MINISTERE DE L'ENSEIGNEMENT SUPERIEURET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLEB – BLIDA 01

FACULTE DES SCIENCES

DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE MASTER

Spécialité: Traitement Automatique de la Langue

THEME

Vers une plateforme de gestion des corpus et d'analyse de texte en langue arabe.

Présenté par :

- ✓ **M. Benhamida Abdennour**

Proposé et Encadré par :

- ✓ **Mme. Ouahrani Leila**

Soutenu le : 09/09/2020

Devant le jury Composé de

M.	FERFERA SOFIANE	Président
M.	BENAISSI SELAMI	Examinateur

Année Universitaire : 2019/2020

Abstract

For languages like English, the resources available are very broad, varied and available. But for some languages, there are few such resources. Arabic is a suitable example. Although it is a widely spoken language, it has been widely recognized that it has few tools and resources available to the public. In particular, the Arabic NLP lacks resources such as corpora, lexicons, dictionaries, datasets in addition to fully automated fundamental tools such as tokenizers, part of speech markers, analyzers, stemmers and more. semantic role taggers. This lack negatively affects language searches and prevents proper progress in automatic processing.

The objective is to develop a platform for corpus management and textual analysis to consolidate the needs of NLP in Arabic.

In this work, we propose a platform for corpus management which consolidates NLP resources starting with corpora and their categorization up to the various possible textual, syntactic and semantic analyzes. A platform that addresses the lack of public resources and the limitations of data collection and analysis.

Keywords: NLP, Corpus, Arabic, textual analysis, data collection, lack of tools, platform development.

Résumé

Pour des langues comme l'Anglais, les ressources disponibles sont très larges, variées et disponibles. Mais pour certaines langues, il existe peu de telles ressources. L'arabe est un exemple approprié. Bien qu'il s'agisse d'une langue largement parlée, il a été largement reconnu qu'elle dispose de peu d'outils et de ressources accessibles au public. En particulier, le TAL Arabe manque de ressources telles que les corpus, les lexiques, les dictionnaires, les jeux de données en plus des outils de fondamentaux entièrement automatisés tels que les tokeniseurs, les marqueurs de partie du discours, les analyseurs, les stemmers et les étiqueteurs de rôles sémantiques. Ce manque affecte de façon négatives les recherches concernant la langue et l'empêche d'avancer correctement dans le traitement automatique.

L'objectif est de développer une plateforme de gestion des corpus et d'analyse textuelle pour consolider les besoins du TAL en arabe.

Dans ce travail, nous proposons une plateforme pour de gestion de corpus qui consolide les ressources du TAL en partant par les corpus et leur catégorisation jusqu'aux différents analyses textuelles possibles, syntaxique et sémantique. Une plateforme qui répond au manque de ressources public et aux limitations concernant l'assemblement et l'analyse des données.

Mots clés : TAL, Corpus, l'Arabe, analyse textuelle, collection de données, manque d'outils, développement d'une plateforme.

ملخص

بالنسبة للغات مثل الإنجليزية، فإن الموارد الموجودة متنوعة ومتاحة. لكن بالنسبة لبعض اللغات، هناك القليل من هذه الموارد. اللغة العربية هي مثال مناسب. على الرغم من أنها لغة يتم التحدث بها على نطاق واسع، إلا أنه من المعروف على نطاق واسع أن لديها القليل من الأدوات والموارد المتاحة للجمهور. على وجه الخصوص، تفتقر البرمجة اللغوية العصبية العربية إلى الموارد مثل البيانات، والمعاجم، والقواميس، ومجموعات البيانات بالإضافة إلى الأدوات الأساسية المؤتمتة بالكامل مثل الرموز المميزة وجزء من علامات الكلام والمحللات والمشتقات والمزيد. علامات الدور الدلالية. يؤثر هذا النقص سلبًا على عمليات البحث عن اللغة ويمنع التقدم المناسب في المعالجة التلقائية.

الهدف هو تطوير منصة لإدارة النصوص والتحليل النصي لتوحيد احتياجات البرمجة اللغوية العصبية في اللغة العربية.

في هذا العمل، نقترح منصة للاستفادة من الموارد التي تدمج موارد البرمجة اللغوية العصبية بدءًا من المجموعة وتصنيفها وصولاً إلى مختلف التحليلات النصية والنحوية والدلالية الممكنة. منصة تعالج نقص الموارد العامة والقيود المفروضة على جمع البيانات وتحليلها.

الكلمات المفتاحية: البرمجة اللغوية العصبية, تجميعات النصوص, اللغة العربية, تحليل النصوص, نقص الموارد, تطوير منصة.

Remerciements

Ce travail de mémoire met fin à notre cycle de Master et n'a pu aboutir qu'avec l'aide et les encouragements de plusieurs personnes. Nous espérons pouvoir exprimer notre profonde gratitude envers toutes ces personnes.

D'abord, à **Allah** tout puissant, qui nous a aidé, comblé d'amour et nous a donnér la force de continuer aux moments les plus dures de toute notre vie.

Nos sincères remerciements vont à nos enseignants et membres de l'équipe pédagogique du département d'informatique qui ont contribué, directement ou indirectement de près ou de loin à la réalisation de ce travail. Leurs leçons, commentaires, suggestions et conseils ont influencé notre parcours académique et de vie.

Avec un sincère sentiment de respect et de gratitude, nous remercions notre promotrice, Mme. Ouahrani Leila, nous sommes reconnaissants pour ses efforts dans notre travail et pour sa motivation et ses qualités pédagogiques tout au long de cette période difficile, et qui nous serviront aussi à l'avenir. Son soutien, ses conseils, les conseils et son investissement lors de la préparation de ce mémoire étaient sans doute crucial.

Nous tenons à remercier les membres du jury pour leur aimable participation à l'évaluation de ce travail, nous en sommes honorés et nous leur exprimons notre profonde reconnaissance.

Nos vifs remerciements de reconnaissance sont également adressés à nos chers parents qui ont sacrifié énormément pour nous. Ils ont toujours été là pour nous encourager et nous aider avec grand amour. Il va de soi pour nos grands-parents qui nous ont comblé de joie durant notre enfance. Une pensée à notre grand-père « Boualam Abdelkader », il aurait été fier de nous voir en arriver là.

Nous tenons aussi à présenter notre gratitude et remercîments envers les familles « Benhamida et Boualam », y compris les cousins, cousines, frères et sœurs de nous avoir aider moralement avec leurs paroles, conseils et expériences.

Un grand remerciement va aussi à Mohamed Hadjersi et Oussama Benguergoura, nos camarades avec qui nous avons pu travailler plusieurs fois durant cette période.

En fin que les amis, frères et sœurs dont les noms ne sont pas cités ne nous tiennent pas rigueur, nos pensées vont aussi vers eux.

Liste des figures

Liste des Tables

Table des matières

Table des Abréviations

Abréviation	Signification
TAL	**Traitement automatique de la langue**
SVM	**Support Vector Machine**
POS	**Part Of Speech**
HTML	**Hypertext Markup Language**
API	**Application Programming Interface**
TF	**Terme Frequency**
IDF	**Inverse Document Frequency**
IA	**Intelligence Artificielle**
ASAG	**Automated Short Answer Grading**
COALS	**Correlated Occurrence Analogue to Lexical Semantics**
WE	**Word Embedding**
NLTK	**Natural Language Toolkit**

Introduction générale

1-Introduction

Les chercheurs en linguistique ont développé une large gamme d'outils de Traitement Automatique en Langage naturel (TAL) pour analyser et annoter automatiquement différentes langues. Ces ressources TAL sont documentées par des efforts tels que la carte des ressources linguistiques et de l'évaluation. Mais pour certaines langues, il existe peu de telles ressources. L'arabe est un exemple approprié. Bien qu'il s'agisse d'une langue largement parlée, il a été largement reconnu qu'elle dispose de peu d'outils et de ressources accessibles au public. En particulier, le TAL Arabe manque de ressources telles que les corpus, les lexiques, les dictionnaires, les jeux de données en plus des outils de fondamentaux entièrement automatisés tels que les tokeniseurs, les marqueurs de partie du discours, les analyseurs, les stemmers et les étiqueteurs de rôles sémantiques. Un manque de données et de recherches suffisantes a affecté négativement les praticiens du traitement du langage naturel arabe.

