République Algérienne Démocratique et Populaire

Ministère d'Enseignement Supérieur et de la Recherche Scientifique
Université ABBAS LAGHROUR- Khenchela-
département MI

Mémoire

Présenté en vue d'obtenir le diplôme de

Master en Informatique (LMD)

Création d'un réseau sémantique par l'utilisation de WordEmbeddings

Présenté Par : BEKHOUCHE MAAMAR

Encadrer Par : Dr BAKHOUCHE ABDELAALI.

Année universitaire : 2019 /2020

Dédicace

A mes très chers parents,

A mes chers frères et sœurs,

A mes Enseignants

A mes collègues de travail,

A mes meilleurs amis.

Je dédie ce mémoire.

Remerciement

Je tiens à remercier Monsieur ABDELAALI BEKHOUCHE, qui m'a encadré

Tout au long de cette thèse et qui m'a fait partager ses brillantes intuitions. Qu'il

soit aussi remercié pour sa gentillesse, sa disponibilité permanente et pour les

nombreux encouragements qu'il m'a prodiguée.

Résumé

Les algorithmes et modèles de Machine Learning utilisent des entrées numériques, et pour cette raison on travaille souvent avec des espaces topologiques voire métriques.

Lorsqu'on traite des données brutes à grande dimensions comme des images ou du spectre audio, on utilise directement des vecteurs de coefficients associées à l'intensité de pixel dans le premier cas, ou les coefficients de densité spectrales dans le second cas.

Le problème avec un texte, c'est que l'on va traiter des mots et groupes de mots comme des entités symboliques non porteuses de sens, c'est à dire indépendantes les unes des autres.

Pour résoudre ce problème nous avons utilisés d'une part le modèle Word Embeddings pour représenter les sens des mots dans la langue arabes, et qui ne dispose pas beaucoup de ressources linguistiques, électroniques et d'autre part nous avons exploité cette représentation dans le cas de plagiat.

Abstract

Machine Learning algorithms and models use digital inputs, and for this reason we often Work with topological or even metric spaces.

When treating large raw data like images or the audio spectrum, we directly use vectors of coefficients associated with the pixel intensity in the first case , or the spectral density coefficients in the second case.

The problem with a text is that we are going to process the words and groups of words as symbolic entities that do not carry meaning, that is, independent of each other.

For to solve this problem we used on the one hand the Word Embeddings model to represent the meanings of words in the Arabic language which does not have many electronic linguistic resources and on the other hand we have exploited this representation in the case of plagiarism.

Table des matières

Listes des figures et tableaux

Figures:

Tableaux

Abréviations

- WE : Word Embeddings
- CBOW : Continious Bag Of Words
- TAL : Traitement Automatique de la langue
- ADT : analyse de données textuelles
- RD : recherche documentaire
- TM : Text Mining
- TF-IDF Term Frequency–Inverse Document Frequency
- Word2vec représentation de mot par un vecteur
- CSV Valeurs séparées par des virgules

Introduction générale

➢ La mise à disposition massive de documents via Internet (pages Web, entrepôts de données, documents numériques, numérisés ou retranscrits, etc.) rend de plus en plus aisée la récupération d'idées.

Malheureusement, ce phénomène s'accompagne d'une augmentation des cas de plagiat. En effet, s'approprier du contenu, peu importe sa forme, sans le consentement de son auteur (ou de ses ayants droit) et sans citer ses sources, dans le but de le présenter comme sa propre œuvre ou création est considéré comme plagiat. De plus, ces dernières années, l'expansion d'Internet a également facilité l'accès à des documents du monde entier (écrits dans des langues étrangères) et à des outils de traduction automatique de plus en plus performants, accentuant ainsi la progression d'un nouveau type de plagiat : le plagiat translingue. Ce plagiat implique l'emprunt d'un texte tout en le traduisant (manuellement ou automatiquement) de sa langue originale vers la langue du document dans lequel le plagiaire veut l'inclure. Pour cette raison nous avons besoins de représenter le sens de segment textuel (le plus souvent des mots) d'une manière compréhensibles dans toutes les langues.

La représentation de sens est une pierre de base pour de nombreuses applications du traitement automatique des langues. Cependant, elle nécessite toutefois des efforts de développement conséquents, qu'il s'agisse d'annoter des corpus ou de produire des lexiques et des outils pour plusieurs applications linguistiques comme le plagiat par exemple.

Il existe aujourd'hui plusieurs approches proposées pour représenter le sens tel que le Word Embeddings qui est une méthode d'apprentissage d'une représentation de mots utilisée notamment en traitement automatique des langues **(TAL)** . Le principe général de cette technique est la représentation de chaque mot d'un dictionnaire par un vecteur de nombres réels, ainsi que d'énormes Ressources de corpus annotés destinées à des traitements variés dans nombreuses langues.

➢ Le but de notre travail et notre souhait, est d'essayer de mettre en évidence l'utilité d'un Modèle de représentation sémantique et son application sur le plagiat pour la langue arabe qui, bien que largement utilisée aujourd'hui, et n'a pas encore bénéficiée de recherche et de résultats matures dans le domaine de l'évaluation automatique.

➢ Ce manuscrit s'articule autour de trois chapitres :

- Nous revenons d'abord dans le premier chapitre de présenter et d'expliquer le contexte d'étude et le concept de base utilisés dans ce travail le Word embeddings, avant de décrire, d'une manière plus approfondie, les différents types de Word Embeddings.

- Nous présenterons ensuite dans le second chapitre quelques particularités du plagiat avant d'aborder ses propriétés et ses types
- Dans le troisième chapitre, nous présenterons notre propre modèle, en utilisant dans ce cas un exemple illustratif, nous illustrerons notamment les démarches de prétraitement du texte.

➢ Ensuite Nous montrerons les phases de représentation et d'apprentissage du sens. Enfin nous utiliserons les mesures de similarités pour calculer la similarité entre deux documents.

Chapitre 1 : Word Embeddings

1.1 Introduction

Pour une meilleure compréhension de Word Embeddings on va citer quelques exemples :

- Vous ouvrez Google, recherchez un article d'actualité quelconque et vous obtenez centaines résultats de recherche en retour.
- Gmail est réputé pour avoir l'un des filtres anti-spam et classer si un email est un spam ou non.
- Vous tapez une phrase dans Google Translate en anglais et vous obtenez une conversion en équivalente en arabe.

Alors, qu'est-ce que les exemples ci-dessus ont en commun ?

Les trois scénarios ci-dessus concernent des quantités énormes de texte pour effectuer différentes tâches, telles que :

- le regroupement : dans l'exemple de recherche Google,

-la classification : dans le second.

- la traduction automatique : dans le troisième.

Nous ne pouvons pas laisser des humains effectuant les trois tâches susmentionnées. Ce n'est ni évolutif ni efficace. Alors

comment pouvons-nous faire en sorte que les ordinateurs d'aujourd'hui effectuent des regroupements, des classifications, ….etc. Bien sûr, un ordinateur peut faire correspondre deux chaînes et vous dire si elles sont identiques ou non. Mais comment pouvons-nous faire en sorte que les ordinateurs vous parlent du football ou de Ronaldo lorsque vous recherchez Messi ?

Comment faites-vous comprendre à un ordinateur que Apple est une entreprise et non un fruit La réponse aux Questions ci-dessus réside dans la création d'une représentation des mots qui capturent leurs significations et leurs relations sémantiques et les différents types de contextes dans lesquels ils sont utilisés.

Et tout cela est implémenté en utilisant **Word Embeddings** ou des représentations numériques de textes afin que les Ordinateurs puissent les traiter.

1.2 Que sont les Word Embeddings ?

En termes très simple, **Word Embeddings** sont les textes convertis en nombres et il peut y avoir différentes représentations numériques du même texte.

Mais avant de plonger dans les détails de **Word Embeddings**, il convient de poser la question suivante: pourquoi avons-nous besoin de **Word Embeddings**?

De nombreux algorithmes d'apprentissage automatique et presque toutes les architectures **d'apprentissage profond (Deep Learning)** sont incapables de traiter des chaînes ou des textes dans leurs forme brute. Ils ont besoin des chiffres comme données pour effectuer tout type de travail, que ce soit une **classification**, une **régression**...etc. En termes généraux et avec la quantité énorme de données présentés dans le format texte.

Il est impératif d'en extraire des connaissances et de créer des applications textuelles comme l'analyse de l'opinion, des critiques d'Amazon, la classification de documents ou d'actualités par Google,… etc. Définissons maintenant formellement les **Word Embeddings**.

Un format de Word Embeddings tente généralement de mapper un mot à l'aide d'un dictionnaire sur un vecteur. Décomposons cette phrase en détails plus fins pour avoir une vision claire. Jetez un coup d'œil à cet exemple. une **phrase** = **"Word Embeddings are Word converted into numbers "**

Un dictionnaire peut être la liste de tous les mots uniques de la phrase. Ainsi, un dictionnaire peut ressembler à ça **Dictionnaire = ['Word', 'Embeddings', 'are', 'Converted', ' into ', 'numbers']**

Une représentation de vecteur d'un mot peut être un vecteur codé à un seul **1** représente la position où le mot existe et **0** partout ailleurs.

La représentation vectorielle du mot «numbers» dans ce format, selon le dictionnaire ci-dessus, est **'numbers' = [0,0,0,0,0,1] et celle de «Converted» est [0,0,0,1,0,0].**

Ceci est juste une méthode très simple pour représenter un mot sous la forme vectorielle.

Examinons différents types de **Word Embeddings** et leurs avantages et inconvénients par rapport aux autres.

Word Embeddings **,** également connu sous le nom de **Représentation distribuée des mots** est un espace vectoriel censé garantir l'affichage des relations syntaxique et sémantique des mots (la distance entre eux) Levy et Goldberg (2014b). Le vecteur d-dimensions d'un mot est nous aide à donner un

contexte a ce mot. Cette idée est apparue pour la première fois dans l'hypothèse de distribution de Harris (1954), déclarant que: "les mots dans des contextes similaires ont des significations similaires" . Une erreur courante dans la compréhension des **Word Embeddings** est que les mots proches dans l'espace vectoriel sont synonymes. Alors qu'en réalité, ces mots sont juste syntaxiquement et / ou sémantiquement similaire Mikolov et al. (2013a). Sur cette base, la distance entre les mots est la clé de la reconnaissance des phrases et des similarités entre documents Levyet al. (2015).

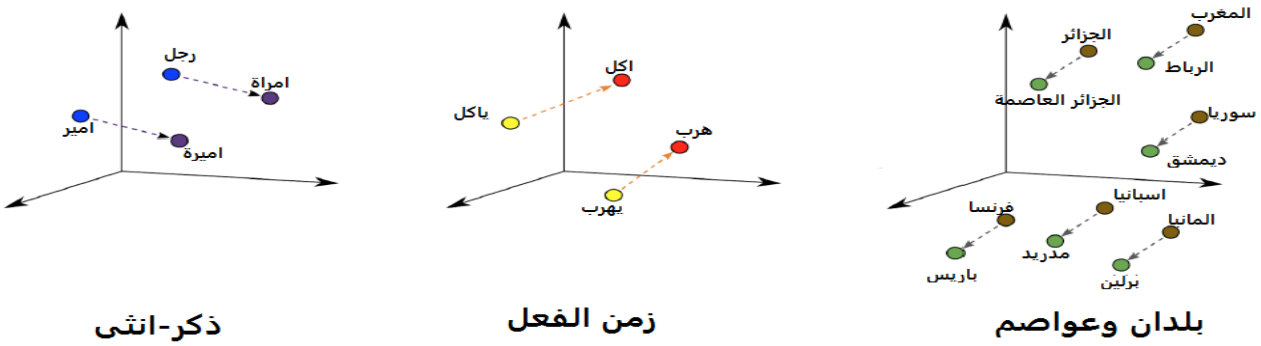


Figure 1.1: Relations entre les mots selon les plongements de mots Mikolov et al. (2013a).

- **Similarité sémantique sur des données textuelles (STS :** Semantic Textual Similarity)

Évaluer la similarité entre entités textuelles est un des problèmes centraux dans plusieurs disciplines tel que l'analyse de données textuelles, Le Plagiat , la recherche documentaire ou l'extraction de connaissances à partir de données textuelles (TextMining).

Dans chacun de ces domaines, les similarités sont en effet utilisées pour une large variété de traitements :

- en analyse de données textuelles (ADT), les similarités sémantique sont utilisées pour la description et l'exploration de données, pour l'identification de structures cachées et pour la prédiction.

- en recherche documentaire (RD), l'évaluation des similarités entre documents et requêtes est utilisée pour identifier les documents pertinents par rapport à des besoins d'information exprimés par les utilisateurs.

- en Text Mining (TM), les similarités sont utilisées pour produire des représentations synthétiques de vastes collections de documents, dans le cadre de procédures d'extraction d'information à partir de données textuelles.

Les techniques mises en œuvre pour calculer les similarités varient bien évidemment selon les disciplines, mais elles s'intègrent cependant le plus souvent dans une même approche générale:

Les entités textuelles sont tout d'abord associées à des représentations spécifiques qui vont servir de base au calcul des similarités.

Voici un schéma illustrant la similarité sémantique entre des entités textuelles :

Figure 1.2: Utilisation de STS dans une conversation.

1.3 Différents types de Word Embeddings

Les différents types de Word Embeddings peuvent être classés en deux catégories:

- **Embeddings basée sur la fréquence.**
- **Embeddings basée sur la prédiction.**

Essayons de comprendre chacune de ces méthodes en détail.

1.3.1 Embeddings basée sur la fréquence

Il existe généralement trois types de vecteurs que nous rencontrons dans cette catégorie.

- **Vecteur par comptage**
- **Vecteur TF-IDF**
- **Vecteur de co-occurrence**

1.3.1.1 Vecteur par comptage

Considérons un corpus C des documents D {d1, d2… ..dk} et N mots uniques extraits du corpus C. Les N mots formeront notre dictionnaire et la taille de la matrice de vecteur par comptage M sera donnée par **k* N**. Chaque ligne de la matrice M contient la fréquence des mots dans le document D (i). Laissez-nous comprendre cela en utilisant un exemple simple.

D1: هو طفل كسول .هي كسولة.

D2: امينة هي كسولة.

le corpus C = [‘**هو**’,’**هي**’,’**كسول**’,’**طفل**’,’**امينة**’,’**كسولة**’]

Alors, k=2, N=6

La matrice de comptage M de taille 2 * 6 sera représentée par :

	هو	هي	كسول	طفل	امينة	كسولة
D1	**1**	**1**	**1**	**1**	**0**	**1**
D2	**0**	**1**	**0**	**0**	**1**	**1**

Maintenant, une colonne peut également être comprise comme un **mot vecteur** pour le mot correspondant dans la matrice M. Par exemple, le mot vecteur pour '**كسول**' dans la matrice ci-dessus est **[1,0]** et ainsi de suite. Ici, les lignes correspondent aux documents du corpus et les colonnes correspondent aux mots du dictionnaire.

La deuxième ligne de la matrice ci-dessus peut être lue comme suit

- D2 contient «**هي**»: une fois, «**امينة**»: une fois et «**كسولة**» une fois.

Maintenant, il peut y avoir pas mal de variations lors de la préparation de la matrice M. Les variations seront Généralement La façon dont le dictionnaire est préparé. Pourquoi ?

Parce que dans les applications du monde réel, nous pourrions avoir un corpus contenant des millions de documents. Et avec des millions de documents, nous pouvons extraire des centaines de millions de mots uniques. Donc, fondamentalement, la matrice qui sera préparée comme ci-dessus sera très maigre et inefficace pour tout calcul. Ainsi, une alternative à l'utilisationde chaque mot unique en tant qu'élément du dictionnaire serait de choisir les 10 000 mots les plus utilisés en fonction de la Fréquence, puis de préparer un dictionnaire.

Le nombre de voies est pris pour chaque mot. On peut soit prendre la fréquence (nombre de fois qu'un mot est apparu dans le document), soit la présence (le mot est-il apparu dans le document ?) Comme étant l'entrée dans la matrice de comptage M.

Mais en général, la méthode de la fréquence est préférée.

Vous trouverez ci-dessous une image représentative de la matrice M pour faciliter la compréhension.

