".join([i.lower() for i in sentence if i not in string.punctuation])

  sentence_w_num = ''.join(i for i in sentence_w_punct if not i.isdigit())

  tokenize_sentence = nltk.tokenize.word_tokenize(sentence_w_num)

  words_w_stopwords = [i for i in tokenize_sentence if i not in stopwords]

  words_lemmatize = (lemmatizer.lemmatize(w) for w in words_w_stopwords)

  sentence_clean = ' '.join(w for w in words_lemmatize if w.lower() in words or not w.isalpha())

  preprocess_list.append(sentence_clean)

 return preprocess_list
```

Figure 34: code de fonction de prétraitement.

La figure ci-dessous présente le code source de l'extraction des caractéristiques de notre données en utilisant Word2vec pour l'étape de vectorisation de notre modèle (figure 37).

```
[ ]  tokenize_sentences = []
     for paragraph in preprocess_list:
      for i in range(len(paragraph)):
       tokenize_sentences.append(nltk.tokenize.word_tokenize(paragraph[i]))
```

```
[ ]  from gensim.test.utils import common_texts
     from gensim.models import Word2Vec

     model_W2V = Word2Vec(sentences=tokenize_sentences , size=100,  window=5, min_count=1, workers=4)
     model_W2V.train(tokenize_sentences, total_examples=len(tokenize_sentences), epochs=100)
     model_W2V.save("/content/word2vec.model")
```

```
WARNING:gensim.models.base_any2vec:Effective 'alpha' higher than previous training cycles
```

Figure 35: code de méthode Word2Vec.

La figure ci-dessous présente le code source qui nous avons utilisé pour remplacer les noms des classes par des chiffres (figure 38).

```
[ ]  i = 0
     for service in train_labels :
         if service == 'Financial':
             train_labels[i] = 1
         elif service == 'Messaging':
             train_labels[i] = 2
         elif service == 'eCommerce':
             train_labels[i] = 3
         elif service == 'Payments':
             train_labels[i] = 4
         elif service == 'Social':
             train_labels[i] = 5
         elif service == 'Entreprise':
             train_labels[i] = 6
         elif service == 'Mapping':
             train_labels[i] = 7
         elif service == 'Telephony':
             train_labels[i] = 8
         elif service == 'Science':
             train_labels[i] = 9
         else:
             train_labels[i] = 0
         i+=1
```

Figure 36: code de remplacement les noms des classes par des chiffres.

La figure ci-dessous présente le code source qui nous avons utilisé pour la division de notre jeu de données (figure 39).

```
[ ]  train_data , test_data =train_test_split(tokenize_sentences,
                                              test_size=0.2,
                                              shuffle=False
                                                          )

     #tokenize_sentences=None
```

```
[ ]  train_X,valid_X = train_test_split(train_data,
                                        test_size=0.1,
                                        random_state=10)
```

Figure 37: code de division de données.

La figure ci-dessous présente le code source qui nous avons utilisé pour créer notre 1er auto-encodeur et sa compilation (figure 40).

```
[ ]  input_text= Input(shape=(70,))
     #encoded and decoded layer for the autoencoder
     encoded = Dense(30, activation='relu')(input_text)
     encoded = Dense(10, activation='relu')(encoded)
     decoded = Dense(30, activation='relu')(encoded)
     decoded = Dense(70, activation='sigmoid')(decoded)


     # Building autoencoder
     autoencoder = Model(input_text, decoded)
     encoder = Model(input_text, encoded)

     # compiling the autoencoder
     autoencoder.compile(optimizer= RMSprop(), loss='mean_squared_error')
```

Figure 38: code de 1er auto-encodeur et sa compilation.

La figure ci-dessous présente le code source de la création 2eme auto-encodeur et la création de modèle finale (figure 41).

```
[ ]  def fc(enco):
         flat = Flatten()(enco)
         den = Dense(5, activation='relu')(flat)
         out = Dense(num_classes, activation='softmax')(den)
         return out
```

```
[ ]  full_model = Model(input_text,fc(encoded))
```

Figure 39; code de 2eme auto-encodeur et le modèle finale.

La figure ci-dessous présente le code source de la compilation de modèle finale (figure 42).