2-Problématique

L'utilisation de corpus est indispensable dans les approches basés corpus très favorables pour la langue arabe dans le cadre de plusieurs contextes : évaluation automatique, calcul de similarité, recherche d'information. Concernant l'Arabe, quelques points s'imposent :

- Manque évident de ressources linguistiques en langue arabe,
- Le besoin de consolidation de ressources linguistiques
- Nous avons déjà élaboré, dans un précédent travail, une approche de conception automatique de corpus mais rien n'est encore fait concernant la gestion de ces corpus ainsi que l'analyse linguistique (plutôt statistique) sur ces corpus. Nous voulons de plus exploiter le DUMP Wikipédia pour développer une approche de conception de corpus spécifiques de domaine.

3-Objectifs :

L'objectif principal consiste à développer une plateforme pour des "ready-to-use corpora" pour la langue arabe tout en restant générique par rapport à d'autres langues. Le développement de la plateforme passe par la réalisation des objectifs suivants :

- Procéder à la catégorisation des corpus par utilisation est importante (annotés, web, manuels, ...),
- Explorer toutes les plateformes de gestion de corpus pour les comparer et retenir les fonctionnalités.
- Élaborer toutes les fonctionnalités liées à :
- La gestion des corpus (ajout, création, enrichissement, extraction depuis un corpus, …)
- L'analyse textuelle et fréquentielle statistique des textes dans le corpus.
- Intégrer les fonctionnalités liées au calcul de similarité et à la génération d'espaces sémantique et de dictionnaire distribué. Nous penserons aussi à donner un modèle d'apprentissage automatique (SVM ou de régression linéaire) pour combiner les approches de similarité entre deux textes courts développées dans des travaux précédents.
- Exportation des analyses sous une forme utilisable en assurant les exportations nécessaires.
- Les plateformes déjà disponibles montrent toutes des limites et c'est le facteur principal à dépasser lors du développement du notre.
- Il faut offrir le choix du texte à utiliser dans les analyses, ce qui donnera plus de liberté aux utilisateurs.

4-Importance du travail :

L'importance de notre travail réside fortement aussi dans ses objectifs. Le travail final apportera des plus pour le domaine et donnera un outil pouvant faciliter le travail de plusieurs personnes, et sur tout, accessible à tout le monde :

- Offrir une plateforme aux étudiants, enseignants et chercheurs pour aboutir à leurs objectifs linguistiques.
- Contribuer à la résolution du problème de manque d'outils non disponibles gratuitement et qui n'offrent pas assez de fonctionnalités.
- Donner l'accès à ces outils permettra aux utilisateurs de les exploiter et les enrichir.
- Fournir des résultats immédiats et exploitables en dehors de la plateforme dans d'autres recherches.

5-Structure du mémoire :

Les prochains chapitres sont structurés comme suit :

- Le chapitre 1 présentera l'état de l'art du thème proposé, en passant par les différents domaines présentés dans la problématique et les recherches actuelles et avancements des domaines
- Le chapitre 2 expose l'architecture fonctionnelle de notre plateforme de gestion de corpus ensuite la conception et la réalisation.
- Le dernier chapitre représente les performances de notre outil ainsi que son l'évaluation générale.
- Nous terminons par une conclusion mettant le point sur le travail réalisé ainsi que des perspectives possibles.

Chapitre 1 :
État de l’art.

Cette section représente une exposition ainsi que l’état des connaissances dans le domaine du thème étudié. Cette étude bibliographique nous permettra de mieux cerner le sujet et apprendre des autres recherches :

- Les ressources linguistiques en langue arabe.
- Les Corpus.
- Les Gestionnaires de corpus.
- Similarité Textuelle.
- Distribution des mots.

1-Les ressources linguistiques en Arabe

1.1-Présentation de la langue

L'arabe est largement parlée dans le monde aujourd'hui. La population arabophone est assez grande dans le monde avec ces différents dialectes. Originaire des langues sémitiques, l'arabe est également la langue dans laquelle le livre sacré du Coran est écrit, ce qui en fait une incitation principale pour les étrangers pieux à prendre la maîtrise de cette langue sacrée. L'influence de la culture arabe est visible dans des langues telles que le turc, le bosniaque, le persan, l'hindi et l'anglais avec certains mots qui ont des origines arabes lointaines.

1.2-L'Arabe dans le domaine du TAL

Bien que cette langue soit très connue et largement parlée dans le monde entier, ses ressources linguistiques utilisables dans le domaine du traitement automatique de la langue ne connaissent pas le même succès. Comparé à d'autres langues, l'arabe est une langue très pauvre en termes d'outils de traitement, même les plus basics d'entre elles, ainsi que le nombre de corpus ouverts au public pour les utilisations académiques. Ce qui la rend parmi les langues dites « pauvres » dans le domaine.

Cela est d'un côté dus au fait que cette langue, contrairement à l'anglais par exemple, est une langue très complexe et qui ne manque pas d'ambigüités. Dans la recherche d'étiquetage grammatical [1] les auteurs ont pu déceler par exemple que le manque total de voyelles est un facteur qui rend un texte arabe très difficile à traiter car 74% de ses mots sont ambigus rien qu'à cause de ce phénomène à lui seul, notons aussi ce qui suit :

- **L'absence des voyelles :** L'arabe est une langue qui se base sur le « tachkile » qui est une sorte de signes (ًٌٍَُِّْ) pour remplacer les voyelles utilisées par d'autres langues. Donc pour un mot qui peut être lu de plusieurs façon comme le mot "كتب" qui a 7 façons de l'écrire, avec un sens différent pour chacune des façon, « katab كَتَبَ : a écrit », « kutiba كُتِبَ : a été écrit » ou encore « kutub كُتُب : livres ». Un programme qui fait la différence entre ces formes est très difficile à concevoir.
- **Vocabulaire très large :** Le vocabulaire de la langue arabe est très large, au point ou le mot « lion » peut avoir 500 autres mots qui l'expriment, comme le cite Ibn Khâlawiha[2]
- **La gémination :** Représente ou doublement d'une lettre pour marquer sa prononciation, pour le français par exemple, le mot grammaire à le « m » en double mais qui ne se voit pas lors de la prononciation du mot, par contre « kattaba كَتَّبَ» sera bien plus visible et ainsi affectera le sens et le type du mot en question, en effet, « kattaba » signifie « il a fait écrire » alors que « kataba » avec un seul t veut dire « il a écrit ».

1.3-Le projet de « الذخيرة العربية[1] » :

Ce projet proposé par l'organisation arabe de l'éducation et culture et la science, est un projet de grande envergure. Depuis son apparition en 1988, il consiste à construire une banque linguistique pour la langue arabe, avec la contribution de tous les pays arabes y compris l'Algérie. Ce projet cherche à construire un lieu où les utilisateurs

[1] https://www.marefa.org/مشروع_الذخيرة_العربية

pourront bénéficier d'un large sommaire d'outil arabe afin d'avancer dans leurs recherches, mais aussi de faire connaitre l'arabe dans le monde en contribuant d'avantage à l'avancement du Tal et des langues en générale.

L'outil projette d'utiliser des méthodes modernes ainsi que des machines puissantes pour calculer tous les besoins des utilisateurs, et projettera plus de lumière sur la langue de notre temps, en plus de son utilisation pendant l'histoire de certaines civilisations du passé, ce qui représente une grande opportunité aux chercheurs de mieux comprendre l'histoire dans tous ces domaines à travers le temps. Le projet ne voit aucun rival actuellement sous tous ses aspects, dans ces dimensions et avec cette exhaustivité. Il est d'une grande importance avec des dimensions culturelles très larges, et il est naturel que ce projet soit mis en œuvre dans le cadre d'une action commune en raison de son ampleur.