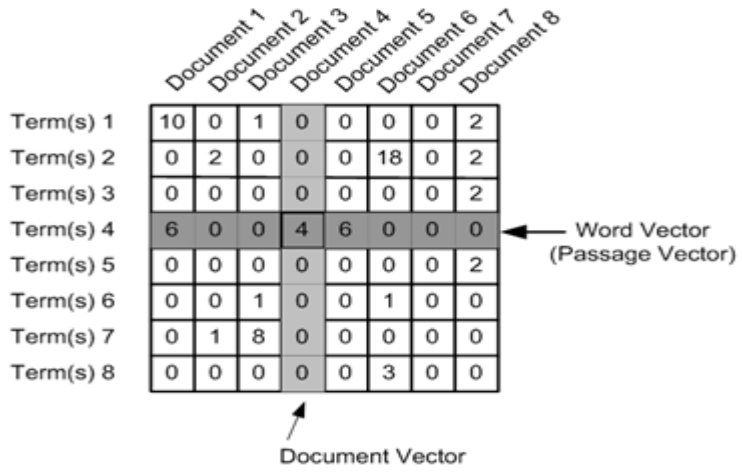


Figure 1.3 : La matrice de comptage M

1.3.1.2 Vectorisation TF-IDF(term frequency–inverse document frequency)

C'est une autre méthode qui est basée sur la méthode de la fréquence, mais elle diffère de la vectorisation par comptage en ce sens qu'elle prend en compte non seulement l'occurrence d'un mot dans un seul document mais dans l'ensemble du corpus. Alors, quelle est la raison derrière cela ? Essayons de comprendre.

Des mots courants tels que ‘is’, ‘the’, ‘a’ etc. ont tendance à apparaître assez fréquemment par rapport aux mots qui sont importants pour un document. Par exemple, un document **A** sur Lionel Messi va contenir davantage d’occurrences du mot «Messi» par rapport à d’autres documents. Mais des mots courants tels que «the», etc. vont également être présents à une fréquence plus élevée dans presque tous les documents.

Nous souhaiterions minimiser les mots courants présents dans presque tous les documents et accorder plus d’importance aux mots figurant dans un sous-ensemble de documents.

- **TF-IDF** travaille en pénalisant ces mots courants en leur attribuant des poids plus faibles tout en donnant une importance aux mots comme Messi dans un document particulier. Alors, comment

fonctionne exactement **TF-IDF**?

Examinons le tableau ci-dessous, qui donne le nombre de termes (mots / mots) dans deux documents.

Document 1

Term	Count
This	1
is	1
about	2
Messi	4

Document 2

Term	Count
This	1
is	2
about	1
Tf-idf	1

Définissons maintenant quelques termes liés à TF-IDF.

TF = (Nombre de fois que le terme t apparaît dans un document) / (Nombre de termes dans ce document) Donc, TF (This, Document1) = 1/8

TF (This, Document2) = 1/5

Cela dénote la contribution du mot au document, c'est-à-dire que les mots relatifs au document doivent être fréquents Exemple: un document sur Messi doit contenir le mot 'Messi' en grand nombre.

IDF = log (N / n), où N est le nombre de documents et n le nombre de documents dans lesquels le terme t est apparu.

Donc, IDF (This) = log (2/2) = 0.

Alors, comment expliquons-nous le raisonnement derrière IDF? Idéalement, si un mot est apparu dans Tout les document, alors ce mot n'est probablement pas pertinent pour un document particulier. Mais s'il est apparu dans un sous-ensemble de documents, alors le mot est probablement **pertinent** pour les documents dans lesquels il est présent. Calculons l'IDF pour le mot 'Messi'.

IDF (Messi) = log (2/1) = 0.301. IDF(This) =log (2/2) = 0.

Comparons maintenant la TF-IDF pour un mot commun **"This"** et un mot **"Messi"** qui semble être pertinent pour le Document 1.

TF-IDF (This, Document1) = (1/8) * (0) = 0

TF-IDF (This, Document2) = (1/5) * (0) = 0

TF-IDF (Messi, Document1) = (4/8) * 0,301 = 0,15

On peut constater pour Document1, la méthode TF-IDF pénalise fortement le mot 'This' mais attribue

un poids plus important à 'Messi'. Cela peut donc être compris comme «Messi» est un mot important pour Document1 dans le contexte du corpus entier.

1.3.1.3 Matrice de co-occurrence avec une fenêtre de contexte fixe

La grande idée - Des mots similaires ont tendance à apparaître ensemble et auront un contexte similaire, par exemple Pomme est un fruit. La mangue est un fruit. La pomme et la mangue ont tendance à avoir un contexte similaire, à savoir le fruit. Avant de plonger dans les détails de la construction d'une matrice de co-occurrence, il convient de clarifier deux concepts: **la co-occurrence et la fenêtre de contexte.**

- Co-occurrence : Pour un corpus donné, la co-occurrence d'une paire de mots disons w1 et w2 est le nombre de fois où ils sont apparus ensemble dans une fenêtre de contexte.

- Fenêtre contextuelle : La fenêtre contextuelle est spécifiée par un nombre et la direction. Alors, que signifie une
fenêtre de contexte de 2 (autour de)? Voyons un exemple ci-dessous,

Quick Brown Fox Jump Over The Lazy Dog

Les mots verts sont une fenêtre contextuelle de 2 (**autour de)** pour le mot 'Fox' et pour le calcul de la **co-occurrence**, seuls Ces mots seront comptés. Voyons la fenêtre de contexte pour le mot 'Over'.

Quick Brown Fox Jump Over The Lazy Dog

Prenons maintenant un exemple de corpus pour calculer une matrice de co-occurrence.

Corpus = **He is not lazy. He is intelligent. He is smart**.

	He	Is	Not	lazy	intelligent	smart
He	0	4	2	1	2	1
is	4	0	1	2	2	1
not	2	1	0	1	0	0
lazy	1	2	1	0	0	0
intelligent	2	2	0	0	0	0
smart	1	1	0	0	0	0

Laissez-nous comprendre cette matrice de co-occurrence en voyant deux exemples dans le tableau ci-dessus. Rouge et la boîte bleue.

Boîte rouge - C'est le nombre de fois que " He " et "is" sont apparus dans la fenêtre de contexte 2 et on peut voir que le nombre est égal à 4. Le tableau ci-dessous vous aidera à visualiser le nombre.

He	**Is**	not	lazy	He	is	intelligent	He	Is	Smart
He	Is	not	lazy	**He**	**is**	intelligent	He	Is	Smart
He	Is	not	lazy	He	**is**	intelligent	**He**	Is	Smart
He	Is	not	lazy	He	is	intelligent	**He**	**Is**	Smart

tandis que le mot " lazy " n'est jamais apparu avec "intelligent" dans la fenêtre de contexte et a donc été attribué un 0 dans la zone bleue.

-**Avantages de la matrice de co-occurrence :** Il préserve la relation **sémantique** entre les mots. Autrement dit, l'homme et la femme ont tendance à être plus proches que l'homme et la pomme.

- **Inconvénients de la matrice de co-occurrence :**Il faut beaucoup de mémoire pour stocker la matrice de co-occurrence.

Mais, ce problème peut être contourné en factorisant la matrice hors du système, par exemple dans les clusters Hadoop, etc. Et peut être enregistré.

1.3.2 Embeddings basée sur la prédiction

Pré-requis : cette section repose sur les réseaux de neurones et les mécanismes de mise à jour des pondérations.

Jusqu'à présent, nous avons vu des méthodes déterministes pour déterminer les vecteurs de mots. Mais ces méthodes se Sont avérées limitées dans leurs représentations verbales jusqu'à ce que Mitoloet el ait introduit **Word2vec (répresentation de mot par un vecteur)** dans la communauté des PNL.

Ces méthodes reposaient sur la prédiction, dans le sens où elles fournissaient des probabilités aux mots et constituaient l'état de la technique pour des tâches telles que les analogies et les similarités de mots. Ils étaient également capables de réaliser des tâches comme **King-Man + Femme = Reine**, ce qui est considéré comme un résultat presque magique.

- Word2vec n'est pas un algorithme unique mais une combinaison de deux techniques - **CBOW (sac continu de mots)** et Modèle **Skip-gram**. Ces deux réseaux sont des **réseaux neuronaux** peu profonds qui mappent un ou plusieurs mots à la variable cible, qui est également un ou plusieurs mots. Ces deux techniques apprennent des poids qui agissent comme des représentations de vecteur de mots.

Discutons ces deux méthodes séparément et gagnons en intuition dans leur travail.

1.3.2.1 CBOW (sac continu de mots)

La façon dont fonctionne CBOW est qu'il a tendance à prédire la **probabilité** d'un mot dans un **contexte** donné. Un **contexte** peut être un mot unique ou un groupe de mots. Mais pour plus de simplicité, On va prendre un mot de contexte unique et essayons de prédire un mot cible unique.

Supposons que nous ayons un corpus **C = "Hey, this is sample corpus using only one context Word."** Et que nous avons défini une **fenêtre de contexte** égale a 1. Ce corpus peut être converti en

un ensemble d'entraînement Pour un modèle **CBOW** comme suit. L'entrée est indiquée ci-dessous.

La matrice de droite dans l'image ci-dessous contient le code **one-hot (0 et 1)** de l'entrée de gauche.

Input	Output			Hey	This	is	sample	corpus	using	only	one	context	word
Hey	this		Datapoint 1	1	0	0	0	0	0	0	0	0	0
this	hey		Datapoint 2	0	1	0	0	0	0	0	0	0	0
is	this		Datapoint 3	0	0	1	0	0	0	0	0	0	0
is	sample		Datapoint 4	0	0	1	0	0	0	0	0	0	0
sample	is		Datapoint 5	0	0	0	1	0	0	0	0	0	0
sample	corpus		Datapoint 6	0	0	0	1	0	0	0	0	0	0
corpus	sample		Datapoint 7	0	0	0	0	1	0	0	0	0	0
corpus	using		Datapoint 8	0	0	0	0	1	0	0	0	0	0
using	corpus		Datapoint 9	0	0	0	0	0	1	0	0	0	0
using	only		Datapoint 10	0	0	0	0	0	1	0	0	0	0
only	using		Datapoint 11	0	0	0	0	0	0	1	0	0	0
only	one		Datapoint 12	0	0	0	0	0	0	1	0	0	0
one	only		Datapoint 13	0	0	0	0	0	0	0	1	0	0
one	context		Datapoint 14	0	0	0	0	0	0	0	1	0	0
context	one		Datapoint 15	0	0	0	0	0	0	0	0	1	0
context	word		Datapoint 16	0	0	0	0	0	0	0	0	1	0
word	context		Datapoint 17	0	0	0	0	0	0	0	0	0	1

Figure 1.4 : Présentation de code one-hot des mots

La cible pour un **datapoint** unique, par exemple, le point de données 4 est indiquée ci-dessous

Hey	this	is	sample	corpus	using	Only	One	context	Word
0	0	0	1	0	0	0	0	0	0

Cette matrice illustrée dans l'image ci-dessus est envoyée dans un réseau de neurones avec trois couches: Une couche d'entrée, une couche cachée et une couche de sortie.

La couche en sortie est une couche **softmax** qui est utilisée pour additionner les probabilités obtenues dans la couche en Sortie La somme de ces probabilités décimales doit être égale à 1 .

Voyons d'abord une représentation schématique du modèle CBOW.

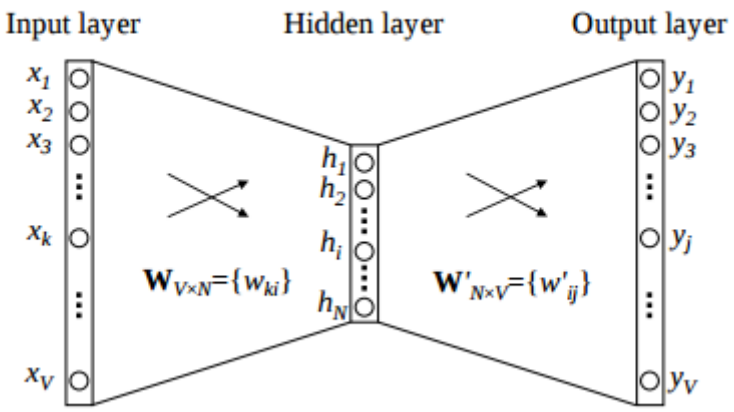


Figure 1.5 : représentation schématique du modèle CBOW

Et voyons maintenant comment la propagation en avant fonctionnera pour calculer l'activation de la couche cachée.

La représentation matricielle de l'image ci-dessus pour un seul point de données est présentée ci-dessous.

Figure 1.6 : La représentation matricielle du model CBOW

La couche d'entrée et la cible sont toutes deux codées à one-hot la taille [1 * V]. Ici V = 10.

Il y a deux séries de poids. l'un se trouve entre la couche d'entrée et la couche cachée et l'autre entre la couche cachée et la couche de sortie. Taille de la matrice de couche cachée en entrée = [V*N], taille de matrice cachée -en sortie = [N*V]: N est le nombre de dimensions dans lesquelles nous choisissons de représenter notre mot. Il est arbitraire et constitue un hyper-paramètre pour un réseau neuronal.

De plus, **N est le nombre de neurones** dans la couche cachée. Ici, N=4.

Il n'y a pas de **fonction d'activation** entre les couches (plus précisément, je parle d'activation linéaire).

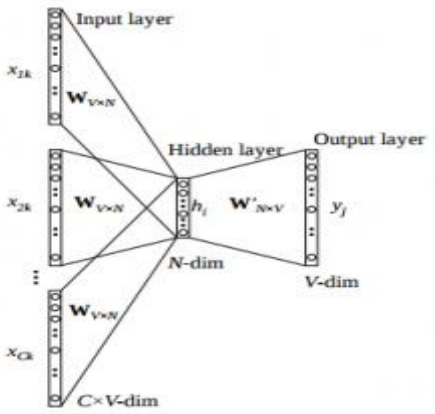
L'entrée est multipliée par les poids cachée entrés et appelée **activation cachée**. C'est simplement la ligne correspondante dans la matrice cachée en entrée copiée.

L'entrée cachée est multipliée par les poids de sortie cachés et la sortie est calculée.

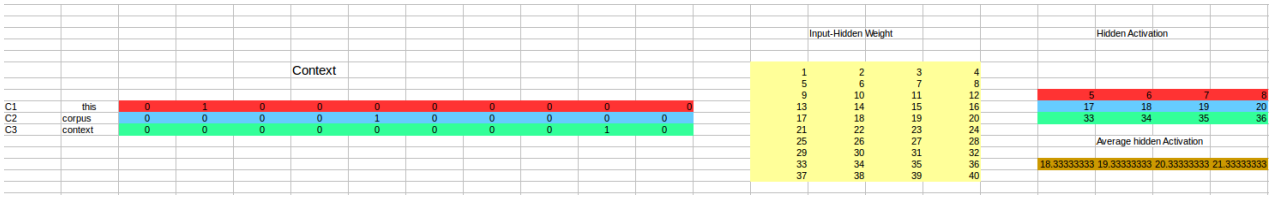
L'erreur entre la sortie et la cible est calculée et renvoyée pour réajuster les poids.

Le poids entre la couche cachée et la couche en sortie est considéré comme la représentation vectorielle du mot.

Nous avons vu les étapes ci-dessus pour un seul mot de contexte. Maintenant, qu'en est-il si nous avons plusieurs mots de contexte? L'image ci-dessous décrit l'architecture de plusieurs mots de contexte.



Vous trouverez ci-dessous une représentation matricielle de l'architecture ci-dessus pour faciliter la compréhension.



L'image ci-dessus prend 3 mots de contexte et prédit la probabilité d'un mot cible. L'entrée peut être considérée comme trois vecteurs codés à une séquence dans la couche d'entrée, comme indiqué ci-dessus en rouge, bleu et vert.

Ainsi, la couche d'entrée aura 3 [1 * V] vecteurs dans l'entrée comme indiqué ci-dessus et 1 [1 * V] dans la couche de sortie. Le reste de l'architecture est identique à celui d'un CBOW à 1 contexte.

Les étapes restent les mêmes, seul le calcul de l'activation cachée change. Au lieu de simplement copier les lignes correspondantes de la matrice de pondération cachée en entrée dans la couche cachée, une moyenne est établie sur toutes les lignes correspondantes de la matrice. Nous pouvons comprendre cela avec la figure ci-dessus.

Le vecteur moyen calculé devient l'activation cachée. Donc, si nous avons trois mots de contexte pour un seul mot cible, nous aurons trois activations cachées initiales qui seront ensuite moyennées pour obtenir l'activation finale.

À la fois pour un mot de contexte unique et pour un mot de contexte multiple, j'ai montré les images jusqu'au calcul des activations cachée, car c'est la partie où CBOW diffère d'un simple réseau MLP. Les étapes après le calcul de la couche cachée sont identiques à celles du MLP.