```
[ ] full_model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adam(),metrics=['accuracy'])
```

Figure 40: code de compilation de modèle finale.

La figure ci-dessous présente le code source pour sauvegarder le modèle final et de processus de la précision et la perte pour tracé les courbes (figure 43).

```
[ ]  full_model.save_weights('/content/services_model_final.h5')
```

```
[ ]  accuracy = classify_train.history['accuracy']
     val_accuracy = classify_train.history['val_accuracy']
     loss = classify_train.history['loss']
     val_loss = classify_train.history['val_loss']
     epochs = range(len(accuracy))
     plt.plot(epochs, accuracy, 'bo', label='Training accuracy')
     plt.plot(epochs, val_accuracy, 'b', label='Validation accuracy')
     plt.title('Training and validation accuracy')
     plt.legend()
     plt.figure()
     plt.plot(epochs, loss, 'bo', label='Training loss')
     plt.plot(epochs, val_loss, 'b', label='Validation loss')
     plt.title('Training and validation loss')
     plt.legend()
     plt.show()
```

Figure 41: code pour sauvegarder de modèle finale et le processus de précision et perte.

5.4 Mesures d'évaluation de performances

Après avoir entrainé et sauvegardé notre nouveau modèle, dans cette section, nous présentons les mesures d'évaluation de performances que nous avons utilisées pour évaluer notre modèle pour la classification des services web.

Nous évaluons les résultats des prédictions en utilisant des mesures d'évaluation comme le f1-score et la précision, et nous calculons la perte des prédictions en utilisant la fonction de perte.

- **Precision :** Elle permet de connaître le nombre de prédictions positifs bien effectuées, en d'autres termes c'est le nombre de positifs bien prédit (Vrai Positif) divisé par l'ensemble des positifs prédit (Vrai Positif + Faux Positif) [51].

$$\text{Precision} = \frac{\text{Vrai Positif}}{\text{Vrai Positif} + \text{Faux Positif}}$$

- **Recall :** permet de savoir le pourcentage de positifs bien prédit par notre modèle, en d'autres termes c'est le nombre de positifs bien prédit (Vrai Positif) divisé par l'ensemble des positifs (Vrai Positif + Faux Négatif) [51].

$$\text{Recall} = \frac{\text{Vrai Positif}}{\text{Vrai Positif} + \text{Faux Négatif}}$$

- **F1-score :** est une métrique permettant de combiner la Precision et le Recall existe, pour effectuer une bonne évaluation de la performance de notre modèle [51].

$$\text{F1-score} = 2 * \frac{\text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$$

- **Accuracy :** est l'indicateur le plus simple, il indique le pourcentage de bonnes prédictions, c'est un très bon indicateur parce qu'il est très simple à comprendre [52].