Tous ces documents et ouvrages qui seront introduits dans les serveurs, pourront être accessibles de façon automatique et intuitive comme n'importe quel autre moteur de recherche effectuant cette tâche, mais en se concentrant sur la langue arabe, avec un autre côté du projet qui vise les autres publications à travers le monde, afin de les traduire et publier aussi dans le cadre de la science, en offrant aux utilisateurs arabe une façon de rester à jour et suivre les dernières actualités du vaste monde.

L'Algérie a lancé un site pour l'organisation afin que les membres puissent communiquer et avancer dans le travail avec plusieurs personne recrutés pour enregistrer les données. Par contre aucun outil concret n'est disponible depuis toutes ses années, le projet n'a pas beaucoup avancé malheureusement.

2- Concepts fondamentaux de la gestion de corpus

2.1- Introduction

Dans le traitement automatique de la langue, il y a ce qu'on appelle dans le domaine « ressources linguistiques », représentant un outil indispensable pour toute sorte d'étude et impossible de travailler sans ce dernier. Les corpus sont disponibles à travers le monde sous plusieurs formes, tailles et domaines. Défini (en TAL) comme une collection de textes authentiques lisibles par machine (y compris les transcriptions de données parlées) qui sont échantillonnés pour être représentatifs d'une langue naturelle ou d'une variété de langue particulière. Les corpus jouent un rôle essentiel dans la recherche sur le traitement du langage naturel (TAL) ainsi que sur un large éventail d'enquêtes linguistiques. Ils fournissent une base matérielle et un banc d'essai pour la construction des systèmes TAL.

Il existe des milliers de corpus dans le monde, mais la plupart d'entre eux sont créés pour des projets de recherche spécifiques et ne sont pas accessibles au public (et souvent payant). Richard Xiao [3] fournit une étude complète d'un large éventail de corpus connus et influents en anglais et dans de nombreuses autres langues. Peu importe la langue étudiée, il existe plusieurs catégories de corpus disponible à l'utilisation, peuvent être général ou spécialisé et catégorisé comme suit :

- **Corpus de texte :** représentant les collections de documents textuelles brut tel que les multilingues et monolingues, articles web et collections de chaque dialectale (généralement des tweets)

- **Corpus annotés :** regroupes les corpus contenants des simplifications et des analyses regroupées sous forme d'ensembles sémantique, POS[2], indication d'erreurs etc.
- **Lexique :** contenant des listes de mot et bases de données lexicales
- **Corpus Manuel :** écrits à la main ou scannés et transformés en textes utilisables dans une machine.
- **Corpus de discours oral :** des textes retranscrit depuis des audios enregistrés
- **Divers :** cette catégorie regroupe les corpus divers du genre Q&A[3] ou la détection de plagiat pour certain cas d'utilisation

Les corpus monolingues sont généralement les plus utilisés.

2.2- Création d'un corpus

Avant de procéder à la création d'un corpus de n'importe quel type, il faut comprendre que comme chaque outil de travail ou étude scientifique, il y'a des caractéristiques et des normes à prendre en considération pour déterminer la validité d'un outil. Les corpus également bénéficient de leurs lots de caractéristiques nécessaires avant l'utilisation de ce dernier. Pour dire qu'un corpus est adéquat à une étude de TAL, il est important de prendre en considération les facteurs suivants :

- **La taille :** La taille d'un corpus est souvent très relative, liée aux besoins de l'études et la langue. Par exemple, pour comparer deux dialectes d'une même langue, il faudra associer deux corpus chacun représentatif de l'un de ces deux derniers. En prenant par exemple l'anglais américain et british, nous nous retrouvons vite avec deux corpus très large (des millions de mots) étant donnée la grande richesse de l'anglais en termes de ressource linguistique. Au début des année 60s où le TAL commençait à prendre son élan dans le monde de l'informatique, les machines de l'époque étaient très limitées en termes de performances (puissance de processeur, mémoire vive etc.) et aussi de stockage, le corpus « Brown[4] » de l'université de Stanford contenant un million de mots était très difficile à gérer, Alors que maintenant un corpus de 500 millions de mot comme le « Bank of English » est considéré comme étant relativement petits face à la grande puissance de calcul des machines de nos jours, la taille d'un corpus va donc être affectée des machines utilisés par les chercheurs de façon directe. Un autre élément concernant la taille, c'est qu'on ne peut pas prendre tous les documents disponibles et les associer pour obtenir un corpus d'une taille souhaitée, car nous sera souvent confronté à des problèmes de droits d'auteurs [5] quant aux livres et publications non libre de droit, pour ça que les corpus basé web sont souvent préférables. Pour conclure, nous pouvons dire que la taille du corpus va être affecté par le matériel disponible, les article et publication libres de droit ainsi que la disponibilité des ressources à utiliser.
- **Représentativité :** Ce qui fait la différence entre un corpus de TAL et des archives (collection de textes aléatoires) est la façon dont le corpus est conçu. Il a une architecture et un design qui représente une langue (ou plusieurs comme le cas des corpus multilingues). Le corpus est considéré comme représentatif de la variété linguistique qu'il est censé représenter si les résultats basés sur son contenu peuvent être généralisés à ladite variété linguistique. Biber [6] définit la représentativité du

[2] Part Of speech : chaque mot à une valeur propre à lui dans une phrase (nom, verbe, adj. etc.)
[3] Question réponse

point de vue de la manière dont cette qualité est obtenue : « La représentativité se réfère à la mesure dans laquelle un échantillon inclut la gamme complète de variabilité dans une population ». Un corpus qui représente bien une langue, ou une population, est un corpus respectant les genres, les domaines et inclus les médias tel que les corpus basés sur la BBC et les journaux, car ils sont écrit ou parlé par des personnes utilisant la langue comme outil de communication en la rendant vif est toujours à jours comme le suggère Hunston[7], pour qu'un corpus reste représentatif, il faut le mettre à jour souvent avec de nouveaux textes pour suivre les changement des dialectes par exemples.

- **L'équilibre :** l'équilibre d'un corpus est déterminé par ses utilisations. Par conséquent, un corpus général qui contient à la fois des données écrites et parlées (par exemple le BNC4) est équilibré, il en est de même des corpus écrits tels que Brown [4], et des corpus parlés ou des corpus spécifiques à un domaine qui peuvent également prétendre être équilibrés. Un corpus équilibré couvre généralement un large éventail de catégories de texte censées être représentatives de la langue ou de la variété linguistique considérée. Ces catégories de texte sont généralement échantillonnées proportionnellement pour inclusion dans un corpus de sorte qu'il offre un modèle à petite échelle gérable du matériel linguistique que les constructeurs de corpus souhaitent étudier.
- **Échantillonnage :** Ce facteur est le résultat de la combinaison des facteurs précédent, représentativité et équilibre. Due au fait que le langage naturel humain ne peut pas être représenter de façon formelle, il est important d'avoir un échantillonnage assez réaliste et qui donne la meilleure représentation de la langue, ainsi que l'équilibre entre la requête et le résultat. Afin d'obtenir un échantillon représentatif d'une population, l'une des préoccupations à traiter est de définir l'unité d'échantillonnage et les limites de la population. Pour le texte écrit, par exemple, une unité d'échantillonnage peut être un livre, un périodique ou un journal. Quant à l'échantillonnage, plusieurs questions se posent, comme la taille de l'extrait d'un livre ou journal (vu que se sont deux éléments qui distingue une population linguistique d'une autre) qui est relatives à chaque besoin, et aussi le type des documents dont nous dispose, mais dans tous les cas, il est nécessaire que l'échantillon tiré d'une fenêtre d'échantillonnage (ensemble de document dans l'extrait viendra) soit adéquat à la représentation du corpus.