La différence entre MLP et CBOW sont mentionnées ci-dessous pour clarification:

La fonction objectif dans MLP est une MSE (moyenne erreur quadratique) alors que dans CBOW, elle est une probabilité logarithmique négative d'un mot donné à un ensemble de contexte i.e.

log (p (wo / wi)), où p (wo / wi) est donné comme

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^{\top} v_{w_I}\right)}{\sum_{w=1}^{W} \exp\left({v'_w}^{\top} v_{w_I}\right)}$$

wo: mot de sortie wi: mots de contexte.

Le gradient d'erreur en ce qui concerne les poids de sortie cachés et les poids de la couche en entrée est différent puisque MLP a des activations **sigmoïdes** (généralement) mais CBOW a des activations **linéaires**.

Cependant, la méthode de calcul du gradient est identique à celle d'un MLP.

- Avantages de CBOW:

Étant probabiliste, c'est la nature, elle est supposée être supérieure aux méthodes déterministes (en général).

Il manque de mémoire. Il n'a pas besoin d'énormes besoins en RAM, comme celui d'une matrice de co-occurrence, où il doit

stocker trois énormes matrices.

Inconvénients de CBOW:

CBOW prend la moyenne du contexte d'un mot (comme indiqué ci-dessus dans le calcul de l'activation cachée). Par exemple, Apple peut être à la fois un fruit ou une entreprise, mais CBOW prend une moyenne des deux contextes et le place entre un cluster pour les fruits et les entreprises.

Former un CBOW à partir de zéro peut prendre une éternité s'il n'est pas correctement optimisé.

1.3.2.2 Modèle Skip – Gram

Il suit la même topologie que **CBOW**. Il renverse simplement l'architecture de ce dernier. Le but de **skip-gram** est de prédire le contexte donné à un mot. Prenons le même corpus que celui sur lequel nous avons construit notre modèle CBOW.

C = " Hey, this is sample corpus using only one context Word." Construisons les données d'apprentissage.

Input	Output(Context1)	Output(Context2)
Hey	this	<padding>
this	Hey	is
is	this	sample
sample	is	corpus
corpus	sample	corpus
using	corpus	only
only	using	one
one	only	context
context	one	word
word	context	<padding>

Le vecteur d'entrée pour **skip-gram** sera similaire à un modèle **CBOW** à 1 contexte. De plus, les calculs jusqu'à l'activation de la couche cachée seront les mêmes. La différence sera dans la variable cible. Comme nous avons défini une fenêtre de contexte de 1 sur les deux côtés, il y aura « **deux» une variable cible codée one hot** et **deux sorties correspondantes,** comme l'indique la section bleue de l'image. Deux erreurs distinctes sont calculées par rapport aux deux variables cibles et les deux vecteurs d'erreur obtenus sont ajoutés élément par élément pour obtenir un vecteur d'erreur final qui est propagé en retour pour mettre à jour les poids.

Les pondérations entre la couche d'entrée et la couche cachée sont prises comme représentation du mot vecteur après la L'entrainement . La fonction de perte ou l'objectif est du même type que le modèle **CBOW**.

L'architecture skip-gram est illustrée ci-dessous.

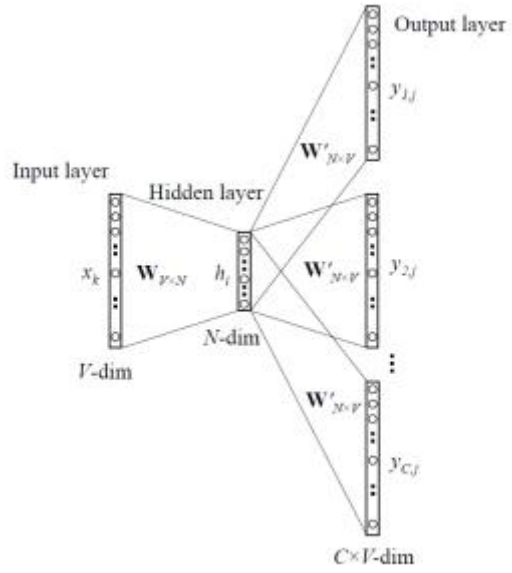


Figure 1.7 : représentation schématique du modèle SKIP-GRAM

Pour une meilleure compréhension, la structure de style matricielle avec calcul est présentée ci-dessous.

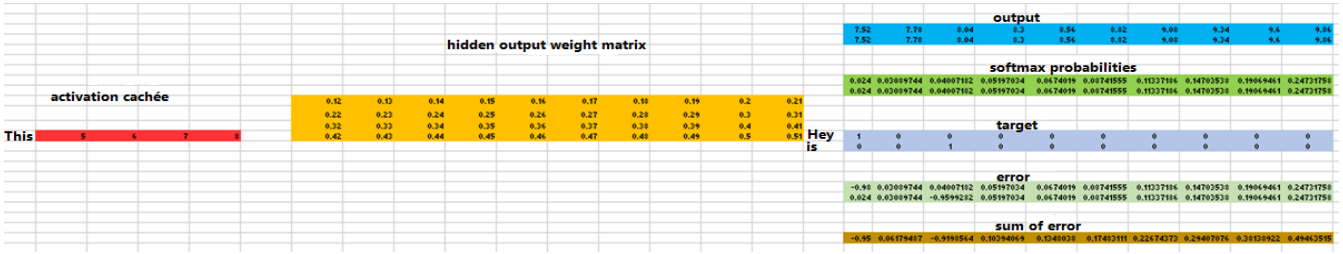


Figure 1.8 : La représentation matricielle du model CBOW

Laissez-nous décomposer l'image ci-dessus. Taille de la couche d'entrée [1 * V], taille de la matrice de poids masquée d'entrée [V * N], nombre de neurones dans la couche masquée N, taille de la matrice de poids de la sortie cachée [N * V].

taille de la couche de sortie C [1 * V]. Dans l'exemple ci-dessus, C est le nombre de mots de contexte = 2, V = 10, N = 4.

La ligne en rouge correspond à l'activation masquée correspondant au vecteur codé one hot unique en entrée.

C'est fondamentalement la ligne correspondante de la matrice cachée en entrée.La matrice jaune est le poids entre la couche cachée et la couche en sortie. La matrice bleue est obtenue par la multiplication

de la matrice **d'activation cachée** et des **poids de sortie cachés.**

Output = (hidden activation) * (hidden output weights matrix) Exemple

Output[0] = 5*0.12 + 6*0.22 + 7*0.32 + 8*0.42 = 7.52

Output[1] = 5*0.13 + 6*0.23 + 7*0.33 + 8*0.43= 7.78

Il y aura deux lignes calculées pour deux mots cibles (contextuels).Chaque ligne de la matrice bleue est convertie en ses Probabilités **softmax** individuellement, comme indiqué dans la zone verte.$\sum_{k=1}^{10} \mathbf{softmax}[\mathbf{k}] = \mathbf{1}$

La matrice grise contient les vecteurs codés one-hot des deux mots de contexte **(cible). Hey ET This**

L'erreur est calculée en soustrayant la première ligne de la matrice grise (cible) de la première ligne de la matrice verte (sortie) par élément. **Error[i] = softmax[i] – target[i]** Ceci est répété pour la ligne suivante. Par conséquent, pour **n** mots de contexte cible, nous aurons **n** vecteurs d'erreur. La somme par élément est prise sur tous les vecteurs d'erreur pour obtenir un vecteur d'erreur final. Ce vecteur d'erreur est renvoyé pour mettre à jour les poids.

- Avantages du modèle Skip-Gram

Le modèle de Skip-gram peut capturer deux sémantiques pour un seul mot. c'est-à-dire qu'il aura deux représentations vectorielles d'Apple. Un pour l'entreprise et un autre pour le fruit.Le skip-gram avec sous-échantillonnage négatif est supérieur à toutes les autres méthodes. Un excellent outil interactif pour visualiser CBOW et ignorer gramme en action.

Puisque le Word Embeddings sont des représentations numériques de similarité contextuelles entre les mots, ils peuvent être manipulés et utilisés pour effectuer des tâches incroyables.

Chapitre 2 : Le Plagiat

2.1 Introduction

Le **plagiat** est une faute d'ordre moral, civil ou commercial, qui peut être sanctionnée au pénal, elle consiste à copier un auteur ou accaparer l'œuvre d'un créateur dans le domaine des arts sans le citer ou le dire, ainsi qu'à fortement s'inspirer d'un modèle que l'on omet, délibérément ou par négligence, de désigner.

Il est souvent assimilé à un vol immatériel.

Le « plagiaire » est celui qui s'approprie indûment ou frauduleusement tout ou partie d'une œuvre littéraire, technique ou artistique (et certains étendent ceci — par extension — à un style, des idées, ou des faits).

Le plagiat diffère de l'art du pastiche, qui consiste à imiter ou à calquer les codes ou les figures d'expression d'un auteur, dans un but d'ironie, d'humour ou de dérision.

Le plagiat, qui ne fait pas l'objet d'une définition juridique, est une forme de contrefaçon.

Certains opèrent une distinction entre le plagiat, emprunt grossier, et le « **démarquage** », où le texte subit des modifications variées pour brouiller les pistes.

De même les anglophones rapprochent parfois aussi du plagiat le *misquoting* (sources manquantes ou insuffisantes), le *Self Plagiarism* (« auto-plagiat », nullement illégal voire impossible en tant qu' oxymorique, souvent utilisé par les artistes (Warhol en particulier, qui en a fait sa marque), mais parfois assimilé à une sorte de manquement à une nouvelle « norme morale » ou à une sorte de « fraude par recyclage » quand il n'est pas signalé) de même que pour le *slice and dice plagerism* (recomposition de ses propres écrits)[2]. Dans le domaine scientifique, l'auto-plagiat ou publication redondante est prohibé.

2.2 Définition du plagiat textuel

Le plagiat textuel, plagiat d'un écrit, est un plagiat impliquant le vol d'une œuvre écrite. Le copier/coller de tout ou partie d'un texte, sans citer sa source afin de faire passer ce texte comme étant

sa propre production est la forme de plagiat textuel la plus triviale et répandue, mais un plagiat textuel peut se présenter sous diverses formes.

Le vol d'idées était également du plagiat à partir du moment où elles avaient donné lieu à un support, comme une œuvre écrite en l'occurrence.

C'est pourquoi, s'attribuer les idées originales d'une autre personne, peu importe que ce soit sous la forme d'une théorie, d'une opinion, d'une d'interprétation de données, d'une méthode, d'un algorithme ou d'une formule, sans citer cette personne même si ces idées sont dites avec ses propres mots, est un plagiat du moment où ces idées avaient déjà été mises à l'écrit.

Il n'est donc pas nécessaire de faire du mot à mot (copier/coller) pour plagier à l'écrit. Il est également considéré comme plagiat le fait de paraphraser ou reformuler un texte.

2.3 Caractéristique et situations de plagiat

• L'incohérence et changement de style d'écrit des paragraphes : Longues passages de texte commun

• D'intégrer dans un travail des images, des graphiques, des statistiques, des données (ex. : fichier Excel, corpus en ligne, CSV) sans en mentionner l'origine ou la source.

• De traduire des textes a partir d'un document, soit partiellement ou totalement, et de copier et coller la traduction sans indiquer la source de document.

• De remplacer les mots et les concepts du texte d'une source par leurs synonymes.

• copier et voler les mots, des passages ou des d'un livre, ou d'un site internet sans en préciser la source

• Fabriquer ou Falsifier les références bibliographiques.

2.4 Les types de plagiat

Le plagiat peut se définir sous plusieurs et différentes formes telles que :

2.4.1 Verbatim (copier/coller) : Les passages sont copiés mot à mot sans aucun changement

Texte Source	Texte Plagié
جائحة فيروس كورونا أو جائحة كوفيد 19 . والمعروفة أيضا باسم جائحة فيروس كورونا . هي جائحةٌ عالميةٌ	جائحة فيروس كورونا أو جائحة كوفيد 19 . والمعروفة أيضا باسم جائحة فيروس كورونا . هي جائحةٌ عالميةٌ

FIGURE 2.1 – exemple d'un cas de verbatim plagiat.

2.4.2 Plagiat avec Paraphraser :

n'est pas un type de plagiat tant les sources sont citez correctement . Cependant, la paraphrase devient du plagiat lorsque la syntaxe des textes des autres sont modifié ou changer (réécrit).

Texte Source	Texte Plagié
جائحة فيروس كورونا أو جائحة كوفيد 19 . والمعروفة أيضا باسم جائحة فيروس كورونا . هي جائحةٌ عالميةٌ	فيروس كورونا هو جائحة كوفيد 19 . اسمه أيضا جائحة فيروس كورونا . انها الجائحةٌ العالميةٌ

FIGURE 2.2 – exemple de cas de plagiat paraphraser

2.4.3 Plagiat par traduction :

les contenus plagié sont traduits (manuellement ou automatiquement).

Texte Source	Texte Plagié
جائحة فيروس كورونا أو جائحة كوفيد 19 . والمعروفة أيضا باسم جائحة فيروس كورونا . هي جائحةٌ عالميةٌ	Coronavirus ou pandémie de COVID-19. Aussi connu sous le nom de pandémie de coronavirus. C'est une pandémie mondiale

FIGURE 2.3 – exemple de cas de plagiat avec traduction .

• Le plagiat des idées : consiste à la réutilisation d'une pensée ou les concepts et les opinions originale à partir d'un texte source sans indique la référence de l'idée .

• Auto plagiat : Quand un auteur réutiliser des passages de texte de ces propre publications ou dans un nouveau travaille.

• plagiat des images et vidéo : intégration dans un travail des images, des schémas, des graphiques ou des vidéos sans citer la source .

2.5 Le plagiat en pleine expansion

De plus en plus de gens ont accès à Internet. Ce phénomène est illustré dans l'issue d'un rapport sur la santé d'Internet rendu par la société Mozilla. On peut, par exemple, constater que fin 2016, plus de 50% de la population mondiale avait accès à Internet, contre 30%début 2011

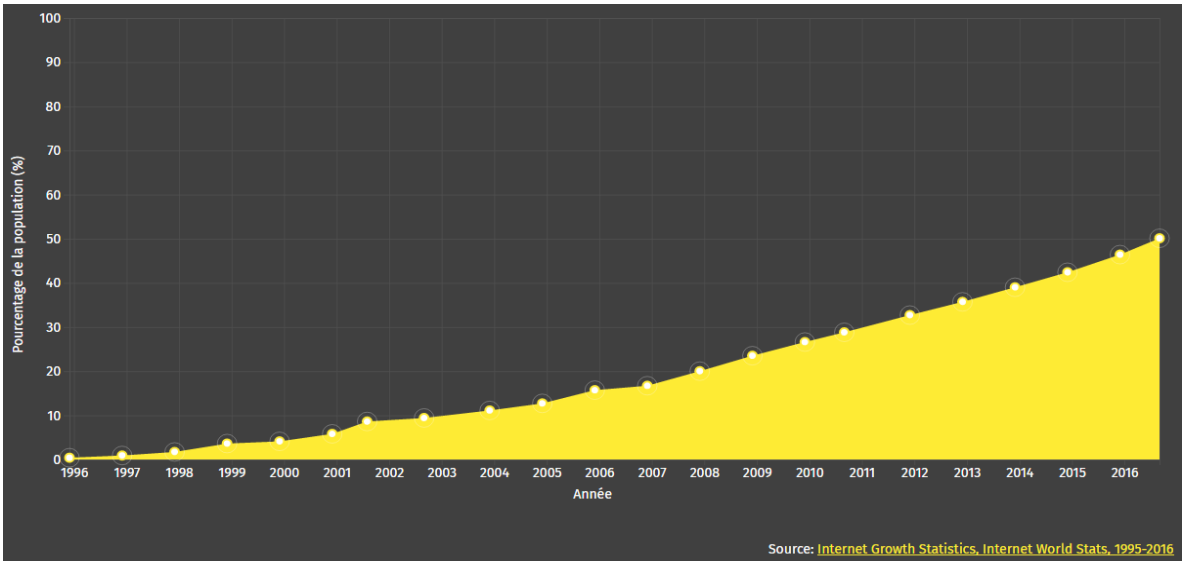


Figure 2.4 – Pourcentage de la population mondiale ayant accès à Internet en fonction l'année.
source : https://internethealthreport.org/v01/fr/digital-inclusion/

La mise à disposition massive de plusieurs milliards de documents via Internet et la digitalisation de textes (pages Web, base de données, documents numériques, numérisés ou retranscrits) rend de plus en plus aisé le copier/coller et la récupération d'idées. Cette nouvelle ère numérique fluidifie les échanges de données et diminue les scrupules des gens à s'approprier le travail des autres.