$$\text{Accuracy} = \frac{\text{Vrai Positif} + \text{Vrai Négatif}}{\text{Totale}}$$

- **Loss :** La perte est une valeur qui représente la somme des erreurs dans notre modèle. Il mesure à quel point (ou mal) notre modèle se porte bien. Si les erreurs sont élevées, la perte sera élevée [53].

5.5 Résultat

Enfin, il s'agit de la dernière section de notre recherche, nous allons présenter notre résulta finale.

En utilisant la méthode d'apprentissage profond auto-encodeur nous avons obtenu de bon résultat par rapport à l'utilisation d'un dataset de petite taille (4MO).

Nous avons obtenu un score de 0.86 pour la métrique de F1-score, et un score de 0.67 pour la métrique Accuracy, le tableau ci-dessous présente les résultats d'évaluation de Accuracy et F1-score que nous avons obtenus (tableau 9).

Tableau 9 : résultats des métriques Accuracy et F1-score.

Accuracy	F1-score
0.67	0.86

De plus, pour la métrique de Précision, nous avons obtenu un score de 0.68 pour la classe 0 et un score de 0.05 pour la classe 8, et pour les autres classes nous avons obtenu un score de 0.00.

Et pour la métrique de Recall, nous avons obtenu un score de 0.97 pour la classe 0 et un score de 0.06 pour la classe 8, et pour les autres classes nous avons obtenu un score de 0.00.

Ces résultats sont dus au cause que la plupart des descriptions de notre dataset sont appartient à la classe 0, et la petite taille de ce dataset explique le reste des résultats pour les autres classes.

Le tableau ci-dessous présente les résultats d'évaluation de Précision et Recall que nous avons obtenus (tableau 10).

Tableau 10: résultats des métriques Precision et Recall.

Classes	Precision	Recall
Classe 0	0.68	0.97
Classe 1	0.00	0.00
Classe 2	0.00	0.00
Classe 3	0.00	0.00
Classe 4	0.00	0.00
Classe 5	0.00	0.00
Classe 6	0.00	0.00
Classe 7	0.00	0.00
Classe 8	0.05	0.06
Classe 9	0.00	0.00

La figure ci-dessous présente les résultats de l'évaluation de notre modèle en utilisant la métrique de précision (figure 44).

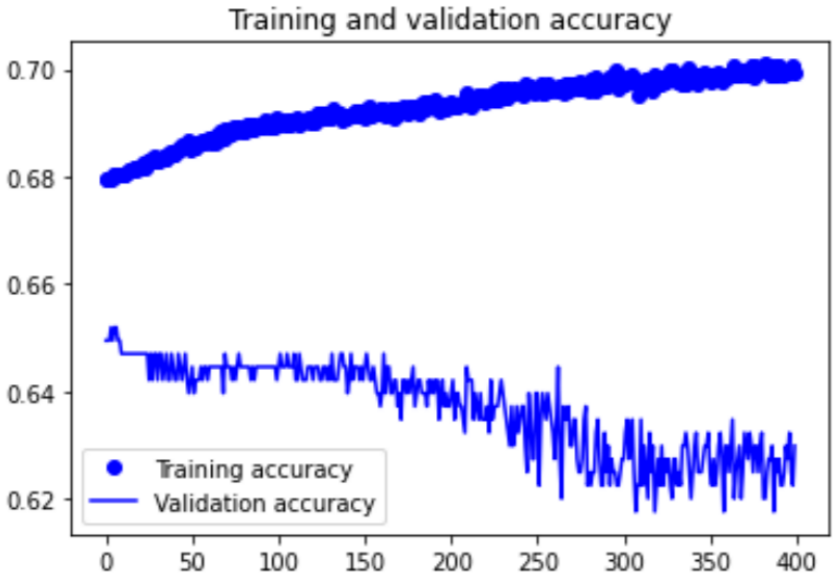


Figure 42: Accuracy de l'entrainement et la validation.

La figure ci-dessous présente les résultats de mauvaise prédiction qu'elle a été calculer en utilisant la fonction de Loss que nous avons obtenue pendant l'évaluation de notre modèle.

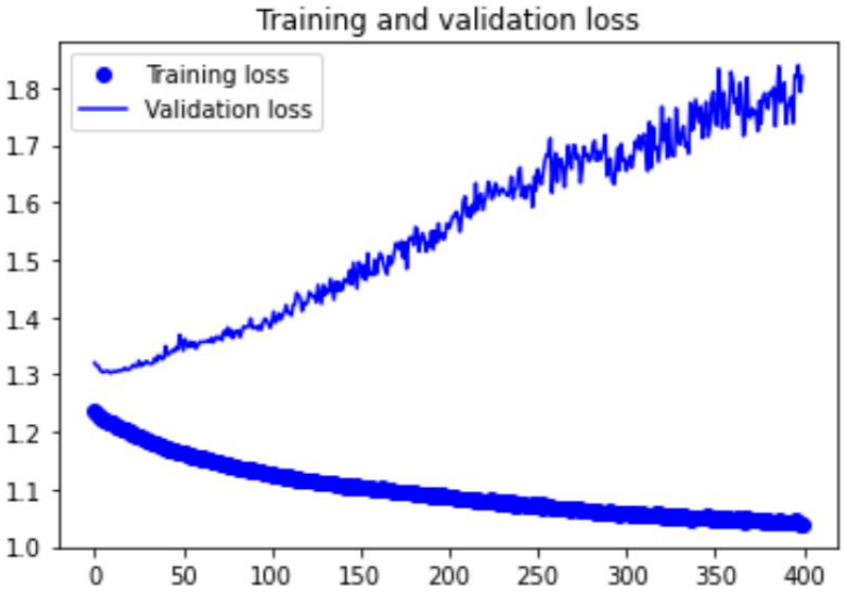


Figure 43: Loss de l'entrainement et la validation.

5.5.1 Comparaison des résultats

Finalement, dans cette section nous comparons 8 méthodes d'apprentissage automatique et d'apprentissage profond avec notre méthode pour la classification de services Web sur le même ensemble de données de services proposé, notamment Naive-Bayes, SVM, AdaBoost, Recurrent-CNN, LSTM, BI-LSTM, C-LSTM, et ServNet.