Après avoir vu les caractéristiques d'un corpus ainsi que les types qui existent, les techniques permettant de créer des corpus utilisables sont relatives au besoin, la suite de cette section mettra en avant les méthodes les plus fiables et les plus utilisés dans le domaine :

- **Utilisation du web :** Cette méthode et récente en considérant l'âge du TAL dans l'histoire. C'est Resnik en [8] et Ghani[9] qui supposent que l'utilisation du web pour les langues minoritaires peut être très intéressante, notamment les journaux de presse qui propose plusieurs langues, pourrait être utiliser dans des travaux de traduction automatique. Cette méthode passe par trois étapes : lancement de requête (ou insertion de site web), extraction de fichier HTML et ensuite le nettoyage des textes brut obtenus. Dans le travail[5] de Dhaou Ghoul[10] dont le but était de construire un corpus arabe depuis un journal de presse « Al watan » car

[4] British national corpus

[5] http://lexicometrica.univ-paris3.fr/jadt/jadt2014/01-ACTES/22-JADT2014.pdf

c’est des sites qui proposent des fichiers Html regroupés par domaine, facilement gérable et exploitables. Le résultat obtenu est un corpus arabe constitué de 7,653,881 mots distribués sur quatre domaines différents : sport, religion, culture et économie. La figure suivante présente un schéma de construction :

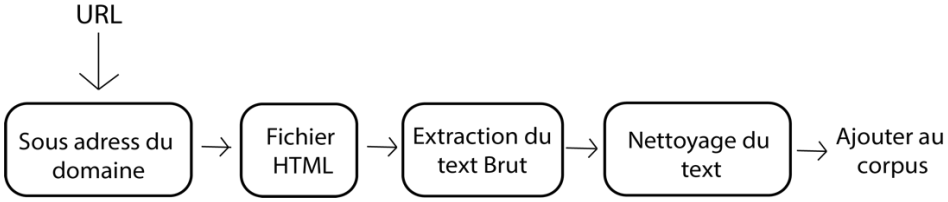

Figure1 : Schéma de l’extraction de données depuis le web

- **Dump[6] Wikipédia pour les corpus de domaine :** Les corpus de texte sont un outil indispensable pour tout étude en TAL, heureusement, Internet regorge de fichiers textuelles et données lisibles par la machine. L’une de ces collections est celui de Wikipédia, qui est l’encyclopédie de référence par excellence utilisée par des millions de personnes venant de tout le globe, et plus important, disponible en plusieurs langues. Celui de l’arabe[7] par exemple est constitué de 31go de données textuelles, ce qui représente une taille de corpus déjà grande et statistiquement parlant, peut avoir des résultats très convaincant. Le Dump présenté peut être utilisé de plusieurs façon, mais la génération de corpus spécifique de domaine, notamment pour la langue arabe, ne nécessite pas la collection dans son intégralité. L’article du MediaWiki [11] montre comment nous pouvons utiliser l’API[8] du moteur de recherche de Wikipédia pour lancer des requêtes et en recenser le contenu des résultats sous forme JSON[9], le résultat est paramétrable de plusieurs façons comme le nombre de pages (articles) à retourner, nombre de mots ou encore combien de sous articles à explorer. Cette méthode est globalement similaire à celle de l’extraction du web, sauf que là nous ne s’intéresse qu’à une seule collection qui sera par la suite filtrer grâce aux mots de la requête pour ne garder que les articles spécifiés, et peut donc générer des corpus spécifiques du domaine souhaité.

- **Méthode manuelle :** L’une des méthodes possibles, est de le faire manuellement, mais ça reste très difficile vu les tailles des corpus nécessaire pour les études de nos jours. Les livres avec plusieurs tomes ou les publications en papiers (qui seront scannées pas la suite) qui sont publiques pour des travaux d’analyse de sentiments par exemple, mais ça reste quand même très couteux en termes en termes de collection et d’équilibrage du corpus finale.

[6] Une copie brute (sans transformation) de données.
[7] https://archive.org/details/arwiki-20190201

[8] Interface de programmation d'application, un intermédiaire logiciel permettant l’interaction entre applications

[9] JavaScript Object Notation, un format de données textuelles dérivé de la notation des objets du langage JavaScript

3-L'Arabe et les corpus publics

Il est possible de générer un corpus arabe depuis le web ou même depuis le dump Wikipédia. Par contre, les corpus disponibles librement dans le web pour la langue restent très limités, surtout si nous le comparons à d'autres langues qui eux ont un nombre très grand de corpus de tout genre bien équilibrés et présentés librement. Cela ne veut pas forcément dire la que la langue arabe n'a pas de corpus disponibles.

La suite de ce rapport se base sur l'étude menée par Wadjdi Zaghouani[12] concernant les corpus disponibles gratuitement pour la langue arabe, regroupés par type :

- **Monolingue :** 11 corpus sont disponibles, avec des tailles très large pouvant aller jusqu'à 113 millions de mots. Globalement stocké sous forme de texte brut ou .XML, mais aussi .PDF qui un format pas évident à exploiter (problème de conversion) :

Source	Corpus	Nbr Mots	Disponibilité
Abdelali	Ajdir Corpora**[13]**	113,000,000	Oui
AlRabiah	KSU classical arabic**[14]**	50,000,000	Non
Saad et Asbour	OSAC**[15]**	18,183,511	Non
Abbas	Al watan**[16]**	10,000,000	Oui
Zarrouki	Tashkeela**[17]**	6,149,726	Oui
Abbas	Al Khaleej**[18]**	3,000,000	Oui
Al-Thubaity et al	Kacst arabic newspaper corpus**[19]**	2,000,000	Oui
Al Saadi	Arabic words corpora**[20]**	1,000,000	Oui
Al Soleiti	Corpus of contemporay arabic**[21]**	842,684	Non
Al Kalhan et al	CRI KACST arabic corpus**[22]**	235,000	Non
Farwaneh	Arabic learners written corpus**[23]**	50,000	Non

Tableau1 : Listes des corpus Monolingue gratuits en arabe.

- **Multilingue :** Il existe plusieurs corpus disponibles, mais selon Wadjdi Zaghouani, le « UN » est le plus important et le plus utilisé parmi ceux de cette catégorie, suivit par le « Meedan » avec 1 million de mots disponible. :

Source	Corpus	Nbr Mots	Disponiblité
Rafalovitch and al	UN Corpus (partie arabe) **[24]**	2,721,463	Oui
Bounhas	Hadith Standard Corpus**[25]**	2,500,000	Non
Meedan	Meedan Translation Memory**[26]**	1,000 ,000	Oui
CLSP / JHU	Egypt Traslation Toolkit**[27]**	80,000	non

Tableau2 : Listes des corpus Multilingue gratuits en arabe.

- **Dialectale :** Les corpus conçu à partir des dialectes sont très utilisés dans le domaine de reconnaissance de speech. Dans le tableau suivant il y'aura deux seulement qui sont disponible librement. Avec celui de « Almeman and lee[28] » et ses 2 millions de mots, il représente une très grande valeur dans la catégorie :

Source	Corpus	Nbr Mots	Disponibilité
Almeman and lee	Arabic multi dialect text corpora**[29]**	2,000,000	Non
Graja et al	Tunisian dialect corpus**[30]**	3,403	Oui

Tableau3 : Listes des corpus Dialectale gratuits en arabe.

- **Corpus basé sur le web :** Les corpus basé sur le web, ils représentent un groupe de corpus très variés, très larges et assez polyvalents, ce qui les rends très facile à exploiter dans le TAL arabe :

Source	Corpus	Nbr Mots	Disponibilité
Al-Thubaity	KACST arabic corpus[31]	732,780,509	Non
Leeds	Leeds arabic internet corpus[32]	317,000,000	Non
Alansary et al	Internation corpus of arabic[33]	100,000,000	Oui
Parkinson	ArabicCorpus[34]	100,000,000	Oui
Abbas N	Qurany[35]	78,000	Oui
Sharaf et al	Quranic text mining dataset[36]	24,000	Non

Tableau4 : Listes des corpus web gratuits en arabe.