2.6 La prévention et la lutte contre le plagiat

2.6.1 La prévention du plagiat monolingue

De nos jours de nombreuses recherches s'intéressent à la détection du plagiat. Certaines se concentrent plutôt sur l'alignement des passages similaires entre deux textes (Barrón-CedeñoetRosso, 2009; Barrón-Cedeño et al.,2013b). D'autres techniques, tentent d'analyser les changements de style d'écriture au sein d'un texte pour en détecter les zones suspectes (OberreuteretVelásquez, 2013) et ainsi faire l'hypothèse que ces zones n'ont pas été écrites par le même auteur que le reste du texte et donc qu'elles ont été plagiées.

Le premier cas est ce que l'on appelle la détection de plagiat **extrinsèque**. Soit un jeu de documents suspects et un jeu de documents sources potentiels, la tâche est de trouver des passages plagiés dans les documents suspects et les passages sources correspondants dans les documents sources.

Le second, est quant à lui appelé la détection de plagiat **intrinsèque**. Soit un jeu de documents suspects, la tâche est d'extraire tous les passages plagiés sans les comparer à des documents sources potentielles, donc sans avoir accès à un corpus de documents sources externes.

La campagne d'évaluation PAN2 s'impose depuis plusieurs années comme la campagne d'évaluation de référence dans les tâches assimilées à la détection du plagiat. Elle a fortement contribué à l'avancée de l'état de l'art en matière de détection du plagiat extrinsèque et intrinsèque.

On peut résumer les différentes avancées effectuées dans ces deux types de détection de plagiat dans la figure suivante.

Figure 2.5 : Adaptation de la taxonomie de eissen et stein (2006) des différentes types de plagiat et de leurs moyens de détection

2.6.1.1 La détection intrinsèque

Utilise des approches dites stylo-métriques. Ces dernières suggèrent qu'en analysant les caractéristiques d'un texte, on peut en reconnaître l'auteur, et ainsi, si un passage du document ne possède pas les mêmes caractéristiques que le reste du document, on peut en déduire que ce passage aura été emprunté à un autre auteur.

La stylo-métriques ou l'étude stylo-métriques d'un texte est une analyse à mi-chemin entre une analyse linguistique et statistique. Elle exploite des variables s stylo-métriques, qui sont des caractéristiques linguistiques du texte, afin d'établir des statistiques propre à l'écriture du texte étudié. Effectuer l'analyse stylo-métriques d'un document consiste à surveiller les variations du style d'écriture du document en surveillant l'évolution des variables stylo-métriques au sein de celui-ci afin d'en détecter les irrégularités et ainsi pouvoir déterminer si certains passages, appelés phases stylistiques, sortent de la norme par rapport au reste du texte. Si de tels passages existent, dans le cadre

d'une détection intrinsèque de plagiat, on pourra faire l'hypothèse que ces passage sont été écrits par un auteur différent (du reste du texte) et donc qu'ils ont été plagiés.

Cette approche de détection du plagiat est utile lorsqu'aucune collection de référence n'est disponible et que donc aucun algorithme de comparaison de documents deux à deux ne peut être appliqué pour retrouver les portions de textes similaires dans deux documents comparés.

2.6.1.2 La détection extrinsèque

La tâche de détection de plagiat extrinsèque est souvent découpée en deux sous-tâches, la tâche de collecte de documents sources candidats et la tâche de comparaison (la recherche d'alignements de passages similaires) de documents deux à deux, entre le document suspect en cours d'analyse et chacune des sources renvoyées par la tâche de collecte. Le sujet de la thèse concernant seulement la seconde tâche, nous porterons notre attention uniquement sur celle-ci :

La détection de similarités textuelles entre deux documents.

Les copier/coller sont en théorie les similitudes textuelles les plus facilement repérables et identifiables.

En effet, la détection de ceux-ci équivaut à vérifier l'égalité entre deux textes.

Pour effectuer naïvement cette recherche de façon automatique, on est obligé de procéder à une comparaison mot à mot des textes. Cette opération, étant beaucoup trop chronophage pour être intégrée dans des solutions à but commercial ou hébergées en ligne, comme c'est le cas des services anti-plagiat, des techniques alternatives ont dû être mises au point.

Les méthodes les plus répandues restent à l'heure actuelle celles à base de **n-grammes**, consistant à représenter les textes sous forme de séquences de n éléments pouvant être des lettres, des syllabes, des mots,des entités nommées, etc. afin de les comparer plus facilement par la suite sur une granularité plus fine (Manning et Schütze, 1999).

La campagne d'évaluation PAN a arrêté en 2014 sa dernière tâche pouvant être apparentée à la

détection du plagiat extrinsèque (les tâches officielles traitant ce sujet avaient déjà toutes été arrêtées en 2011). Depuis, cette campagne se focalise sur des tâches d'identification et de vérification d'auteurs. Lors de la dernière année de la tâche de détection du plagiat extrinsèque, Grman et Ravas(2011) remportèrent la compétition avec un système basé sur l'intersection de mots. La recherche de Barrón-Cedeño et Rosso (2009) avait prouvé qu'en prenant des n-grammes de mots (séquence de n mots se suivant dans un texte) de petites tailles, 2 ou 3 par exemple, les résultats sont bien meilleurs qu'en utilisant des longues séquences avec un n plus important. Grman et Ravas(2011) ont établi qu'à partir de 15 mots consécutifs communs, deux passages dans deux textes différents peuvent être considérés comme suffisamment suspects pour être relevés. Toutefois, les n-grammes les plus pertinents à utiliser lors d'une comparaison ne sont pas toujours des séquences de mots, comme en atteste le travail de Shrestha et Solorio(2013).

Des n-grammes de mots vides et d'entités nommées peuvent également être utilisés pour détecter des parties de textes similaires entre deux documents. Les mots vides sont des mots vides de sens n'apportant aucune valeur sémantique supplémentaire dans un texte, ce sont le plus souvent des articles, des prépositions, des pronoms etc.

Les entités nommées sont des expressions, des groupes de mots faisant référence à des entités remarquables au sein d'un texte, comme un nom d'entreprise, un lieu ou une date par exemple.

D'autres méthodes assez répandues sont les méthodes à base d'empreinte textuelle, créant une empreinte du document pour la comparer avec celle d'autres documents. La plupart de ces méthodes, à l'instar des travaux de Kent et Salim (2010a), utilisent également des n-grammes pour construire l'empreinte des documents. Les méthodes à base d'empreinte textuelle divisent la plupart du temps les documents en n-grammes, ainsi les empreintes de deux documents peuvent être comparées.

Certaines méthodes à base d'empreinte textuelle (Stein et Eissen, 2006, 2007a;Lyon et al.,2001) vont au-delà de la recherche de similitudes exactes et introduisent la notionde similarités proches pouvant

en théorie ainsi détecter les paraphrases. Plus tard, Sanchez-Perez et al. (2014) proposent lors de la dernière tâche d'alignement textuel de la PAN, une mesure de similarité entre phrases basée sur un modèle vectoriel pondéré avec des fréquences de termes permettant de conserver les mots vides tout en évitant d'incrémenter le taux de faux positifs. Ils introduisent un algorithme récursif pour étendre la correspondance des phrases en construisant des passages de longueur maximale. Pour cela si deux fragments similaires dans les deux textes sont contigus avec deux autres fragments également similaires dans les deux textes, on les concatène pour former un fragment similaire plus large. Le but étant de trouver les plus longues séquences communes entre les deux textes.

Plus récemment, avec l'arrivée des **réseaux de neurones**, de nouvelles approches ont vu le jour. Yin et Schütze (2015), par exemple, utilisent des Word Embeddings (Mikolov et al.,2013a) avec une pondération pour les mots jugés plus importants dans le cadre de l'identification d'une paraphrase. pour constituer des représentations vectorielles de phrases plus pertinentes à employer lors d'une tâche de détection de paraphrases.

2.6.2 La prévention du plagiat translingue

Il existe cinq familles d'approches de détection de plagiat translingue :

2.6.2.1 Modèles basés sur une analyse lexicale ou syntaxique :

La première famille est l'ensemble des modèles basés sur la comparaison lexicale ou syntaxique. L'avantage de ces méthodes est qu'elles ne nécessitent aucune traduction des textes à comparer. Elles sont entièrement basées sur une analyse lexicale ou syntaxique et demandent seulement, dans la majorité des cas, une étape de nettoyage ou pre-processing (suppression des espaces, suppression des mots vides de sens, suppression des caractères diacritiques, etc.), ce qui justifie un temps de traitement assez rapide (aussi rapide que dans les cas monolingues).

2.6.2.2 Modèles à base de dictionnaires et thésaurus

Cette famille de méthodes repose sur l'utilisation de bases de connaissances lexicales ou conceptuelles externes, comme des dictionnaires, des thésaurus ou des réseaux sémantiques.

2.6.2.3 Modèles basés sur les corpus parallèles.

On appelle corpus parallèle C, deux ensembles de documents D et D0 dans deux langues différentes (respectivement L et L 0), dans lesquels chaque document di du corpus D est la traduction de l'un des

documents d 0 i du corpus D0 (voir Figure 2.9). Il peut être alors intéressant d'aligner ces corpus, c'est-à-dire de faire correspondre chaque unité textuelle (documents, paragraphes, phrases ou groupes de mots) de l'un des corpus avec l'unité textuelle lui correspondant dans l'autre corpus pour disposer d'un jeu de données bilingues comparables plus fin ou spécifique. À titre d'exemple, citons les JRC-Acquis 15 (Steinberger et al., 2006), qui sont des comptes rendus, disponibles dans plus de 20 langues, de débats du parlement Européen.

2.6.2.4 Modèles basés sur les corpus comparables.

Un corpus comparable est basé sur le même principe qu'un corpus parallèle. Il s'agit de deux ensembles de textes dans deux langues différentes, dans lesquels chaque document de l'un des ensembles est relatif à un document du second ensemble.

Mais contrairement à un corpus parallèle, dans un corpus comparable, deux documents relatifs ne sont pas la traduction l'un de l'autre, ils traitent seulement du même sujet spécifique, c'est-à-dire que les deux documents partagent le même sujet dans le même registre tout en n'étant pas pour autant la traduction exacte l'un de l'autre.

Wikipédia est le corpus comparable le plus utilisé et le plus simple d'accès. En effet, deux articles équivalents dans deux langues différentes sont rarement les traductions exactes et parfaites l'un de

l'autre mais plutôt deux articles indépendants, écrits par deux auteurs différents, dans deux langues différentes, traitant du même sujet. Une présentation d'autres corpus comparables couramment utilisés dans la détection de similarités textuelles sémantiques translingues sera donnée dans la

2.6.2.5 Modèles à base de traduction suivie d'une analyse monolingue (Translation + Monolingual Analysis, T+MA)

Les méthodes présentées précédemment, utilisent des mécanismes et des ressources lexicales utilisés dans la traduction automatique, mais n'effectuent pas réellement de traduction à proprement parler. Les méthodes présentées dans cette section quant à elles, effectuent bel et bien une traduction automatique pour ramener les deux textes à comparer sous une forme textuelle de même langue afin de pouvoir les comparer avec des approches monolingues plus éprouvées dans la littérature.

2.7 Outils et logiciels pour la détection de plagiat

En terme de plagiat la technologie est devenu une arme à double tranchant. Elle a facilité accès à l'information et de faire le copier-coller et a la fois elle la rendue aussi beaucoup plus facile à détecter Grâce au programme de détection de plagiat en ligne. Certains de ces logiciel sont payants, tandis qu'il existe d'autre version qui sont gratuit Voici les plus célèbres avec Une brève description et leurs caractéristiques principales :

- ✓ **PlagScan**

Est un outil de détection de plagiat textuelle en ligne, il utilise un algorithme qui fournit un certain nombre de fonctionnalités robustes basé sur les dernières recherches en linguistique informatique PlagScan est capable de détecter la plupart des types de plagiat soit directement par copier-coller, soit par remplacement des mots .

Les principales caractéristiques de PlagScan sont :

• Equiper avec une locale base de données qui comprend des millions de documents tels que (papier, articles et devoirs)

• PlagScan a différentes politiques de prix pour les individus et les organisations et offre aux nouveaux utilisateurs pour une inscription obligatoire un test gratuit .

• Il fournit une vérification soit en ligne ou a partir de la base de données locale .

• Comparaison avec des plusieurs documents en parallèle .

• Prend en charge toutes les langues qui utilisent le codage international UTF-8 et toutes les langues avec des caractères latins ou arabes .

✓ **Turnitin**

Est un service de détection de plagiat fourni par iParadigms, Il se dispose d'une interface facile à utiliser en effet cet outil été accepté par 15 000 établissements et 30 millions d'étudiants, il peut être considéré comme l'un des meilleurs contrôleurs de plagiat pour les enseignants.

✓ **Quetext**

Est un outil de détection de plagiat gratuit basé sur des algorithmes de traitement du langage naturel et d'apprentissage automatique. Pour vérifier le plagiat avec cet outil, il suffit de copier-coller du document il dispose les propriétés suivant :

• Prend en charge plusieurs langues .

• Compare les documents aux sources internet .

• Il ne considère pas les expressions courantes comme plagiat .

• N'affiche pas le pourcentage d'originalité mais qu'il fournit la liste des sources .

• ne fournit pas rapport de détail . • Il offre que l version gratuit .

✓ **PlagTracker**

Est outil online de détection de plagiat textuelle créé par deux ukrainiens étudiants en informatique

[36] Les principales caractéristiques de PlagTracker sont :

• Il possède une vaste base de données de publications académiques .

• corrige les fautes grammaticales .

• fournit le pourcentage de plagiat avec Rapport téléchargeable .

• Il est disponible en versions gratuites et payantes .

• Support multilingue Exclusion de référence.

Chapitre 3 : SPECIFICITES DE LA LANGUE ARABE

3.1. Introduction

Par ses propriétés morphologiques et syntaxiques et sémantiques la langue arabe est considérée comme une langue difficile à maîtriser dans le domaine du traitement automatique de la langue. L'arabe doit sa formidable expansion à partir du 7eme siècle grâce à la propagation de l'islam et la diffusion du Coran. Les recherches pour le traitement automatique de l'arabe ont débuté vers les années 70. Les premiers travaux concernaient notamment les lexiques et la morphologie[Dou04]. Avec la diffusion de la langue arabe sur le Web et la disponibilité des moyens de manipulation de textes arabes, les travaux de recherche ont abordé des problématiques plus variées comme la syntaxe, la traduction automatique, l'indexation automatique des documents, la recherche d'information, etc.

3.2. Particularité de la langue arabe

En typologie des langues, on appelle *langue flexionnelle* une langue dans laquelle les lemmes[1] « mots » changent de forme (soit le nombre des noms, soit le temps verbal) selon les rapports grammaticaux qu'ils entretiennent avec les autres lemmes. Certains mots modifient donc leur forme; on dit d'eux qu'ils subissent le jeu de la flexion**.**

L'ensemble des formes différentes d'un même mot fléchi forme son paradigme. Selon cette définition, l'arabe se classe comme une langue à morphologie extrêmement riche[Kas05] :

Premièrement, le système dérivationnel se présente comme un jeu de construction basé sur le rôle sémantique de l'élément dérivé. L'association d'affixes à une racine donnée permet d'engendrer des

[1]Le lemme est l'entrée de dictionnaire, il s'agit donc de la forme classique et entièrement vocalisée à laquelle se rattache la forme du texte (classique ou non, vocalisée ou non)

mots avec des significations différentes, mais qui peuvent être classés comme mots de la famille du dérivant. Par exemple, du verbe «كسر »signifiant ‘casser’ sont dérivés, par doublement de la consonne /س/, le verbe "كسّر"casser en mille morceaux’, et, par ajout du préfixe /إن/, le verbe«انكسر » ‘se casser’ [Kas05].

Deuxièmement, le système flexionnel affiche un marquage diversifié. Par exemple, l’arabe possède trois cas nominaux : le nominatif, qui est le cas par défaut, l’accusatif pour les compléments verbaux et le génitif pour le dépendant d’une préposition [Kas05].

Troisièmement, l’arabe obéit à des contraintes communicatives, mais aussi à des contraintes syntaxiques. C’est pour cela qu’elle se classe comme une langue à ordre mixte : ni fixe, ni entièrement libre.

3.3. La grammaire arabe

La grammaire se présente traditionnellement comme la science des règlesde la langue arabe, (قواعد اللغة العربية). Elle comprend deux branches:

3.3.1 La morphologie

Qui se divise en :

3.3.1.1 Morphologie dérivationnelle

La morphologie dérivationnelle étudie la construction des mots et leur transformation selon le sens voulu ; autrement dit, la dérivation morphologique est décrite sur une base morphosémantique : d’une même racine, se dérivent des mots différents, (الصيغ).