Les résultats de l'expérience sont présentés dans le tableau 11.

Tableau 11: : mesure Accuracy de tous les travaux qui ont utilisé le même dataset [40].

Méthode	Accuracy%
AdaBoost	34.93
SVM	39.79
Naive-Bayes	47.74
LSTM	51.18

Recurrent-CNN	60.02
C-LSTM	59.24
BI-LSTM	60.45
ServNet	63.31
Notre méthode	67

Pour une meilleure lisibilité et visualisation des résultats nous présentons l'histogramme suivant (figure 44) :

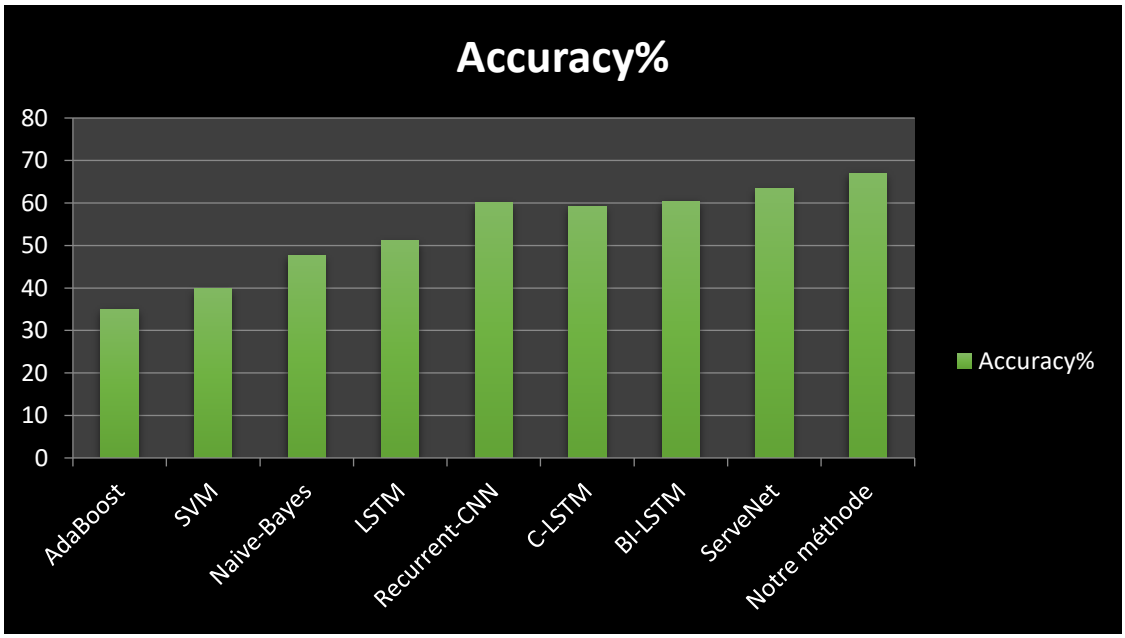


Figure 44: Comparaison entre notre méthode et les autre méthode utilisé

A travers les tableaux et l'histogramme présentés précédemment, nous pouvons clairement constater que de bons résultats ont été obtenus par notre modèle.

Lorsqu'on on compare les résultats des méthodes d'apprentissage classique : AdaBoost, SVM et Naive-Bayes, avec les résultats des méthodes d'apprentissage profond : LSTM, Recurrent-

CNN, C-LSTM, BI-LSTM, ServNet et Notre méthode, on remarque que les méthodes d'apprentissage profond peuvent atteindre la plus haute précision par rapport aux méthodes d'apprentissage classique.

5. Conclusion

Finalement, ce dernier chapitre a été la partie la plus intéressante de notre étude, car nous avons décrit les outils et l'environnement utilisés et nous avons présenté notre code pour le développement, ainsi que les résultats obtenus par notre nouveau modèle, et enfin nous avons comparé notre résultat eux résultat des travaux qui ont été utilisé le même dataset.

Nous passons donc maintenant à la conclusion générale.

CONCLUSION GÉNÉRALE

La classification des services web est le processus qui permet d'organiser un ensemble de services web selon des critères de similitude catégorielle et fonctionnelle. Elle peut permettre aussi de faciliter, d'optimiser, d'automatiser l'efficacité et l'efficience des processus de découverte, de composition, d'exécution et de la gestion des services web.

Le projet qu'on a présenté dans ce mémoire est dans le but de découvrir les nouvelles méthodes de la classification des services web basé sur des algorithmes d'apprentissage automatique, en particulier des modèles d'apprentissage profond.

L'apprentissage profond c'est une méthode facile et donne des bons résultats. Au cours du premier chapitre de ce mémoire nous avons expliqué quelques concepts d'apprentissage automatique et l'apprentissage profond, Nous avons décrire dans le deuxième chapitre les définitions les plus liées aux services Web avec ses différents types, et son architecture, ainsi que nous avons traité quelques travaux précédents. Ensuite, dans le troisième chapitre, nous avons exposé la méthodologie que nous avons suivie dans notre travail, notre solution se compose en trois étapes :