- **Corpus annotés :** Cette catégorie est très importante, car déjà pour les annotations (les ajouts) il faut une supervision pour y arriver, et ils représentent un outil majeur dans le POS et d'autres traitements du Tal qui cherchent à reconnaitre les rôles, les corpus de cette catégorie ont été pris, traité et tagué pour au final avoir des corpus utilisables tout en étant annotés.
- **Corpus nommés :** Cette catégorie qui représente des annotations pouvant reconnaitre les noms et les relations entre ces derniers, permettant de bien comprendre le domaine ainsi que leur utilisation, et les corpus disponible sont :

Source	Corpus	Nbr Mots	Disponibilité
Steinberger et al	JRC-Names[37]	23,000	Non
Ben ajiba et al	AnerCorp[38]	150,000	Non
Mohit et al	Aqmar names entity corpus[39]	75,000	Non
Azab	Named entity translation lexicon[40]	55,000	Oui
Attia et al	Names entities list[41]	45,202	Oui
Ben Ajiba et al	AnerGazet[42]	14,000	Oui

Tableau5 : Listes des corpus annotés gratuits en arabe.

- **Error annotated corpora :** les corpus qui vont être présentés dans cette catégorie, porterons sur des aspects de correction concernant la conjugaison par exemple, des exercices corrigés etc. Zaghouani lui-même prépare un corpus de ce genre contenant plusieurs caractéristiques comme la prise en charge de deux langues, natives et non avec 2 millions de mots, la liste suivante représente 3 corpus déjà prêts :

Source	Corpus	Nbr Mots	Disponibilité
Habash et al	Qatar arabic language bank[43]	2,000,000	Oui
Alfifi et al	Arabic learner corpus[44]	282,000	Oui
Alkanhalet al	KACST error corrected corpus[45]	65,000	Non

Tableau6 : Listes des corpus corrigés gratuits en arabe.

- **Liste de mot :** Il en existe plusieurs y compris :

Source	Corpus	Nbr Mots	Disponibilité
Attia et al	Word count of modern standard Arabic[46]	1,000,000,000	Oui
Attia et al	Arabic wordlist for spellchecking[47]	9,000,000	Oui
Attia et al	Multiword expressions[48]	34,658	Oui
Attia et al	Arabic unknown words[49]	18,000	Oui
Zerrouki	Arabic Stop Words[50]	13,000	Oui
Attia et al	Obsolete arabic words[51]	8,400	Oui

Tableau7 : Listes des corpus de liste de mots gratuits en arabe.

- **Corpus Manuelle :** Les corpus de ce genre sont généralement très rares et surtout pour la langue arabe, globalement parce que ça demande du temps pour être conçu :

Source	Corpus	Nbr Fichiers	Disponibilité
Al Maadeed et al	QUWI handwriting dataset[52]	1,000	Non
Hassaine and maadeed	Writer identification contest for arabic script dataset[53]	200	Non
Al maadeed et al2	AHDB dataset[54]	100	Non
Al maadeed et al	ICDAR2011 competition dataset[55]	50	Non

Tableau8 : Listes des corpus manuellement conçus gratuits en arabe.

- **Corpus divers :** cette catégorie regroupe généralement des corpus qui sont spécialisés dans certains domaines, et ne peuvent pas être exploités dans d'autres, comme des corpus de Q&A(Question Answering) comme celui de Ben Ajiba et al(2007) et celui de détection de plagia de Bensalem et al[56].

Pour Conclure cette section, nous dirons que chaque corpus représente un outil de traitement spécifique à la catégorie pour laquelle il a été conçu. Certain corpus comme les textes bruts peuvent être utilisé dans plusieurs domaine du TAL, mais l'inverse n'est pas forcément vrai, quelques recherches nécessitent des annotations ou des corrections. Pour ce qui concerne les ressources de la langue arabe, le manque est très évidant, même les corpus présentés ne sont pas tous réellement disponibles, plusieurs liens sont morts et donc il y a moins de corpus librement exploitable que ce qui a été conçu.

4-Les Gestionnaires de corpus

4.1- Introduction

Avec l'évolution du domaine du TAL dans le monde et les travaux en cours, les gestionnaires de corpus et les outils de traitement sont de plus en plus nombreux. Un gestionnaire de corpus représente une consolidation des outils du TAL disponible, c'est un regroupement des fonctionnalités et traitements de base du domaine, ou autres d'un niveau avancé. La gestion de corpus peut aller du plus basique (création, ajout, calcul etc.) jusqu'aux fonctions les plus élaborés et spécifiques.

Parmi ses outils il y'à ce qu'on appelle dans le domaine le « calcul fréquentielle », ce qui représente tout ce qui est statistique et fréquence de mots ou d'apparitions, ou encore des outils de nettoyage comme la suppression des mots vide et la tokenization10. D'autres fonctions peuvent être mentionnées comme le calcul de similarité, analyse et POS.

Durant nos recherches, nous nous sommes basés sur trois plateforme pour arriver à concevoir un cadre de référence pouvant servir comme base pour notre travail. Il faut noter que les plateformes ou logiciel qui seront cité, ne sont qu'une partie de ce que l'univers du TAL a à nous offrir en termes d'outils. Les gestionnaires étudiés sont : SketchEngine11[57], TXM12[58] et CQPweb13[59].

4.2- Présentation des trois plateformes

- **SketchEngine :** Un gestionnaire de corpus disponible dans le web avec un payement. Comporte 500 corpus prêt à utiliser et prend en charge plus de 90 langues. Le site est utilisé par plusieurs personnes de plusieurs professions.
- **TXM :** C'est un logiciel de bureau (Windows, Mac et linux) gratuit avec un portail web disponible, qui représente un gestionnaire de corpus assez connu. Prend en charge plusieurs langues et propose beaucoup de fonctions.
- **CQPweb :** Représente une collection d'outil linguistique pour la gestion de corpus et le traitement de données. Une partie de CWB[14] et qui représente une interface graphique pour ces traitements.

4.3- Présentation des fonctionnalités

Pour pouvoir analyser les textes, il existe plusieurs fonctionnalités disponibles dont plusieurs catégories, notons les suivants :

- **Tokenization :** Étant donnée une séquence de mots définie, la tokenisation consiste à la découper en morceaux, appelés tokens représentant une liste de mots, tout en supprimant certains caractères comme la ponctuation et les caractères spéciaux, « m1 m2 m3 … mn » -> [« m1 », « m2 », … « mn »]
- **Lemmatisation :** Ce traitement consiste à appliquer aux mots un codage renvoyant à leur entrée lexicale commune ("forme canonique" enregistrée dans les dictionnaires de la langue, le plus couramment), que l'on désigne sous le terme de lemme, qui représente l'unité de base pour toute autre flexion.