3.3.1.2 Morphologie flexionnelle

La morphologie flexionnelle comprenant d’une part (الإعراب), concerné par le marquage casuel pour le nom et l’adjectif ou la conjugaison du verbe. Et d’autre part, (البناء), essentiellement des règles morpho-phonologiques, on par le ainsi de formes (الممنوع من الصرف)pour désigner des mots dont la terminaison ne fait pas apparaître la marque de cas à la prononciation,

Comme toute langue sémitique[2], la morphologie de l'arabe fonctionne sur le principe de radicaux verbaux, le plus souvent à trois consonnes. Des voyelles s'y ajoutent pour former des dérivés et des formes flexionnelles.

3.3.2. La syntaxe

Qui étudie l'élaboration correcte des phrases. Elle assure donc la grammaticalité de la phrase en examinant :

- La position des mots les uns par rapport aux autres (موقع المفردات), déterminant ainsi l'ordre des mots.
- L'état de la terminaison des mots de la phrase (أحوال أواخر الكلمات). Par conséquent, la fonction syntaxique de l'unité lexicale est identifiée en se basant sur la morphophonologie.

Cette approche peut se résumer en deux points :

Premièrement, les grammairiens arabes ont présenté la langue sous un angle prescriptif et non pas descriptif, dans un souci de donner des règles à appliquer pour écrire et parler correctement. Un livre de grammaire traditionnel ne décritpas la langue de communication mais il montre commentl'employer [Kas05]. C'est pourquoi l'on note beaucoup de confusion entre d'une part grammaticalité et d'autre part ambiguïté et acceptabilité.

Deuxièmement la présentation de la syntaxe de l'arabe n'est pas fondée vraiment sur le lien fonctionnel entre les mots de la phrase. Elle s'appuie sur d'autres critères :

- *La précédence linéaire :* l'on distingue ainsi la phrase nominale (celle qui commence par un nom) de la phrase verbale (celle qui commence par un verbe).
- *Le lien logico-sémantique :* aussi l'on différencie la fonction (الفاعل)(celui qui fait l'action) de la fonction (نائب الفاعل)(celle qui le remplace) alors que dans les deux cas il est question du sujet d'un verbe, à la voix active dans le 1ère cas et à la voix passive dans le second cas.
- *La dépendance morphologique :* la grammaire traditionnelle est largement orientée par la dépendance morphologique. Aussi sont groupés indistinctement sous la fonction de (مفعول به)

[2] Les langues sémitiques font partie de la famille des langues afro-asiatiques, et sont parlées en Afrique septentrionale et saharienne ainsi qu'au Proche-Orient et au Moyen-Orient. Elles se caractérisent entre autres par la prédominance de racines trilitères (composées de trois consonnes)

(complément du verbe) plusieurs compléments nominaux et prédicatifs du verbe uniquement parce qu'ils portent tous la marque de l'accusatif.

La grammaire est élaborée sous un angle descriptif, et la syntaxe de l'arabe est décrite en se basant sur le lien fonctionnel entre les mots de phrase. Les dépendances sémantique et morphologique, bien qu'elles soient prises en considération, ne constituent pas la base sur laquelle est fondée l'identification des fonctions syntaxiques.

3.4. Catégorisation des mots arabes

Nous recensons en arabe huit classes de mots : quatre classes majeures (verbe, nom, adverbe et adjectif), trois classes de mots fonctionnels (référentiel, translatif et connecteur). Chaque classe se distingue par ses propriétés distributionnelles. Certaines sont sujettes à des répartitions plus fines. De façon hiérarchique, les classes se présentent comme suit :

3.4.1. Les quatre classes majeures

- ***Le verbe*** **:** le verbe à la forme finie est la tête syntaxique de la proposition déclarative arabe. Il correspond à un prédicat sémantique et possède une structure actancielle.
- ***Le nom*** **:** il correspond à une entité abstraite ou concrète, et figure, essentiellement, comme un dépendant syntaxique du verbe. Les deux sous-classes principales du nom sont le nom propre et le nom commun.
- ***L'adjectif*** **:** il modifie le nom et remplit les fonctions de codifieur (épithète), de coprédicat ou d'attribut.
- ***L'adverbe*** **:** L'adverbe en tant que modifieur de verbe, remplit principalement la fonction d'un circonstant

3.4.2. Les classes des mots fonctionnels

3.4.2.1 Les référentiels

Sont des mots qui jouent un rôle de pointeur. On distingue deux types des référentiels : les pronoms et les démonstratifs.

a. ***Les pronoms (les pointeurs contextuels)*** **: c**e sont des pointeurs à un constituant de la proposition, essentiellement un constituant nominal. Ils se divisent en pronoms personnels et pronoms interrogatifs.

- ***Les pronoms personnels* :** Les pronoms personnels se divisent, selon les fonctions syntaxiques qu'ils exercent, en pronoms proleptiques et pronoms compléments d'objet. Le pronom sujet est systématiquement omis.

- ***Les pronoms proleptiques* :** Un pronom proleptique ou emphatique s'accorde en genre et en nombre avec son référent. Il figure seulement dans les constructions de mise en relief comme la construction proleptique ou la mise en exclusion
- ***Les pronoms compléments d'objet* :** Un pronom complément d'objet est attaché comme enclitique. Il remplit la fonction d'un complément du verbe, d'une préposition ou du nom.

- ***Les pronoms interrogatifs* :** Le référentiel interrogatif occupe la position initiale de la phrase. Il peut remplir la fonction d'un valent du verbe :

b. ***Les démonstratifs :***Les démonstratifs constituent le groupement des pointeurs sur des objets dans un contexte de communication. En arabe, ils sont étroitement liés au repérage spatio-temporel. L'arabe se base sur la nature du repère spatial, et il distingue deux types de pointeurs : les pointeurs dans l'espace physique (***les démonstratifs I***) et les pointeurs dans l'espace mental (***les démonstratifs II***). Les démonstratifs s'accordent en personne, genre et nombre avec leur référent. Les démonstratifs II exercent des fonctions nominales ou adjectivales tandis que les démonstratifs I, comme les pronoms, exercent seulement les fonctions d'un nom.

- ***Les démonstratifs I***

Les démonstratifs I réfèrent au groupe connu dans la grammaire arabe sous l'appellation (أسماء الإشارة), littéralement « les noms de pointeur » ou « les noms de désignation ». Ils se divisent, d'après l'espace séparant le locuteur de l'objet désigné, en pointeurs à une entité proche, pointeurs à une entité intermédiaire et pointeurs à une entité lointaine.

- ***Les démonstratifs II***

Les démonstratifs réfèrent principalement à une classe connue dans la tradition arabe sous l'appellation *les noms liés* (أسماءالموصول).

3.4.2.2 Les translatifs

La classe des translatifs comprend les lexèmes fonctionnels permettant à un mot d'une classe donnée

d'occuper des positions, ou plus précisément de remplir des fonctions réservées à une autre classe.

Parmi les translatifs, citons les prépositions et les conjonctions de subordination.

- ***Les translatifs de nom en adverbe (des prépositions)***

Une préposition gouverne un nom au génitif et lui permet d'exercer la fonction d'un adverbe.

- ***Les translatifs de nom en construction exprimant l'évidentialité (les prépositions d'évidentialité)***

Une préposition d'évidentialité gouverne un nom à l'accusatif et lui permet de remplir la fonction d'une prolepse

- ***Les translatifs de verbe en adverbe ou en nom***

Les translatifs analytiques de verbe sont communément appelés des conjonctions de subordination ou complémentaire (dans la grammaire générative).

3.4.2.3 Les connecteurs

C'est le groupe des conjonctions de coordination. Parmi les connecteurs, citons : le conjonctif (و) 'et', le disjonctif interrogatif(أم) 'ou exclusif', le disjonctif (أو) 'ou', le successif (ثم) 'ensuite' et la conjonction exprimant l'appartenance (حتى) 'y compris, même'.

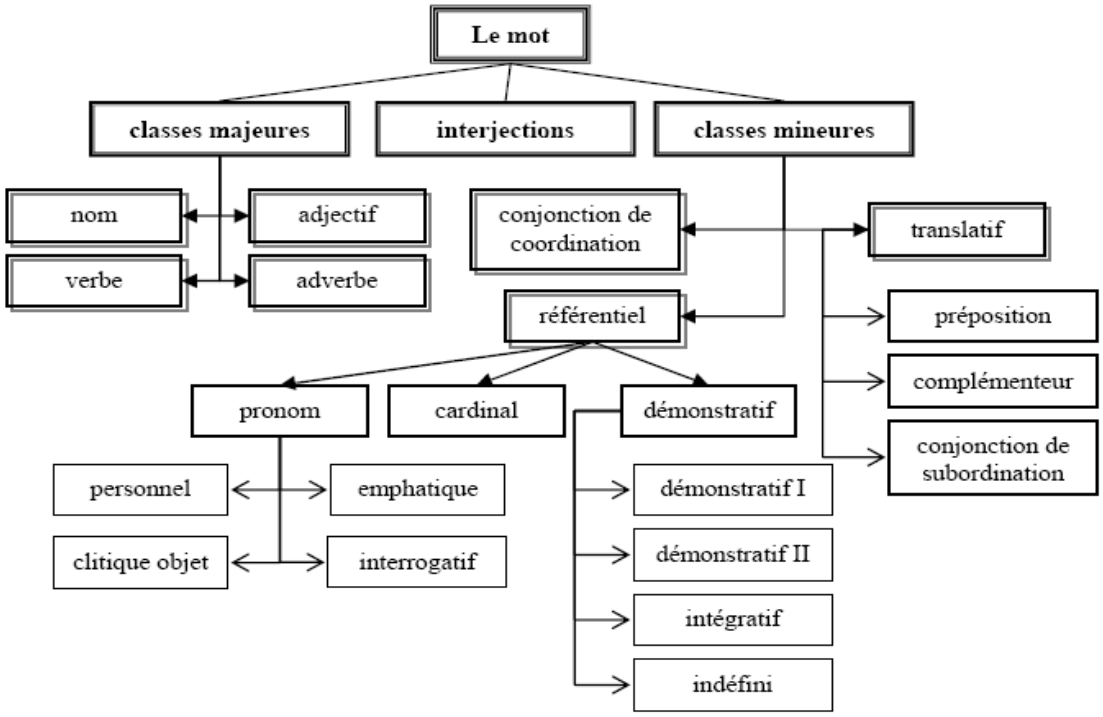


Figure 3.1 : Classification des mots en arabe**[Kas05]**

3.5. La sémantique dans la langue arabe

La sémantique occupe une position importante dans le traitement de la langue arabe. Il n'est pas commode de réaliser des traitements profonds des textes arabes sans informations suffisantes sur la sémantique des termes et les relations sémantiques entre les mots constitutifs de ces textes.

Donc, généralement les chercheurs dans le traitement automatique de la langue arabe ne doivent pas oublier les questions de Synonymie et d'Antonymie et d'autres relations sémantiques, en raison de leur valeur ajoutée qui consiste à enlever l'ambiguïté sémantique et structurelle, et donc formuler l'architecture des applications informatiques selon une forte structure découlant de l'acquisition des concepts linguistiques algorithmiques, et pourrait donc résoudre les problèmes de l'analyse automatique des langues en général.

3.5.1. Les phénomènes sémantiques

Les phénomènes de synonymie et de l'antonymie dans la langue arabe sont considérés parmi les phénomènes linguistiques les plus intéressants sur lesquels on a animé beaucoup de débats et de discussions entre les linguistes, les hommes de lettres ، les anciens et nouveaux chercheurs, la majorité les considère comme l'une des spécificités de la langue arabe, et l'un des aspects de son génie. Cette section envisage de démontrer l'importance de la synonymie et de l'antonymie, en les considérants comme étant la clé pour résoudre de nombreux problèmes au niveau de l'analyse linguistique automatique, citons:

1. Mettre les caractéristiques et les traits des mots et des champs sémantiques (cf. section 5.3.1.1).
2. L'analyse automatique des textes.
3. La compréhension des textes.
4. la résolution des problèmes de la désambiguïsation linguistique.
5. la traduction à partir de la langue source vers langue cible

3.5.1.1. La synonymie

La synonymie est la relation sémantique qu'il existe entre deux items lexicaux qui diffèrent par

leur forme mais expriment le même sens ou un sens très proche. Nous devons savoir que les synonymes ne sont pas complètement des mots identiques, sauf s'ils ont le même opposé, la même utilisation du contexte, et les autres relations [Meh06].Ce qui n'est pas généralement le cas. Donc on ne peut pas dire que la synonymie est la similitude complète et absolue, mais c'est la similitude de la plupart des traits sémantiques parce que même si deux mots ont le même usage, il y a quelques différences sémantiques entre eux et si les mots sont identiques de tous les cotés, il y a une raison derrière l'existence de deux mots.

1- Que l'un est plus utilisé que l'autre (حسام / سيف)
2- Que l'un est plus englobant que l'autre (انتحب / بكى)
3- Le sens dans l'un est plus puissant que dans l'autre (تعب / أنهك)
4- L'un est plus existant que l'autre (أتون / موقد)
5- L'un est plus acceptable que l'autre (فؤاد / قلب)
6- L'un est plus originaire de la langue que l'autre)(آلية / ميكانيزم
7- L'un est plus spécialisé au sens que l'autre (استقلال / حكم ذاتي)
8- L'un est plus fréquent ou plus développé que l'autre (أم / والدة)

Donc, on doit noter que la synonymie doit être incluse dans des niveaux selon les traits et les relations communes :

1- Synonymie complète ou totale
2- Quasi-synonymie (presque synonymie)
3- Synonymie limitée ou restreinte

Toutefois, quelque soient les différences des traits sémantiques, ils peuvent toujours être regroupés dans un même champ sémantique.

3.5.1.1.1 Principales causes d'occurrence de la synonymie dans la langue arabe

Les principales causes d'occurrence de la synonymie dans la langue arabe sont :

1- Le copiage à partir des autres langues et dialectes
   - Multiplicité des tribus arabes et leurs usages de différents termes pour désigner la même chose.
   - L'empreint à partir des autres langues sémiotiques et autres (langue Arménienne, La langue de l'ancienne Yémen, Langue persane….)

2- Les dictionnaires

- Les confectionneurs de dictionnaires ont recueilli les matières à partir des dialectes des différentes tribus. ces matières se diffèrent dans les aspects et dans la forme des mots selon chaque tribu.
- Les confectionneurs des dictionnaires ont recueilli plusieurs noms pour la même chose sans prendre en considération le coté historique.

Exemple: les noms des mois pendant le pré-islam ont changé après l'islam mais ce n'étaient pas forcément des synonymes.

- L'entrée non contrôlée de nombreux mots, et des mots qui ne sont pas certainement arabes exemple : /تاكسي)(سيارة ,(تليفون/هاتف)
- L'habitude d'utiliser un adjectif jusqu'à ce qui 'il remplace le nom du fait de son usage courant : Exemple : عباس remplace le mot أسد

الفصل remplace le mot سيف

3- L'évolution phonétique et sémantique du mot des synonymes, qui autrefois étaient un peu différents, sont devenus identiques.

4- La généralisation de ce qui était spécial

Exemple : « الدفن » L'inhumation des morts, il est devenu pour des autres comme دفن سره

5- Evolution de métaphore

6- Disparition de la différence entre deux mots de force de l'usage répété pour qu'à la fin ils deviennent synonymes. Exemple : (الريبة / الاضطراب)

3.5.1.1.2 Les niveaux de la synonymie

Il y a deux niveaux de synonymes dans la langue arabe sont :

- *Niveau du mot*
  a. Synonymie entre verbes : exemple (أعور / أختر), (أعدم/ أملق)
  b. Synonymie entre noms : exemple (حبور / فرح/سرور/غبطة)
- *Niveau des phrases*
  a. Phrase verbale exemple : « يقرب البعيد » et « يظهر الخافي et « يصيب المفصل » ont le même sens.
  b. Semi- phrase exemple : « إليه منقض الأمر » a le même sens avec (مآله / مصيره)

3.5.1.1.3 Les types des synonymes

Dans la langue arabe il y a plusieurs types de synonymes, citons :

- Synonyme de pose : causé par la 1ere cause de synonyme qui est déjà cité

Exemple : (الأسد / الليث), (الجمال / العير), (عطش / ظمأ)

- Synonyme de dérivation *Exemple* : (المسمع / الأذن), (المحيى / الوجه)
- Synonyme de désordre de lettres : *Exemple* :(جذب / جبذ), (حمد / مدح)
- Synonyme de remplacement d'une lettre : *Exemple* : (نقب /ثقب), (لدغ / لذع)
- Synonyme de métaphore : *Exemple*(الأسل / الرماح)
- Synonyme de d'euphémisme : *Exemple*(بخيل / جعد الكف)
- Synonyme de lexicalisé : *Exemple* : كعك d'origine persan
- Synonyme de compatibilité de prononciation (lettres) : *Exemple*(نعق/ نهق)
- Synonyme de remplacement par une lettre semblable : *Exemple*(نفع / نجع)
- Synonyme de suite : *Exemple*(جائع / نائع)
- Synonyme de suppression : *Exemple*(شوطة / أنشوطة)
- Synonyme de description : *Exemple* (السيف / القاطع)

Chaque formule a un sens bien défini qui existait effectivement, puis ces formules sont disparues et avec le temps les raisons de métaphore sont aussi disparues, ces mots sont devenus des synonymes.