1 Modelé pour le langage nous avons utilisé Word2vect
2 Modelé pour l'entrainement avec auto-encodeur
3 Modelé Final divisé en deux parties :
    - Classification
    - Test

On montre les mécanismes utilisés tel que : Word2Vec, Auto-Encodeur.

Nous avons aussi entrainé notre projet sur un ensemble de données *Service* Dataset. Enfin, dans le dernier chapitre, nous avons présenté nos résultats et nous avons comparé les avec les résultats des travaux précédents.

En conclusion, et après avoir atteint les résultats et les avoir comparés avec les résultats des travaux antérieurs, nous avons conclu que notre projet a donné des résultats très satisfaisants par rapport aux résultats de classification pour les méthodes qu'ils sont utilisées le même jeu de données. Cependant, le domaine de classification des web services à développer afin de résoudre

les problèmes des travaux antérieurs et l'ajout des nouvelles techniques et des méthodes qui permettront d'améliorer le niveau de classification.

Les perspectives

Le travail présenté dans ce mémoire, ouvre la voie à plusieurs pistes de recherche qui pourront faire l'objet d'un ensemble de perspectives, Suite aux travaux effectués, plusieurs points restent à développer et à améliorer. Parmi lesquels l'évaluation des méthodes de classification créée par les experts du domaine, leur point de vue à un impact très important pour vérifier sa complétude.

Notre travail sera enrichi et plus satisfaisant avec l'utilisation d'un grand ensemble de jeu de données.

References

[1] C. Asselin, «Intelligence Artificiel,» 2021. [En ligne].
[2] B. Mahesh, «Machine learning algorithms-a review,» *International Journal of Science and Research(IJSR)*, pp. 381-386, 2020.
[3] «Guide de l'intelligence artificielle,» 2022. [En ligne]. Available: https://www.journaldunet.fr/web-tech/guide-de-l-intelligence-artificielle/1501339-dataset-en-machine-learning-definition/.
[4] G. Sain-Cirgue, Apprendre le machine learning en une semaine, 2019, p. machinelearnia.com.
[5] «glossary,» [En ligne]. Available: https://www.coe.int/fr/web/artificial-intelligence/glossary.
[6] «machine learning,» [En ligne]. Available: https://www.jedha.co/blog/les-algorithmes-de-machine-learning.
[7] «naive bayes algorithm,» 11 september 2017. [En ligne].
[8] «Neerja Negi, Poonam Chaudhary & Satish Chandra (2020) Web service,» 2020.
[9] «Mchine Learning,» 2020. [En ligne]. Available: Machine Learning les 9 types dalgorithmes.
[10] «Wikipédia,» 24 april 2022. [En ligne]. Available: https://fr.wikipedia.org/wiki/Apprentissage_semi-supervis%C3%A9.
[11] G. Saint-Cirgue, «Machine Learnia,» 24 April 2022. [En ligne]. Available: https://machinelearnia.com/comment-fonctionne-machine-learning/.