[10] Segmentation des mots. Prendre un texte et le transformer en une liste de mots.
[11] https://www.sketchengine.eu
[12] http://textometrie.ens-lyon.fr/spip.php?article60
[13] https://cqpweb.lancs.ac.uk
[14] Corpus workbench: http://cwb.sourceforge.net

- **Stemming :** La racine d'un mot correspond à la partie du mot restante une fois que l'on a supprimé son (ses) préfixe(s) et suffixe(s), à savoir son radical pour étudier la morphologie des mots d'une langue et leurs flexions.
- **N-Grams :** est une sous-séquence de n éléments construite à partir d'une séquence donnée. L'idée semble provenir des travaux de Claude Shannon[15] en théorie de l'information. Son idée était que, à partir d'une séquence de lettres donnée (par exemple « par exemple ») il est possible d'obtenir la fonction de vraisemblance de l'apparition de la lettre suivante ainsi que d'autre applications. Les N-Grams sont utilisés largement dans les alignements de texte, le calcul de similarité, …
- **Synonymes :** Une fonction pouvant générer et proposer de synonymes à un mot donné en entrée en se basant sur le principe des termes dans un contexte. Selon la théorie de la sémantique distributionnelle [60], deux termes ont le même sens s'ils apparaissent dans le même contexte, un contexte représente un segment de deux séquences de termes, avec le mot de l'entrée au milieu.
- **Similarité textuelle :** Représente l'équivalence entre deux séquences de texte, que ce soit du point de vue syntaxique ou sémantique. Les différant métrique et méthodes utilisées seront détailler dans la suite de ce rapport.
- **Concordance :** Des séries d'expressions contenant un mot-clé particulier dans un passage ou une page Web, cette fonctionnalité permet de comprendre le contexte et l'utilisation du mot en question.
- **Catégorisation :** Dans chaque langue il y a des types de mots, comme les verbes, les noms, les adjectifs et autres selon les langues, cette fonction permet de donner une catégorisation des mots, connu sous le nom de Part of speech.
- **Nombre d'apparition :** Un calcul de nombre d'occurrence d'un terme (ou une séquence de termes) dans un texte. Pet être utiliser dans les traitements qui nécessitent un score ou un seuil minimum d'apparition, ou même pour les modèles probabilistes.
- **Traduction :** La traduction est la communication du sens d'une langue à une autre langue. La traduction fait référence à des informations écrites, tandis que l'interprétation fait référence à des informations parlées.
- **Tf-IDF :** Le TF-IDF est une méthode de pondération souvent utilisée en recherche d'information et en particulier dans la fouille de textes. Cette mesure statistique permet d'évaluer l'importance d'un terme contenu dans un document, relativement à une collection ou un corpus.

Les fonctions présentées sont des caractéristiques des plateformes étudiés, et représentent les fonctions de base, tout autre traitement passera par l'une ou plusieurs des fonctions présentées pour arriver au résultat.

4.4- Analyse et comparaison

Les tableaux suivants mettront en évidence les différences et les caractéristiques qu'on a pu identifier de chaque plateforme des 3 mentionnés

[15] https://fr.wikipedia.org/wiki/Claude_Shannon

- **Corpus :**

Gest./Fnct	Création	Enrichissement [16]	Téléchargement [17]	Prêt [18]	Importation [19]
SketchEngine	Oui (web)	Non	Oui	Oui	Oui
TXM	Oui(web)	Non	Non	Non	Oui
CQPweb	Non	Non	Non	Oui	Oui

Tableau9 : Comparaison entre les corpus des plateformes étudiés.

Note : Les donnée dans CQPweb doivent être préparer d'une façon qui nécessite des connaissances avancées en TAL, alors que SketchEngine et TXM proposent plusieurs format brute et simple (.txt .html. .Csv etc.).

- **Donnée en sortie :** Les donnée des trois plateformes peuvent être exporter selon le résultat obtenu, et avec le nombre souhaité. Par contre, TXM n'offre que le format .csv et CQPweb quant à lui que du texte brut, il n'y a que SketchEngine qui offre les deux choix.
- **Calcul fréquentiel et traitement du texte :** Représente un groupe de fonctionnalités permettant de traiter le texte d'entrée (tokenizer, lemmatiser et nettoyage) ainsi que la possibilité de calculer le nombre apparition des mots ou de passages et d'autre traitements qui génèrent des formes de texte comme les n-grams.

Fonctions	SketchEngine	TXM	CQPweb
Tokenization	Oui	Oui	Oui
Stemming	Oui	Oui	Oui
N-Gram	Oui	Oui	NA
Synonyme	Oui	Non	Non
Similarité	Non	Non	Non
Concordance	Oui	Oui	Oui
Nombre d'apparition	Oui(non direct)	Oui	Non
Traduction	Oui(quelque langues)	Non	Non
Prise en charge de l'arabe	Oui	Oui	Oui
Plateforme	Web	Desktop	Web
Moyen d'accès	Payant (après 30j)	Gratuit	Payant

Tableau10 : Comparaison entre les différentes fonctionnalités des plateformes étudiés.

[16] Possibilité d'extension d'un corpus avec plus de données textuelles
[17] Possibilité d'exporter le corpus.
[18] Avoir des corpus prêts à l'utilisation
[19] Possibilité d'importer des fichiers locaux pour les assembler en un corpus

4.5- Conclusion

Les 3 plateformes présentent bel et bien des avantages, mais pas sans inconvénients car chaque plateforme à ses limitations qui sont relatives à chaque besoin et aussi à chaque utilisateur :

- Pas assez de fonctionnalités d'analyse.
- Pas de fonctionnalité d'analyse sémantique (similarité, représentation vectorielle …)
- Accès non gratuit pour SketchEngine et CQPWeb.
- TXM est une application de bureau ce qui rajoute l'étape de l'installation.

Certainement, il faut noter que les trois présente un bon nombre de fonctionnalités et de ressources ainsi que la prise en charge de l'arabe pour toutes les fonctionnalités sauf la traduction qui reste un problème concernant cette langue.

Pour ce qui est de l'assistance, ils proposent des manuelles et des vidéos de tutoriels ainsi que des Q&A disponibles à tout moment pour les utilisateurs. Le choix portera surtout sur le besoin, et le niveau de l'utilisateur.

5- Similarité Textuelle

5.1- Introduction

La similarité textuelle représente une comparaison de deux (ou plusieurs) textes et en sortir avec un tôt de similitude entre les deux, qui servira comme donnée d'entrées pour autres recherches comme :

- Les moteurs de recherche doivent modéliser la pertinence d'un document par rapport à une requête, au-delà du chevauchement de mots entre les deux.
- Les sites de questions-réponses tels que Quora ou Stackoverflow doivent déterminer si une question a déjà été posée auparavant.
- Dans les services clients, le système d'IA doit être capable de comprendre les requêtes sémantiquement similaires des utilisateurs et de fournir une réponse uniforme.
- Dans les systèmes ASAG20 utilisent ce genre de calcul pour déterminer le degré de similitude entre la réponse modèle et celle de l'étudiant en question
- Les systèmes de recommandations peuvent aussi utiliser la similarité textuelle pour proposer du contenu approprié et en relation avec ce que l'utilisateur a déjà cherché.
- En matière juridique, la tâche de similarité de texte permet d'atténuer les risques sur un nouveau contrat, en partant de l'hypothèse que si un nouveau contrat est similaire à un existant qui s'est avéré résilient, le risque que ce nouveau contrat soit à l'origine de la perte est minimisée.

Vue son importance dans le domaine du TAL, nous prévoyons d'intégrer la fonctionnalité de calcul de similarité. Par conséquent, nous fournissons dans la suite une étude sur les approches de similarité.

[20] Système automatique d'évaluation de réponse courtes

5.2-Calcul de similarité

Pour le faire, il est obligatoire de passé par l'étape du « prétraitement » afin de pourvoir nettoyer le texte et l'analyser. Ensuite, il faut trouver une représentation vectorielle ou ontologique selon la méthode utilisée qui servira de base de calcul pour chaque méthode. Au final, un modèle mathématique (équation) est posé pour le calcul du taux de similarité entre les deux textes en question. La figure suivante montre deux types de similarité textuelle :

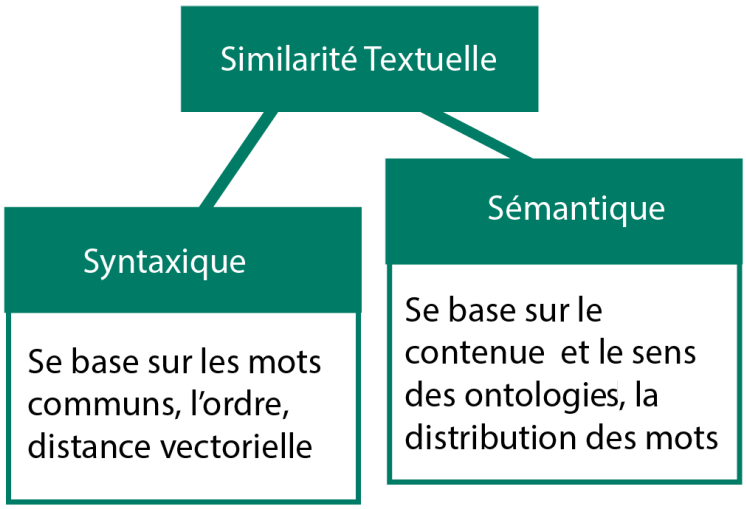

Figure2 : Représentation des types de similarité

5.2.1-Similarité Syntaxique :

En principe, cette approche consiste à calculer la ressemblance entre deux chaines de caractères. Par exemple, « قيادة »et « قلادة » peuvent être très similaire (une seule lettre les sépare) à l'aide de certain algorithme.