3.5.1.1.4 L'importance de la synonymie

Les synonymes visent à atteindre les avantages suivants :

- Offrir à l'utilisateur un lexique très riche et plusieurs termes pour désigner le même sens, donc l'utilisateur a l'occasion de sélectionner ce qui est adéquat avec le contexte.
- Stimuler la jouissance et réduire l'ennui du lecteur, car la diversité offre à l'auteur la possibilité de choisir ses termes loin de toute ambiguïté sémantique et donc il fixe le sens voulu.

3.5.1.2. L'antonymie

L'antonymie est la relation sémantique qui existe entre deux items lexicaux dont les sens s'opposent. Elle est capable d'enrichir les dictionnaires très fortement, l'opposé du mot éclaire le sens de son opposé malgré qu'il n'est généralement pas complet.

3.5.1.2.1 Cause de l'antonymie

La relation entre deux mots

- *RessemblanceExemple* : « بشرة »pour la peau de l'être humain et pour la plante
- *OppositionExemple*: « عسس » pour l'avènement de la nuit (إقبال الليل) et pour L'avènement du matin (إدبار الليل)
- *Différents sens de même champ sémantique* : *Exemple* « السرحان » pour le lion et pour le loup
- *Classe de parole* : *Exemple* « أجم » pour le verbe (اقترب) et pour le nom « كبش بلا قرون »

- *La brièveté et la métaphore* :*Exemple* « مكتب » pour le meuble « ما يكتب عليه », pour la salle du bureau « حجرة المكتب » et pour l'équipe de travail « مكتب الإدارة »

3.5.1.2.2 Les aspects de l'antonymie

Les aspects de cette notion sont :

*3.5.1.2.2.1 L'antonymie parfaite :*la différence entre les deux mots est totale, il n'y a pas d'aspects ou des degrés de rapprochement entre les deux :

(رجل،امرأة), (أعزب،متزوج), (صح،خطأ), (ميت،حي), (ذكر،أنثى)

3.5.1.2.2.2 L'antonymie partielle : entre les deux mots existent des aspects de rapprochement sémantique, exemple :

(سهل،صعب) entre les deux on a des nuances de facilitation de difficulté

(قوي،ضعيف), (كبير،صغير), (بعيد،قريب), (بارد،حار)

3.5.1.2.2.3 L'antonymie des contraires : entre deux mots accompagnes exemple :

(مهزوم،فائز)، (مرءوس،رئيس)(مولود،والدة)، (تعلّم،علّم)، (أخذ،أعطى)، (اشترى،باع)، (زوجة، زوج)، (ابن،أب)

3.5.1.2.2.4 L'antonymie de l'extension : quand les deux mots appartiennent au même axe de direction

Exemple : (فوق ، تحت) ، و (شرق وغرب) ، (شمال وجنوب)

2.5.1.2.3 Le rôle de l'antonymie dans l'enrichissement du lexique

L'antonymie est un phénomène linguistique intéressant dans l'éclaircissement du sens, tel que l'opposé du mot éclaire son sens, malgré que l'antonymie n'est généralement pas complète que dans des rares cas, mais il l'est dans certains traits.

Exemple : « ميت » son opposé est vivant « حي » donc l'opposé est désigné dans un seul trait qui est le trait de vie « الحياة », mais pour le mot « ميت » il y a plusieurs traits comme « الحياة », « الجنس », « النوع »….

L'antonymie est l'un des types de la synonymie négative (les synonymes jouent un rôle de l'aspect positif pour le champ sémantique, alors que les antonymes jouent le rôle de l'aspect négatif). Et de ce fait l'antonymie montre les aspects sémantiques, d'un coté et enrichir le lexique, d'un autre coté, de

surcroît les liaisons sémantiques entre les champs [Meh06].

3.6. Conclusion

La multiplicité des sens des mots est l'une des facteurs de développement de la langue arabe, et si on ne précise pas exactement les sens, on ne pourra, donc jamais, préciser ses relations avec les autres mots comme les synonymes, les antonymes. La majorité des linguistes modernes ont traité ces manifestations (homonymie, synonymie, antonymie) dans le cadre des relations sémantiques où ce qu'on appelle la théorie des champs sémantiques qui est en relation fondamentale avec la théorie sémantique de structuralisme.

Le travail dans le cadre du traitement automatique du langage naturel nécessite de s'appui sur le cadre théorique linguistique efficace et conceptuel qui comporte des concepts et de dynamique procédurale capable de décrire les manifestations linguistiques automatiques programmés dans les cerveaux humains selon des algorithmes.

Chapitre 4 : Implémentation de notre system

4.1 Introduction

- Dans les travaux de Mikolov et al. (2013c), les approches de Collobert & Weston (2008), Turian et al. (2010), Mnih & Hinton (2009) et Mikolov et al. (2013c) sont évaluées et comparées.

Ils montrent ainsi que CBOW et Skip-gram sont beaucoup plus efficaces en termes de rapidité et de précision.

C'est d'ailleurs pour cette raison que j'ai adopté dans mon travail le modèle SKIP-GRAM.

Par conséquent, dans le but de mesurer la similarité sémantique entre les mots arabe, j'ai utilisé la similarité cosinus pour construire mes données de l'entrainement.

mon modèle a été appris a partir de deux fichiers texte non volumineux afin que le programme puisse s'exécuter on peut également exécuter le programme sur différentes sources comme le Wikipedia Arabe sauf qu'il exige un grand débit internet.

Les performances du modèle SKIP-GRAM dépendent essentiellement de la tache et de la bonne définitions des paramétrés d'apprentissage, et aussi pour cette raison que j'ai utilisé différentes paramètres.

Notre system consiste à mesurer la similarité sémantique entre deux documents on utilisant le réseau sémantique par l'utilisation de Word Embeddings sur un cas de détection de plagiat programmé avec le langage python .

Les étapes suivies par mon system sont :

4.2 Echantillons de teste

Afin de mesurer efficacement les performances de notre système, pour un simple teste j'ai utilisé deux

Fichiers.

arabe-01.txt

فيروس كورونا أو **جائحة** كوفيد 19 . والمعروفة أيضا باسم **جائحة** فيروس كورونا . هي **جائحةٌ** عالميةٌ مستمرةً

حاليًا . سببها فيروس كورونا 2 المرتبط بالمتلازمة التنفسية الحادة الشديدة سارس كوف 2 . تفشى المرض للمرة الأولى في مدينة ووهان الصينية . في أوائل شهر ديسمبر عام 2019 . أعلنت منظمة الصحة العالمية رسميا . في 30 يناير أن تفشي .الفيروس يشكل حالة طوارئ صحية عامة . تبعث على القلق الدولي

arabe-02.txt

فيروس كورونا هو **جائحة** كوفيد 19 . اسمه أيضا فيروس كورونا . هذه **جائحةٌ** عالميةٌ موجودة

حاليًا . سببها فيروس كورونا 2 المرتبط بالمتلازمة التنفسية الحادة الشديدة سارس كوف 2 . المرة الأولى كانت في مدينة ووهان الصينية . في شهر ديسمبر سنة 2019 ميلادي . أعلنت منظمة الصحة العالمية رسميا . في 30 يناير أن انتشار الفيروس يشكل حالة خطيرة صحية عالمية . تزيد في القلق الدولي

4.3 Prétraitement

Lisons nos deux document avec le code :

```
filename                    =                   'D:\Data/arab_01.txt'
file            =           open(filename,'r',  encoding='utf-8-sig')
d01                         =                   file.read()
file.close()
```

```
d01 :
```

فيروس كورونا أو جائحة كوفيد 19 . والمعروفة أيضا باسم جائحة فيروس كورونا . هي جائحةٌ عالميةٌ مستمرةً
حاليًا . سببها فيروس كورونا 2 المرتبط بالمتلازمة التنفسية الحادة الشديدة سارس كوف 2 . تفشى المرض للمرة الأولى في مدينة ووهان الصينية . في أوائل شهر ديسمبر عام 2019 . أعلنت منظمة الصحة العالمية رسميا . في 30 يناير أن تفشي .الفيروس يشكل حالة طوارئ صحية عامة . تبعث على القلق الدولي

```
d02 :
```

فيروس كورونا هو جائحة كوفيد 19 . اسمه أيضا فيروس كورونا . هذه جائحةٌ عالميةٌ موجودة
حاليًا . سببها فيروس كورونا 2 المرتبط بالمتلازمة التنفسية الحادة الشديدة سارس كوف 2 . المرة الأولى كانت في مدينة ووهان الصينية . في شهر ديسمبر سنة 2019 ميلادي . أعلنت منظمة الصحة العالمية رسميا . في 30 يناير أن انتشار الفيروس يشكل حالة خطيرة صحية عالمية . تزيد في القلق الدولي

Afin de normaliser les phrases pour la sémantique, un ensemble de prétraitement est effectué sur

l'ensemble de données. Les deux documents ont été exécutés par les étapes suivantes:

- Supprimer les signes de ponctuation, les signes diacritiques et les stopWords.
- Normalisé إ آ أ par ا et ة par ه
- Remplacer le caractère ى suivi par ء par le caractère ئ .

- Remplacer les verbes les noms et les synonymes :

A partir des deux documents nous allons construire deux listes d_01 et d_02 dans lesquelles :

Les verbes sont conjuguer au présent.

- ✓ Exemple 1 : يشكل سيشكل شكل seront remplacer par يشكل .

Les noms seront remplacer par l'original du mot.

- ✓ Exemple 2 : الصينيين الصينية الصين seront remplacer par الصين

Et même travail sera effectuer sur des mot avec des synonymes .

- ✓ Exemple 3 : القلق الارتباك seront remplacer القلق

Pour réaliser cette tache j'ai utilisé trois **fichiers CSV** (verb.csv , noms.csv et synonyme.csv)

4.4 Construire le corpus le dictionnaire et fragmenter les documents en phrases

A partir du premier document nous allons construire la première table d1 qui contient tout les mots

normalisés du document_1 :

d1 = ['فيروس', 'كورونا', 'جائحة', 'كوفيد', 'والمعروفة', 'أيضا', 'باسم', 'عالمية', 'مستمرة', 'حاليا', 'سببها', 'المرتبط', 'بالمتلازمة', 'التنفسية', ' الحادة', ' الشديدة', ' سارس', ' كوف', ' تفشى', ' المرض', ' للمرة', ' الأولى', ' مدينة', ' ووهان', ' الصينية', ' أوائل', ' شهر', 'ديسمبر', 'عام', 'أعلنت', 'منظمة', 'الصحة', 'العالمية', 'رسميا', 'يناير', 'تفشي', 'الفيروس', 'يشكل', 'حالة', 'طوارئ', 'صحية', 'عامة', "تبعث", 'القلق', 'الدولي']

A partir du deuxième document nous allons construire la deuxième table d2 qui contient tout les mots

normalisés du document_2 :

d2 = ['فيروس', 'كورونا', 'جائحة', 'كوفيد', 'اسمه', 'أيضا', 'عالمية', 'موجودة', 'حاليا', 'سببها', 'المرتبط', 'بالمتلازمة', 'التنفسية', 'الحادة', 'الشديدة', 'سارس', 'كوف', 'المرة', 'الأولى', 'مدينة', 'ووهان', 'الصينية', 'شهر', 'ديسمبر', 'سنة', 'ميلادي', 'أعلنت', 'منظمة', "الصحة", 'العالمية', 'رسميا', 'يناير', 'انتشار', 'الفيروس', 'يشكل', 'حالة', 'خطيرة', 'صحية', 'تزيد', 'القلق', 'الدولي']

A partir de ces deux tables nous allons construire deux autres tables la première est notre dictionnaire

Dictionnaire = {'الأولى', 'حالة', 'الصحة', 'الفيروس', 'مستمرة', 'المرة', 'الدولي', 'عام', 'تفشى', 'سنة', 'باسم', 'التنفسية', 'شهر', 'الحادة', 'عامة', 'سببها', 'انتشار', 'الشديدة', 'أوائل', 'كوف', 'بالمتلازمة', 'القلق', 'يشكل', 'للمرة', 'سارس', 'أعلنت', 'كورونا', 'اسمه', 'حاليا', 'الصينية', 'رسميا', 'والمعروفة', 'فيروس', 'أيضا', 'ميلادي', 'موجودة', 'تزيد', 'منظمة', 'طوارئ', 'ديسمبر', 'العالمية', 'مدينة', "خطيرة", 'تفشي', 'صحية', 'جائحة', 'ووهان', 'المرتبط', 'عالمية', 'المرض', 'كوفيد', 'تبعث', 'يناير'}

Tandis que la deuxième est **MotInTwo** qui contient les mots en communs entre d1 et d2

mot in two : ['أعلنت', 'أيضا', 'الأولى', 'التنفسية', 'الحادة', 'الدولي', 'الشديدة', 'الصحة', 'الصينية', 'العالمية', 'الفيروس', 'القلق', 'المرتبط', 'بالمتلازمة', 'جائحة', 'حالة', 'حاليا', 'ديسمبر', 'رسميا', 'سارس', 'سببها', 'شهر', 'صحية', 'عالمية', 'فيروس', 'كورونا', "كوف", 'كوفيد', 'مدينة', 'منظمة', 'ووهان', 'يشكل', 'يناير']

L'étape suivante et de numéroter les mots de d1 et d2 et aussi le dictionnaire (pour chacune d'eux

créer une autre table dans laquelle attribuer un indice a chaque mot)

Exemple pour le dictionnaire :

```
Words_dict_2int = {}
int2Words_dict = {}
for i,Word in enumerate(dictionnaire):
Words_dict_2int[Word] = i
int2Words_dict[i] = Word
print('Words_dict_2int' , Words_dict_2int)
```

Ce qui me donne le résultat pour les trois table respectivement :

Words_doc1_2int ={'الأولى': 0, 'حالة': 1, 'الصحة': 2, 'الفيروس': 3, 'مستمرة': 4, 'الدولي': 5, 'عام': 6, 'تفشى': 7, 'باسم': 8, 'التنفسية': 9, 'شهر': 10, 'الحادة': 11, 'عامة': 12, 'سببها': 13, 'الشديدة': 14, 'أوائل': 15, 'كوف': 16, 'بالمتلازمة': 17, 'القلق': 18, 'يشكل': 19, 'للمرة': 20, 'سارس': 21, 'أعلنت': 22, 'كورونا': 23, 'حاليا': 24, 'الصينية': 25, 'رسميا': 26, 'والمعروفة': 27, 'فيروس': 28, 'أيضا': 29, 'منظمة': 30, 'طوارئ': 31, 'ديسمبر': 32, 'العالمية': 33, 'مدينة': 34, 'تفشي': 35, 'صحية': 36, 'جائحة': 37, 'ووهان': 38, 'المرتبط': 39, 'عالمية': 40, 'المرض': 41, 'كوفيد': 42, 'تبعث': 43, 'يناير': 44}

Words_doc2_2int ={'الأولى': 0, 'حالة': 1, 'الصحة': 2, 'الفيروس': 3, 'المرة': 4, 'الدولي': 5, 'سنة': 6, 'التنفسية': 7, 'شهر': 8, 'الحادة': 9, 'سببها': 10, 'انتشار': 11, 'الشديدة': 12, 'كوف': 13, 'بالمتلازمة': 14, 'القلق': 15, 'يشكل': 16, 'سارس': 17, 'أعلنت': 18, 'كورونا': 19, 'اسمه': 20, 'حاليا': 21, 'الصينية': 22, 'رسميا': 23, 'فيروس': 24, 'أيضا': 25, 'ميلادي': 26, 'موجودة': 27, 'تزيد': 28, 'منظمة': 29, 'ديسمبر': 30, 'العالمية': 31, 'مدينة': 32, 'خطيرة': 33, 'صحية': 34, 'جائحة': 35, 'ووهان': 36, 'المرتبط': 37, 'عالمية': 38, 'كوفيد': 39, 'يناير': 40}