[12] Y. B. Yann Lecun, Deep Learning, Nature Pblishing Group, 2015, pp. 197-387.
[13] «Deep Learning,» [En ligne]. Available: https://www.futura-sciences.com/tech/definitions/intelligence-artificielle-deep-learning-17262/.
[14] Salvail-Bérard, «Réseaux de neurones,» 2012.
[15] «Intelligence Artificielle,» [En ligne]. Available: https://sites.google.com/site/intelligenceartificielledreux/un-cerveau-artificiel.

[16] [En ligne]. Available: https://www-v7labs-com.translate.goog/blog/neural-network-architectures-guide.
[17] J. Patterson, Deep Learning, A practitioner's approach. " O'Reilly Media, Inc.", 2017.
[18] «wikipedia,» [En ligne]. Available: https://fr.wikipedia.org/wiki/R%C3%A9seau_de_neurones_r%C3%A9currents.
[19] «Top 5 Deep Learning Architectures,» 24 july 2018. [En ligne]. Available: https://hub.packtpub.com/top-5-deep-learning-architectures/.
[20] J. Jordan, «autoencoder,» 19 mars 2018. [En ligne]. Available: https://www-jeremyjordan-me.translate.goog/autoencoders/.
[21] H. Bandyopadhyay, «autoencoder guide,» 19 juillet 2022. [En ligne]. Available: https://www-v7labs-com.translate.goog/blog/autoencoders-guide?.
[22] G. Alonso, Casatif, F, H. Kuno et V. Machiraju, «Web services,» *Web services*, pp. 123-149, 2004.
[23] J. Bean, SOA and web services interface design: principales, techninques, and standards, Morgan Kaufmann, 2009.
[24] M. Papazoglou, Web services: principles and technologiy, Pearson Education, 2008.
[25] «What is SOA,» [En ligne]. Available: https://latona.eu/index.php/2018/07/23/what-is-soa/.
[26] M. Hafner et R. Breu, «SOA-Standards & Technology,» *Springer Berlin Heidelberg*, pp. 15-25, 2009.
[27] «What are Web Services? Architecture, Types, Exemple,» 18 May 2022. [En ligne]. Available: https://www.guru99.com/web-service-architecture.html.
[28] E. F.-S. Amal, H. Serge, M. Tarak et S. Alexandru, «Interopérabilité des systèmes multi-agents,» 2004.
[29] F. Rossi, «ServicesWeb-WSDL,» [En ligne]. Available: http://www-inf.it-sudparis.eu/cours/WebServices.
[30] «wikipedia,» 2018. [En ligne]. Available: https://fr.wikipedia.org/wiki/Representational_state_transfer.
[31] «Département Informatique Toulouse,» *Pierre Gambarotto : « Technologies pour Web Services faciles REST, JSON », INPT DSI ENSEEIHT Département Informatique Toulouse , 2009.,* 2009.

[32] «Fiverr,» 2022. [En ligne]. Available: https://fr.fiverr.com/azeem4220/consume-web-services-wisdl-soap-restful?context_referrer=tag_page&source=TagPage&ref_ctx_id=42e35ec55c95e9d84ac752850058ecba&pckg_id=1&pos=7&imp_id=83ce0522-3cec-4384-b40d-4450f67f95f9.
[33] I. Docs, «Service WSDL,» 19 may 2022. [En ligne]. Available: https://www.ibm.com/docs/fr/was/9.0.5?topic=services-wsdl.