Il existe plusieurs mesures qui se basent sur cette approche, mais d'abords il faut passer par un « Modèle d'espace Vectoriel » afin de faciliter l'exploitation des documents et de réduire leur complexité. Pour s'y faire, en passera par la phase du prétraitement pour ensuite avoir un texte représenté de façon vectoriel, en éliminant les mots non pertinents, le poids des termes sera calculé aussi et est très important.

- **Distance Euclidienne :** Pour représenter la similarité entre deux documents, cette méthode considère la distance entre les deux représentations vectorielles dans un espace donné. Définie par l'équation suivante :

$$d(X,Y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$

Figure3 : Formule de la distance Euclidienne

X : Document 1 *yi,xi: mots des documents*
Y : Document 2 *n: taille du vecteur*

- **Cosinus :** Cette méthode de calcul de similarité [61] se base sur l'angle entre deux document (ou leur représentation vectorielle respectives). Définie par l'équation suivante :

$$\mathrm{SimCos(X,\ Y)} = \cos\theta = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

Figure4 : Formule cosinus

Les résultats sont définis dans [-1,1], par contre pour la ressemblance entre deux textes, les valeurs compris dans [-1,0[seront rejeter, le 0 indiquera la non similarité et les valeurs dans]0,1] seront considérer comme étant le taux de similarité entre X et Y qui sont les deux documents en question.

- **Coefficient de Pearson [62]:** Le principe de cette méthode est le même que celui de la similarité cosinus, d'ailleurs, elle utilise la même équation pour arriver au résultat. Par contre, le coefficient de Pearson consiste à soustraire la moyenne de chaque document à sa représentation vectorielle en passant par chaque case individuellement. Ce qui donne :

$$X' = X - Moyenne(X)$$
$$Y' = Y - Moyenne(Y)$$

Après avoir eu les nouveaux vecteurs des deux documents en question, en pourra ensuite procéder au calcul avec la formule classique du Cosinus *SimCos(X', Y').*

- **Coefficient de Jaccard [63]:** L'indice de Jaccard calcule la similarité entre deux textes en se basant sur les tailles des ensembles « union » et « intersection » de deux documents. Il est donc possible de calculer la similarité de deux textes en se basant sur les mots qu'il nous en commun relativement aux mots non communs, sans passer par la représentation vectorielle cette fois ci, mais il existe quand même une formule pour la représentation vectorielle comme les mesures citées avant. Ce qui donne la formule suivante :

$$\mathrm{SimJaccard} = \| \mathrm{Intersection(D1,D2)} \| / \| \mathrm{Union(D1,D2)} \|$$

- **Distance de Levenshtein[64]:** Cette méthode est très simple, elle permet de calculer la similarité en passant par la modification du document D1 selon D2 avec leur représentation en chaine de caractère. Ce qui veut dire :
    1. Si D1[i] est différent de D2[i] alors : D1[i] <- D2[i] (modification).
    2. Si D1[i] est blanc ou vide et que D2[i] ne l'est pas alors dans ce cas-là: D1[i] <- D2[i] (ajout).
    3. Si D2[i] est vide ou blanc et que D1[i] ne l'est pas, alors dans ce cas-là: D[1] <- blanc (suppression).

Ces trois conditions de modification, permettent donc de rendre D1 similaire à D2 en comparant leurs caractères. Chaque modification à un cout égale à 1, sauf

pour D1[i] = D2[i] puisque rien n'est modifié. Plus le cout de cette opération est petit, plus la similarité est grande.

- **Similarité de Dice[65]:** La similarité calculée avec l'indice de Dice, est très similaire à celle de **Jaccard,** en revanche celle-ci se base sur les mots similaires (uniquement l'intersection) de deux documents, par rapport au nombre totale de terme. Avec la formule suivante :

$$SimDice(D1, D2) = (2*nC) / (|D1|+|D2|)$$

nC: Nombre de terme en commun.
|D1|,|D2|: nombre totale de terme dans D1 et D2 respectivement.

5.2.2-Similarité Sémantique :

Cette approche de calcul de similarité, est très différente de la précédente. La similarité sémantique se base sur le sens des termes d'un document. Donc pour faire le calcul, ces métriques se basent sur le calcul de ressemblance de leur « signification ». Pour dire que deux concepts sont sémantiquement similaires (même signification) il faut qu'il ait une relation de synonymie entre eux, ou encore qu'il soit tous les deux antonymes à un autre concept pour qu'il soit considéré équivalent en termes de sens. Pour cette approche, il existe plusieurs métriques, qui se basent eu même sur des approches différentes :

- Basé connaissance.
- Basé Corpus

1. Approche basé connaissance (topologique):

Principalement basé sur les « taxonomies » qui représente des branches ou des arbres qui représentent un mot, par exemple un chat est un animal, un animal est un être vivant et ainsi de suite. Donc chaque mot est représenté de façon arborescente basé généralement sur la base de connaissance de WordNet [66], qui regroupe les taxonomies et ainsi les concepts des mots. Les approches qui traitent la similarité sémantique avec cet outil, se base sur le les arcs, permettant d'arriver de C1 à C2 (C pour concept). La figure suivante montre un schéma explicatif :

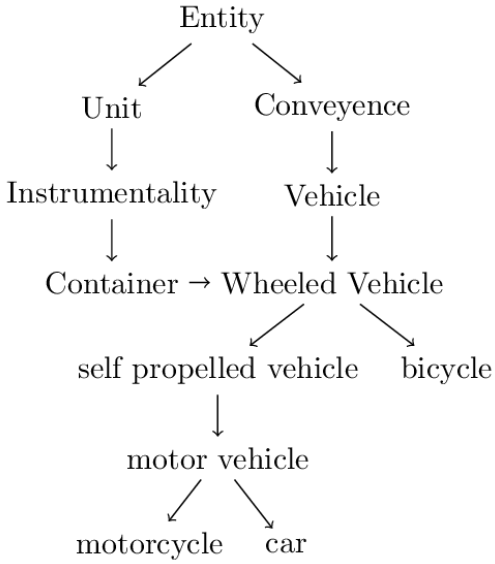

Figure5 : schéma explicatif d'une arborescence WordNet

- **Leacock et Chodorow [67]:** Ils ont considéré la similarité de deux mots en se basant sur leur hiérarchie de leur hyponymie. Donc pour s'y faire, nous calculons le plus court chemin entre deux mots, ce qui signifie que les arcs les plus bas dans l'arborescence sont les plus similaires. La formule est donnée par :

  *SimLCH = -log (Longueur (W1, W2) / 2*D)*

  Longeur(w1, w2): Représente le plus court chemin entre les deux mots(nombre de nœuds), D: Hauteur maximal de la taxonomie.
- **Wu et Palmer [68]:** Cette métrique estime la similarité en se base sur deux calculs dans WorNet. En premier, elle détermine le plus bas ancêtre des deux mots. Ensuite, elle calcule la profondeur des deux mots respectivement. Ce qui donc veut dire, que cette méthode calcule la similarité en calculant la distance des deux mots par rapport à leur ancêtre commun le plus bas. Avec la formule suivante :

  SimWuPal = (2*Profondeur (LCS)) / (Profondeur(W1) + Profondeur(W2))

  LCS : lowest common submer (plus bas ancêtre).