Words_dict_2int ={'الأولى': 0, 'حالة': 1, 'الصحة': 2, 'الفيروس': 3, 'مستمرة': 4, 'المرة': 5, 'الدولي': 6, 'عام': 7, 'تفشى': 8, 'سنة': 9, 'باسم': 10, 'التنفسية': 11, 'شهر': 12, 'الحادة': 13, 'عامة': 14, 'سببها': 15, 'انتشار': 16, 'الشديدة': 17, 'أوائل': 18, 'كوف': 19, 'بالمتلازمة': 20, 'القلق': 21, 'يشكل': 22, 'للمرة': 23, 'سارس': 24, 'أعلنت': 25, 'كورونا': 26, 'اسمه': 27, 'حاليا': 28, 'الصينية': 29, 'رسميا': 30, 'والمعروفة': 31, 'فيروس': 32, 'أيضا': 33, 'ميلادي': 34, 'موجودة': 35, 'تزيد': 36, 'منظمة': 37, 'طوارئ': 38, 'ديسمبر': 39, 'عالمية': 40, 'مدينة': 41, 'خطيرة': 42, 'تفشي': 43, 'صحية': 44, 'جائحة': 45, 'ووهان': 46, 'المرتبط': 47, 'عالمية': 48, 'المرض': 49, 'كوفيد': 50, 'تبعث': 51, 'يناير': 52}

L'objectif de cette numérotation est de la servir pour la codification **one-hot (0,1) pour une**

présentation WordToVec de nos contextes. Retournant maintenant à nos documents et fragmenter

chacun en phrases :

sentences1 = [['فيروس', 'كورونا', 'أو', 'جائحة', 'كوفيد', '19'], ['والمعروفة', 'أيضا', 'باسم', 'جائحة', 'فيروس', 'كورونا'], ['هي', 'جائحةٌ', 'عالميةٌ', 'مستمرةً', 'حاليًا'], ['سببها', 'فيروس', 'كورونا', '2', 'المرتبط', 'بالمتلازمة', 'التنفسية', 'الحادة', 'الشديدة', 'سارس', 'كوف', '2'], ['تفشى',

'المرض', 'للمرة', 'الأولى', 'في', 'مدينة', 'ووهان', 'الصينية'], ['في', 'أوائل', 'شهر', 'ديسمبر', 'عام', '2019'], ['أعلنت', 'منظمة', 'الصحة', 'العالمية', 'رسميا'], ['في', '30', 'يناير', 'أن', 'تفشي'], ['الفيروس', 'يشكل', 'حالة', 'طوارئ', 'صحية', 'عامة'], ['تبعث', 'على', 'القلق', 'الدولي

sentences2 = [['هذه', 'جائحةٌ', 'اسمه', 'أيضا', 'فيروس', 'كورونا'], ['هو', 'جائحة', 'كوفيد', '19'], ['سببها', 'فيروس', 'كورونا', '2', 'المرتبط', 'بالمتلازمة', 'التنفسية', 'الحادة', 'الشديدة', 'سارس', 'كوف', '2'], ['المرة', 'الأولى', 'كانت', 'في', 'مدينة', 'ووهان', 'الصينية'], ['في', 'شهر', 'ديسمبر', 'سنة', '2019', 'ميلادي'], ['أعلنت', 'منظمة', 'الصحة', 'العالمية', 'رسميا'], ['في', '30', 'يناير', 'أن', 'انتشار', 'الفيروس', 'يشكل', 'حالة', 'خطيرة', 'صحية', 'عالمية'], ['تزيد', 'في', 'القلق', 'الدولي

4.5 Utilisation de Word Embeddings :

J'ai utilisé le model **SKIP-GRAM** de **Word Embeddings** l'idée est de chercher le contexte de chaque mot en commun alors pour chaque mots de la table **mots_in_two** je cherche les mots voisins de ce dernier dans la phrase ou il se trouve dans sentence_1 pour le premier document et dans sentence_2 pour le deuxième.

On choisissant une fenêtre de contexte **windows_size = 2** ce travail est exécuté sur les deux documents.

- ✓ Exemple pour document_1

```
data1 = []
WINDOW_SIZE = 2
for sentence in sentences1   :
   for Word_index, Word in enumerate(sentence):
      for nb_Word in sentence[max(Word_index - WINDOW_SIZE, 0) : min(Word_index + WINDOW_SIZE, len(sentence)) + 1] :
         if nb_Word != Word and estun_mot(nb_Word) and estun_mot(Word) :
data1.append([Word, nb_Word])
```

ce qui me donne le résultat suivant

- **Data_1 :** [['فيروس', 'كورونا'], ['كورونا', 'فيروس'], ['كورونا', 'جائحة'], ['جائحة', 'كورونا'], ['جائحة', 'كوفيد'], ['كوفيد', 'جائحة'], ['والمعروفة', 'أيضا'], ['والمعروفة', 'باسم'], ['أيضا', 'والمعروفة'], ['أيضا', 'باسم'], ['باسم', 'أيضا'], ['باسم', 'جائحة'], ['جائحة', 'أيضا'], ['جائحة', 'باسم'], ['باسم', 'فيروس'], ['جائحة', 'أيضا'], ['جائحة', 'باسم'], ['فيروس', 'كورونا'], ['جائحة', 'كورونا'], ['فيروس', 'باسم'], ['فيروس', 'جائحة'], ['كورونا', 'جائحة'], ['كورونا', 'فيروس'], ['جائحةٌ', 'عالميةٌ'], ['عالميةٌ', 'جائحةٌ'], ['عالميةٌ', 'مستمرةً'], ['مستمرةً', 'عالميةٌ'], ['عالميةٌ', 'حاليًا'], ['مستمرةً', 'جائحةٌ'], ['مستمرةً', ' عالميةٌ'], [' مستمرةً', ' حاليًا'], [' حاليًا', ' عالميةٌ'], [' حاليًا', ' مستمرةً'], [' سببها', ' فيروس'], [' سببها', 'جائحةٌ'], ['فيروس', 'كورونا'], ['فيروس', 'سببها'], ['سببها', 'كورونا'],

……..

……

……..

['حالة', 'صحية'], ['طوارئ', 'يشكل'], ['طوارئ', 'حالة'], ['طوارئ', 'صحية'], ['طوارئ', 'عامة'], ['صحية', 'حالة'], ['صحية', 'طوارئ'], [' صحية', ' عامة'], [' عامة', ' طوارئ'], [' عامة', ' صحية'], [' تبعث', ' القلق'], [' القلق', ' تبعث'], [' القلق', ' الدولي'], ['الدولي', 'القلق']]

- **Data_2 :** [[' فيروس', 'كورونا'], ['كورونا', 'فيروس'], ['كورونا', 'جائحة'], ['جائحة', 'كورونا'], ['جائحة', 'كوفيد'], ['كوفيد', 'جائحة'], ['اسمه', 'أيضا'], ['اسمه', 'فيروس'], ['أيضا', 'اسمه'], ['أيضا', 'فيروس'], ['أيضا', 'كورونا'], ['فيروس', 'اسمه'], ['فيروس', 'أيضا'], ['فيروس', 'كورونا'], ['كورونا', 'أيضا'], ['كورونا', 'فيروس'], ['جائحةٌ', 'عالميةٌ'], ['جائحةٌ', 'موجودة'], ['عالميةٌ', 'جائحةٌ'], ['عالميةٌ', 'موجودة'], ['عالميةٌ', 'حاليًا'], ['موجودة', 'جائحةٌ'], ['موجودة', 'عالميةٌ'], ['موجودة', 'حاليًا'], [' حاليًا', ' عالميةٌ'], [' حاليًا', ' موجودة'], [' سببها', ' فيروس'], [' سببها', ' كورونا'], [' فيروس', ' سببها'], [' فيروس', 'كورونا'], ['كورونا', 'سببها'],

……

……..

['الفيروس', 'حالة'], ['يشكل', 'انتشار'], ['يشكل', 'الفيروس'], ['يشكل', 'حالة'], ['يشكل', 'خطيرة'], ['حالة', 'الفيروس'], ['حالة', 'يشكل'], ['حالة', 'خطيرة'], ['حالة', 'صحية'], ['خطيرة', 'يشكل'], ['خطيرة', 'حالة'], ['خطيرة', 'صحية'], ['خطيرة', 'عالمية'], ['صحية', 'حالة'], ['صحية', 'خطيرة'], ['صحية', 'عالمية'], ['عالمية', 'خطيرة'], ['عالمية', 'صحية'], ['تزيد', 'القلق'], [' القلق', 'صحية'], [' حالة', 'صحية'], [' خطيرة', 'صحية'], [' عالمية', 'صحية'], [' صحية', 'عالمية'], [' عالمية', 'خطيرة'], [' تزيد', 'القلق'], [' القلق', 'الدولي'], ['الدولي', 'القلق']]

Les mots de l'entrainement (MotInTwo) sont convertis en deux tables .

La première **xtrains_1_i** représentée par la codification **one-hot** du mot_i dans le document_1

La deuxième **xtrains_2_i** représentée par la codification **one-hot** du mot_i (même mot) dans le document_2. Rappelons que la longueur de xtrains_1 et xtrains_2 sont len(Dictionnaire) Alors ma table xtrains au final sera De trois dimension xtreains(**len(mot_in_two) , 2 , len(dictionnaire)).**

Le résultat de la similarité cosinus entre le deux tables xtrains_1_i et xtrains_2_i est mis dans ytrains_i.

pour construire La table final de notre résultat ytrains, J'ai utilisé le code suivant pour faire ce travail.

```
k=0
sim_cos = 0
for mot in motintwo :
    xtrains1 = np.zeros(dict_size)
    xtrains1[Words_dict_2int[mot]] = 1
    for i in range(len(data1)):
        if mot == data1[i][0]:
            xtrains1[Words_dict_2int[data1[i][1]]] = 1
            i = i + 1
    xtrains2 = np.zeros(dict_size)
    xtrains2[Words_dict_2int[mot]] = 1

    for j in range(len(data2)):
        if mot == data2[j][0]:
            xtrains2[Words_dict_2int[data2[j][1]]] = 1
            j = j + 1
    xtrains[k][0] = xtrains1
    xtrains[k][1] = xtrains2
    ytrains[k][0] = 1 - spatial.distance.cosine(xtrains1 , xtrains2)
    sim_cos = sim_cos +  ytrains[k][0]
    k = k + 1
sim_cos = sim_cos /len(motintwo)


print(xtrains)
print(ytrains)
```

Prenons comme exemple le mot :**(45 : ' جائحة ') :**

Nous avons les deux tables xtrains (33, 2, 53) , ytrains(33, 1) tel que :

33= la longueur de motintwo .

53=la longueur de notre dictionnaire .

Ses voisins dans Data_1 qui appartient au mot_in_two sont **:**

32 : 'فيروس' , 26 : 'كورونا' , 50 : 'كوفيد' , 33 : 'ايضا' et 40 :'عالمية' .

Ses voisins dans Data_2 qui appartient à mot_in_two sont **:**

26 : 'كورونا' , 50 : 'كوفيد' , et 40 :'عالمية' .

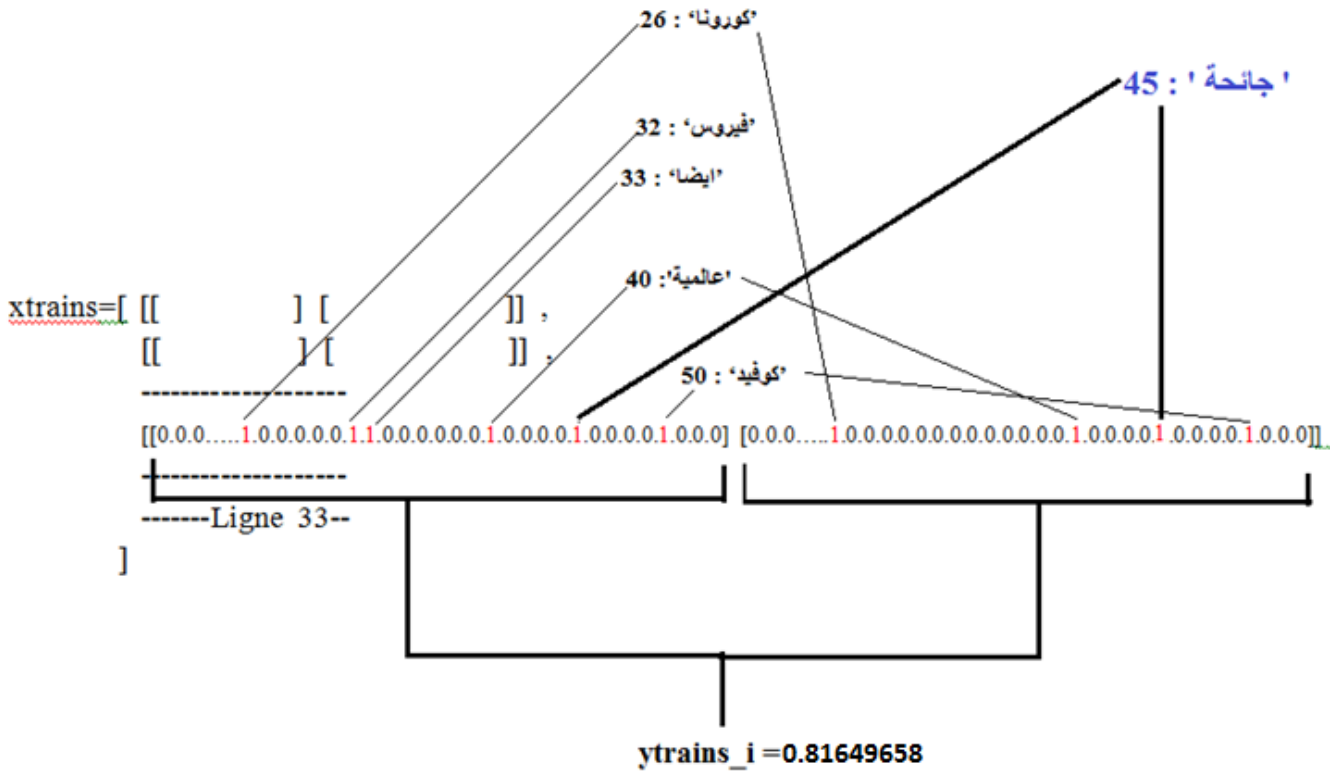


Figure 4.1 : Format ONE-HOT (0,1) d'un mot

Xtrains1 représente le **contexte** du mot dans le document1 et Xtrains2 représente a son tour le **contexte** du même mot dans le document2.

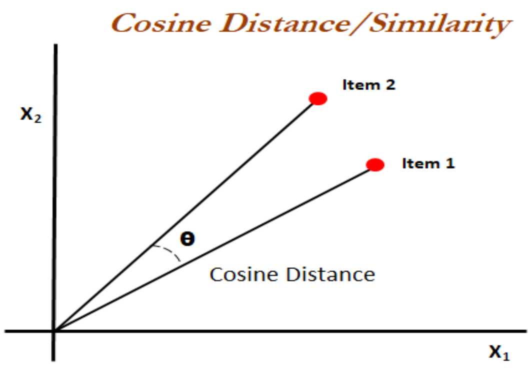
4.6 Similarité Cosinus

La similarité cosinus est l'une des mesures permettant de mesurer la similarité textuelle entre deux documents, quelle que soit leur taille dans le traitement du langage naturel. Un mot est représenté sous forme vectorielle.

Les documents textuels sont représentés dans un espace vectoriel à n dimensions.

Mathématiquement, la métrique de similarité cosinus mesure le cosinus de l'angle entre deux vecteurs à n dimensions projetés dans un espace multidimensionnel.

La similarité cosinus de deux documents va de 0 à 1. Si le score de similarité cosinus est de 1, cela signifie que les deux Vecteurs ont la même orientation. La valeur plus proche de 0 indique que les deux documents ont moins de similarité.



Cosine Distance. Ref: https://bit.ly/2X5470I

L'équation mathématique de la similarité cosinus entre deux vecteurs non nuls est:

Similarité_cos = $\cos(\theta) = \frac{A*B}{||A||*||B||} = \frac{\sum_{i=1}^{n} AiBi}{\sqrt{\sum_{i=1}^{n} Ai^2}\sqrt{\sum_{i=1}^{n} Bi^2}}$

Retournant a notre exemple et cherchant la similarité sémantique du mot «جائحة» entre les deux documents .