[34] S. Savic et Hao Shi, «"TEAMTRACKER"- An Innovative Team Collaboration System,» *International Journal of Computer Networks and Communications,* p. September , 2010.
[35] K. WOOLLAMS, «Uddi.Html,» 19 may 2022. [En ligne]. Available: http://www-igm.univ-mlv.fr/~dr/XPOSE2004/woollams/uddi.html..
[36] «Mamoun Mohamad Jamous et Safaai Bin Deris vers une approche de Web Services,» 2011.
[37] «B. Saravana Balaji,S. Balakrishnan,Journal of Ambient Intelligence and Humanized Computing,» 2020.
[38] «Marco Crasso, Alejandro Zunino and Marcelo Campo ISISTAN Research Institute. UNICEN University. Campus Universitario,,» 2008.
[39] M. S. Alshafaey, S. Ahmed et A. Mohamed F, «A new cloud-based classification methodology (CBCM) for efficient».
[40] Y. Yang, K. Wei , W. Weiru et Z. Yongxin , «Deep Learning forWeb Services Classification,» 2019.
[41] A. S. Mustafa, «WEB SERVICE CLASSIFICATION WITH MULTILAYER PERCEPTRON».
[42] M. Grosvalet, «Word2Vect,» 27 01 2017. [En ligne]. Available: https://www.search-foresight.com/rankbrain-seo/.
[43] J. Brownlee, «softmax function,» 19 octobre 2020. [En ligne]. Available: https://machinelearningmastery-com.translate.goog/softmax-activation-function-with-python.
[44] Y. Derfouf, «Programmation en langage Python,» 2019.
[45] «Natural Language Toolkit,» 27 July 2022. [En ligne]. Available: https://www.techopedia.com/definition/30343/natural-language-toolkit-nltk.
[46] J. Vaughan, février 2018. [En ligne]. Available: https://www-techtarget-com.translate.goog/searchdatamanagement/definition/TensorFlow.
[47] J. Brownlee, «Introduction Python,» 10 May 2016. [En ligne]. Available: https://machinelearningmastery.com/introduction-python-deep-learning-library-keras.

[48] «GENSIM,» 2020. [En ligne]. Available: https://radimrehurek.com/gensim/intro.html.
[49] «Techopedia,» 2022. [En ligne]. Available: https://www.techopedia.com/definition/33860/scikit-learn.
[50] «What is numpy,» [En ligne]. Available: https://numpy-org.translate.goog/doc/stable/user/whatisnumpy.html.
[51] T. KELDENICH, 2 SEPTEMBRE 2021. [En ligne]. Available: https://inside-machinelearning.com/recall-precision-f1-score/.
[52] M.-J. Vieille. [En ligne]. Available: https://www.lovelyanalytics.com/2020/05/26/accuracy-recall-precision/.
[53] M. Riva, 19 January 2021. [En ligne]. Available: https://www.baeldung.com/cs/ml-loss-accuracy.
[54] [En ligne]. Available: https://dataanalyticspost.com/Lexique/reseaux-de-neurones-recurrents/.
[55] O. Carton, «L'essentiel de XML: cours XML,» 2007-2013.
[56] «la syntaxe XML en résumé [introduction à XML: principes, syntaxe, schémas et manipulation],» 19 may 2022. [En ligne]. Available: https://stph.scenari-community.org/lo17/xml/co/xmlUL37resume.html.