V_doc01 = [0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.0.0.1.1.0.0.0.0.0.0.0.1.0.0.0.0.1.0.0.0.0.0.1.0.0.0]

V_doc01 = [0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.0.1.0.0.0.0.0.1.0.0.0]

A * B = $\sum_{i=1}^{n} AiBi$ = (0*0) + (0*0) + ….. + (1*1) + …..+ (1*0) + (1*0)+ …+ (1*1)+.. + (1*1)+… + (1*1) +…(0*0)

= **4**

$\sqrt{\sum_{i=1}^{n}(Ai)^2} = \sqrt{0^2 + 0^2 + \cdots 1^2 + \ ... + 1^2 + 1^2 + \cdots 1^2 + \cdots + 1^2 + \cdots + 1^2} = \sqrt{6}$

$\sqrt{\sum_{i=1}^{n}(Bi)^2} = \sqrt{4}$

Similarité_cos = $\cos(\theta) = \frac{4}{\sqrt{6}*\sqrt{4}} = \frac{2}{\sqrt{6}}$ = 0.8164965809277261

Maintenant calculant le cosinus entre chaque valeur de xtrains[i][1] et xtraint[i][2] (pour chaque mot) et les mètres comme Résultat de teste dans y_trains[i] , j'ai utilisé la fonction :

From scipy import spatial

ytrains[k] = 1 - spatial.distance.cosine(xtrains1_k , xtrains2_k)

La figure ci-dessous explique bien le format des données de l'entrainement xtrains et ytrains pour un dictionnaire de longueur égale a 10.

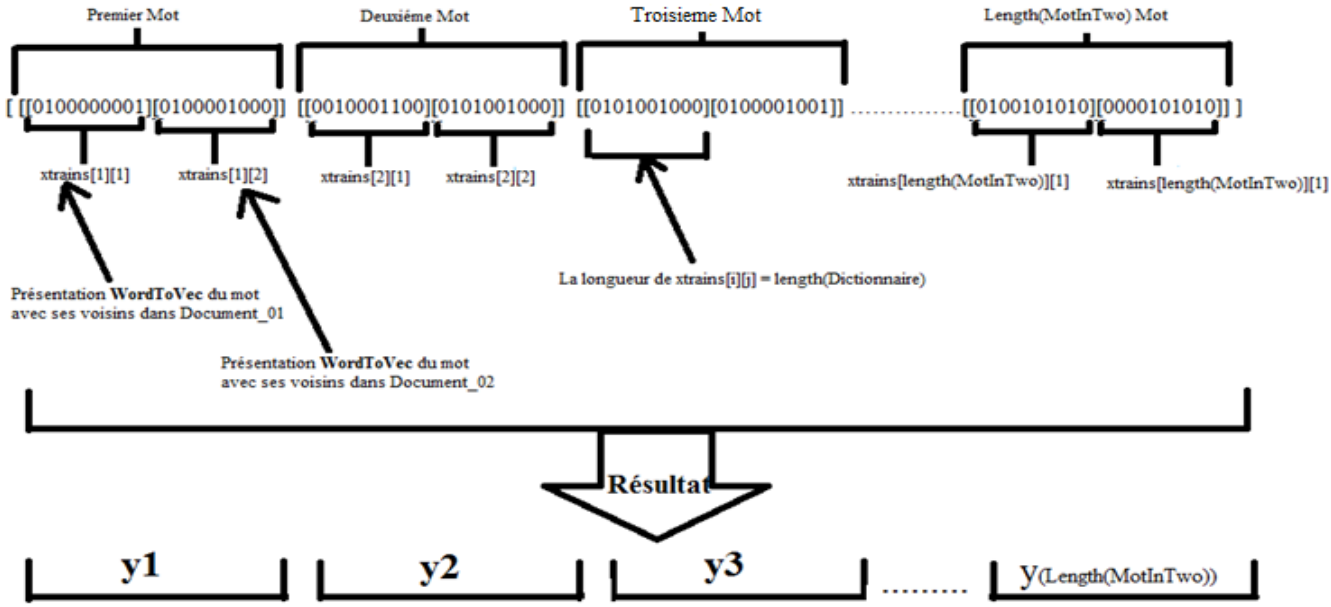


Figure 4.2 : Format ONE-HOT (0,1) des données de l'entrainement xtrains et ytrains

4.7 Le poids d'un mot

Un mot est peut être un nom , verbe ou adjectif afin que je puisse distinguer entre les trois types J'ai accordé un poids à chaque type :

Multiplier le résultat d'un mot comme suite :

- ✓ Si le mot est un nom résultat = résultat * 1
- ✓ Si le mot est un verbe résultat = résultat * 0.6
- ✓ Si le mot est un adjectif résultat = résultat * 0.8

4.8 Model Réseau de neurones

Nous avons maintenant nos données X_trains et le résultat Y_trains

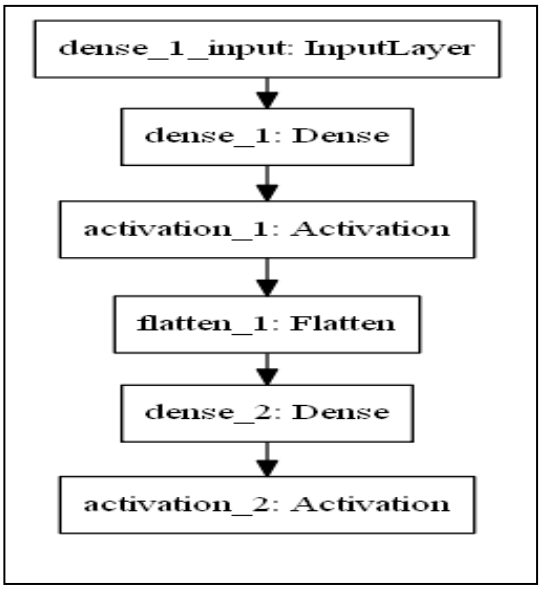
Alors on va entrainer notre model sur ses donner pour qu'il approche au résultat ytrains (Trouvé par calcul de cosinus) pour cela j'ai choisir le deep learning comme méthode.

Mon model est illustrer sur le code suivant :

```
x=xtrains
y=ytrains
v_size = len(motintwo) * 2
model = Sequential()
model.add(Dense(v_size , input_shape= (2,dict_size)))
model.add(Activation('tanh'))
model.add(Flatten())
model.add(Dense(1))
model.add(Activation('sigmoid'))
sgd=SGD(lr = 0.1)

model.compile(loss='binary_crossentropy' , optimizer=sgd)
model.fit(x,y,batch_size = 1 , nb_epoch = 1000)
print('La Simlarite Mot a mot avec RN est :  ' , model.predict_proba(x))
sum_sim = np.sum(model.predict_proba(x))/len(motintwo)
print('La Simlarite entre documents avec RN est  :  ' , sum_sim )
```

la représentation graphique de ce modèle est la suivante :



°°°°°°°°°°°°°°°°°°°°°°°°°°GRAPH°°°°°°°°°°°°°°°°°°°°°°°°°°

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 2, 66)	3564
activation_1 (Activation)	(None, 2, 66)	0
flatten_1 (Flatten)	(None, 132)	0
dense_2 (Dense)	(None, 1)	133
activation_2 (Activation)	(None, 1)	0

Total params: 3,697

Trainable params: 3,697

Non-trainable params: 0

Et les itérations effectuées par le réseau de neurones est les suivants :

```
Epoch 1/1000
 1/33 [..............................] - ETA: 3s - loss: 0.7027
33/33 [==============================] - 0s 4ms/step - loss: 0.4976
Epoch 2/1000
 1/33 [..............................] - ETA: 0s - loss: 0.0874
33/33 [==============================] - 0s 1ms/step - loss: 0.3826
```

Epoch 3/1000

1/33 [.....] - ETA: 0s - loss: 0.6295

30/33 [=========================>...] - ETA: 0s - loss: 0.3464

33/33 [==============================] - 0s 2ms/step - loss: 0.3475

Epoch 4/1000

1/33 [.....] - ETA: 0s - loss: 0.1910

24/33 [====================>.....] - ETA: 0s - loss: 0.2845

33/33 [==============================] - 0s 2ms/step - loss: 0.3232

Epoch 5/1000

1/33 [.....] - ETA: 0s - loss: 0.0142

33/33 [==============================] - 0s 1ms/step - loss: 0.3107

.....

.....

.....

Epoch 990/1000

1/33 [.....] - ETA: 0s - loss: 0.6932

30/33 [=========================>...] - ETA: 0s - loss: 0.2929

33/33 [==============================] - 0s 2ms/step - loss: 0.2663

Epoch 991/1000

1/33 [.....] - ETA: 0s - loss: 3.5192e-07

33/33 [==============================] - 0s 1ms/step - loss: 0.2663

Epoch 992/1000

1/33 [.....] - ETA: 0s - loss: 0.5042

33/33 [==============================] - 0s 1ms/step - loss: 0.2662

Epoch 993/1000

1/33 [.....] - ETA: 0s - loss: 3.4211e-06

25/33 [====================>.....] - ETA: 0s - loss: 0.2525

33/33 [==============================] - 0s 2ms/step - loss: 0.2661

Epoch 994/1000

1/33 [.....] - ETA: 0s - loss: 0.6812

27/33 [======================>.....] - ETA: 0s - loss: 0.2613

33/33 [==============================] - 0s 2ms/step - loss: 0.2664

Epoch 995/1000

1/33 [.....] - ETA: 0s - loss: 8.0885e-04

33/33 [==============================] - 0s 1ms/step - loss: 0.2663

Epoch 996/1000

1/33 [.....] - ETA: 0s - loss: 0.3939

33/33 [==============================] - 0s 1ms/step - loss: 0.2663

Epoch 997/1000

1/33 [.....] - ETA: 0s - loss: 0.6500

33/33 [==============================] - 0s 947us/step - loss: 0.2663

Epoch 998/1000

1/33 [.....] - ETA: 0s - loss: 1.2563e-06

33/33 [==============================] - 0s 1ms/step - loss: 0.2661

Epoch 999/1000

1/33 [.....] - ETA: 0s - loss: 3.4277e-07

32/33 [============================>.] - ETA: 0s - loss: 0.2744

33/33 [==============================] - 0s 2ms/step - loss: 0.2661

Epoch 1000/1000

1/33 [.....] - ETA: 0s - loss: 6.9797e-06

33/33 [==============================] - 0s 1ms/step - loss: 0.2661

Le résultat pour chaque mot est dans la table suivante :

MOT	similarité cosinus	La Similarité avec les réseaux de neurones
'أعلنت'	**[1.] * 0.6 = 0.6**	[0.99999416] *** 0.6 = 0,599996514**
'أيضا'	[0.25]	[0.2443564]
'الأولى'	[0.35355339]	[0.3510189]
'التنفسية'	**[1.] * 0.8 = 0.8**	[0.9999967] *** 0.8 = 0,79999736**
'الحادة'	[1.]	[0.9999995]
'الدولي'	[1.]	[0.999346]
'الشديدة'	**[1.] * 0.8 = 0.8**	[0.99999964] *** 0.8 = 0,799999712**
'الصحة'	[1.]	[0.9999995]
'الصينية'	[1.]	[0.99928224]
'العالمية'	[1.]	[0.99999785]
'الفيروس'	[0.8660254]	[0.86570543]
'القلق'	[0.66666667]	[0.67156255]
'المرتبط'	[1.]	[0.9998981]
'بالمتلازمة'	[1.]	[0.99998665]
'جائحة'	[0.81649658]	[0.81649658]
'حالة'	[0.8]	[0.79338235]
'حاليا'	[1.]	[0.9999316]
'ديسمبر'	[0.57735027]	[0.5634389]
'رسميا'	[1.]	[0.9999902]
'سارس'	[1.]	[0.9999972]
'سببها'	[1.]	[0.9991565]
'شهر'	[0.57735027]	[0.5634389]
'صحية'	**[0.5] * 0.8 = 0.4**	[0.50407785] *** 0.8 = 0,40326228**
'عالمية'	[0.57735027]	[0.5680795]
'فيروس'	[0.6]	[0.58485794]
'كورونا'	[0.91287093]	[0.9100591]
'كوف'	[1.]	[0.999993]
'كوفيد'	[1.]	[0.99944365]
'مدينة'	[0.8660254]	[0.8642119]
'منظمة'	[1.]	[0.9999987]
'ووهان'	[1.]	[0.99928224]
'يشكل'	**[0.67082039] * 0.6 =0,402492234**	[0.66487694] *** 0.6 =**
'يناير''	[0.5]	[0.4908973]

La similarité entre les deux documents est la somme des valeurs de la table y_trains divisé par le nombre de ces éléments

Résultat avec les réseaux de neurones $\text{Sim}_{\text{rn}} = \frac{\sum_{k=1}^{n} y_{\text{trains [k]}}}{n} = 0.8287766196511008$ n = 33

Tandis que le résultat cosinus $\text{Sim}_{\text{cos}} = 0.8310642356366503$

4.9 Changement des paramètres

On modifiant les différents paramètres nous obtenons la table suivante :

Taille de la fenêtre	Nombre itérations	Similarité Cosinus	Similarité Resultat_RN	Différence absolue
2	1000	0,8310642356366503	0,8309122721354166	0,0001519635012337
3	1000	0,8180580271967325	0,8179545547022964	**0,0001034724944361**
5	1000	0,8024500100288126	0,8025675108938506	0,000117500865038
7	1000	0,7918472469024334	0,7920054233435428	0,0001581764411094

4.10 Conclusion

Après cette expérience remarquons que le résultat le plus proche est la fenêtre de contexte de taille 3 ce qui explique que dans la langue Arabe la signification d'un mot est repose sur les trois mots qui l'entourent (Trois mots coté droite et trois coté gauche) .

Autrement dit Un mot dans la langue arabe peut avoir un sens propre si Le champ sémantique de ce dernier est composer de ce mot plus six autres mots.

Conclusion Générale:

Nous avons décrit le modèle proposé pour la représentation du sens des mots arabes dans un espace vectoriel basé sur les modèles de Word Embeddings. Ce modèle est inspiré des modèles vectoriels utilisés dans le domaine de la recherche documentaire, plus précisément le modèle des vecteurs conceptuels. En utilisant des mesures de similarité, nous avons calculé les distances entre les vecteurs qui représente les mots; et ce pour calculer les composantes des vecteurs du sens dans l'espace vectoriel, et en fin nous avons utilisé ce modèle pour le plagiat des textes dans un corpus.

Les avantages de notre modèle par rapport aux autres modèles, tiennent au fait qu'il se distingue par la proposition d'une base d'un espace vectoriel (par collection des champs sémantiques). La dimension d'un tel espace qui est variable, assure une exploitation optimale de l'espace mémoire selon le nombre des champs sémantiques de corpus en question.

Au stade de l'aboutissement de notre recherche, nous estimons que notre apport qui explore la problématique de la représentation de sens est importante bien que modeste comparé aux travaux de nos prédécesseurs. Nous souhaitons que notre travail ouvre d'autres pistes de recherche pour approfondir les travaux relatifs à la langue Arabe concernant notamment :

- prendre en compte les particularités morphologiques et syntaxiques de la langue dans le modèle proposé de notre étude.
- développement d'une ressource linguistique (lexiques, thésaurus) pour la langue arabe contient les champs sémantiques, les relations sémantiques entre les mots et les traits sémantiques des mots.
- Prendre en compte l'intensité des traits sémantique dans les mots .

Bibliographie

- **Jérémy Ferrero :** Similarités textuelles sémantiques translingues : vers la détection automatique du plagiat par traduction

- **EmmanuelleDusserre :** Utilisation de la méthode distributionnelle pour la constitution de classes sémantiques d'une liste de formes du Lexique scientifique transdisciplinaire .

- **Palash Goyal, Sumit Pandey , Karan Jain :** Deep Learning for Natural Language Processing.

- https://www.analyticsvidhya.com/blog/2017/06/Word-Embeddings-count-Word2veec/

- **El Moatez Billah Nagoudi, Jérémy Ferrero, Didier Schwab :** Amélioration de la similarité sémantique vectorielle par méthodes non-supervisées

- https://blog.cambridgespark.com/tutorial-build-your-own-embedding-and-use-it-in-a-neural-network-e9cde4a81296

- **Wikipedia** https://fr.wikipedia.org/wiki/Plagiat

- https://www.freecodecamp.org/news/demystify-state-of-the-art-text-classification-word-embeddings/

- **Word2vec (Skip-gram) :** https://towardsdatascience.com/word2vec-from-scratch-with-numpy-8786ddd49e72