Il y a aussi des approches sont aussi appelé « basé information » étant donné que les nœuds représentent les informations. La similarité est simplement la probabilité que le nœuds (super class) qui donne le concept W1 (avec une distance donnée) soient le même que celui de W2, c.à.d qu'il partage des informations en commun et donc une signification très proche. Cette probabilité du contenue d'information (IC) est donnée par « IC(c) = −logP(c) » où « P (c) » est la probabilité de rencontrer une instance du concept c. Et donc les approches suivantes se base sur cette formule de probabilité :

- **Resnik:** Cette mesure proposée en 1995[69], retourne directement l'IC du plus bas ancêtre commun entre deux mots « SimRes(W1, W2) = IC(LCS(W1, W2)) = −logP (LCS(W1, W2)) » Deux autres mesures existent, qui sont principalement basé sur celle de Resnik, mais qui représente une normalisation de cette dernière :
  2. **Lin [70]:** SimLin (c1, c2) = (2∗IC(LCSc1,c2)) / (IC(W1)+ID(W2))
  3. **Jiang and Conrath[71]:**SimJC = 1 / (IC(W1) +IC(W2)-2IC (LCS (W1, W2)))

2- Approche basé Corpus (Statistiques) :

Le principe de cette technique, est de calculer la sémantique d'un terme, en se basant sur ses apparitions devant d'autre mots dans les documents. Donc pour faire simple, si deux mot M1 et M2 apparaissent souvent devant les mêmes mots, ça voudra dire qu'ils ont la même utilisation dans les documents étudiés, et donc peuvent être considérer comme très similaire.

Pour s'y faire, chaque mot se voit attribuer un vecteur propre à lui, regroupant le nombre d'occurrences de ce dernier avec chaque mot du deuxième document, nous parle ici de contexte. Le calcul de similarité de deux mots se fait à partir de la distance des deux vecteurs obtenus comme pour la similarité syntaxique, sauf que dans ce cas nous calculons la distance de deux sens.

- **Bi-Clustering [72] :** Cette approche est une méthode de classification non supervisée qui permet de diviser les lignes et les colonnes d'une matrice. Il existe plusieurs types de Bi-Clustering, ou plusieurs représentations pour être plus précis, notablement les représentations graphiques à base d'arbres, dans ce cas-là nous nous intéressons à la représentation vectorielle. Le double clustering matricielle permet

de calculer la similarité entre deux documents en se basant sur une fonction de terme en commun calculer pour chaque couple de document dans la matrice « Document x terme », avec les mots qui ont en commun. Donc par exemple :

M	W1	W2	W3	W4
D1	1	1	0	0
D2	0	0	1	1
D3	0	1	1	0

Tableau11 : Exemple d'une Matrice document x termes du Be-Clustering.

À première vue, D1 et D2 n'ont aucun mot en commun selon la matrice M, aussi, la matrice montre que D2 et D3 ont W3 en commun et W2 est commun entre D1 et D3. Donc, nous peut dire qu'il y a une certaine similitude entre W2 et W3. Donc le Bi-Clustering permettra de donner une certaine similitude entre D1 et D2, après la segmentation des matrice Document et terme en commun respectivement, contrairement à une mesure syntaxique classique qui aura pour résultat une distance maximal est donc impliquera la non similarité des deux documents.

- **Analyse sémantique explicite (ESA):** ESA [73] est une méthode de représentation vectorielle qui se base sur Wikipédia comme base de connaissances. En principe, chaque mot d'un vecteur de document, est représenté avec son « TF-IDF » depuis son article dans Wikipédia. Cette mesure se base sur le fait que les article de Wikipédia soient orthogonaux, en revanche, l'ESA donne de meilleurs résultats quand elle est basée sur un autre corpus qui ne l'est pas.

5.3-Similarité des textes arabes

La langue pose des problèmes quant au calcul de similarité de deux textes rédiger en arabe, comme la grande flexion et variété des termes ou encore le fait qu'il existe plusieurs mots en arabe toujours pas connus et qui peuvent avoir plusieurs significations. Contrairement à la similarité syntaxique, les techniques citées avant pour le calcul de similarité sémantique ne sont pas toutes applicable à la langue arabe à cause de la nature morphologique et sémantique dont elle dispose. Voici quelques travaux dans le domaine :

- **Soori et al[74] :** Ils divisent un document en paragraphes, puis créent un dictionnaire pour chaque paragraphe sans supprimer les mots vides. Cependant, ils considèrent les paragraphes pour la similitude du texte car le nombre de mots contenus dans un paragraphe ne doit pas être affecté par les mots vides.
- **Al-Ramahi et Mustafa [75] :** ont appliqué trois techniques N-grams utilisées pour mesurer la similitude du transfert de crédits de cours entre les universités. Les trois techniques étaient basées sur des mots avec la mesure de similitude des dice.
- **Hussien [76 & 77] :** a étudié et visualisé l'analyse de similarité entre les documents arabes pour la détection du plagiat. Il a essayé de modéliser la relation entre les documents et les phrases n-gramme générées à partir de ces documents. Un algorithme heuristique d'appariement par paire est utilisé pour construire le modèle tandis que l'analyse sémantique latente LSA est utilisée pour étudier les relations cachées entre les documents et leurs phrases de construction.
- **Selamat et Ismail [78] :** Ils analysent la similitude des nouvelles des textes traduits en arabe et en anglais en appliquant deux techniques non supervisées ; Carte d'auto-organisation (SOM) et Carte d'auto-organisation hiérarchique croissante (GHSOM), puis

comparaison des résultats pour les deux techniques. SOM est l'un des modèles de réseau neuronal utilisé pour cartographier des données de haute dimension.

6- Distribution des mots : Approche computationnelle

6.1-Introduction

La distribution des mots est un modèle de représentation vectoriel des mots selon leurs concept et signification en se basant sur leurs poids dans les documents étudiés. Ce genre de modèle contribue de façon directe dans plusieurs domaines du TAL comme le calcul des synonymes, la similarité entre mots ou phrases, ou encore dans notre cas la génération des espaces sémantiques des textes d'entrés, ce qui représente une collection de vecteurs sémantique des mots disponible après normalisation bien évidemment.

La TAL essaye toujours de faire comprendre notre langage naturel à la machine, et trouvé une représentation facilement assimilée par la machine et le but principal et la seule façon pour pouvoir y arriver, car cette dernière ne bénéficie pas de sens de référencement de l'être humain, mais plutôt d'une logique mathématique et raisonnement à base de données.

6.2-Les espaces sémantiques

6.2.1- Présentation des espaces sémantique

Les éléments comparés dans l'analyse sémantique sont définis dans un espace sémantique multidimensionnelle, un mot est considéré comme un point dans un espace multidimensionnel représentant la diversité du vocabulaire utilisé ou plus généralement les différents contextes utilisés pour représenter un mot.

Deux mots sont donc comparés quant à leur emplacement dans cet espace multidimensionnel, les dimensions sont définies par les contextes utilisé pour construire le modèle de distribution. Les mots sont représentés par leurs vecteurs correspondants dans la matrice et sont donc comparé grâce à des mesures.

Dans le domaine du traitement automatique de la langue, les espaces sémantiques représentent un moyen pour capturer le sens sémantique d'une phrase donnée. Les deux challenges posés dans le TAL sont l'incohérence du vocabulaire (le fait qu'on peut exprimer la même chose différemment) et l'ambiguïté (le fait qu'un terme donné peut avoir plusieurs significations). La suite de cette section présentera quelques modèles d'espace sémantique utilisés.

6.2.2- LSA (Latent semantic analysis)

Aussi connus sous le nom de LSI ou l'analyse sémantique latente, LSA [79] est une matrice de distribution des mots sémantiquement reliés grâce à leurs concordances et apparitions. Cette approche consiste à créer des relations entre les mots des documents étudiés et les concepts qu'ils reflètent selon les contextes dont les mots sont employés dans le corpus. LSA est utilisé dans plusieurs domaines du TAL comme l'extraction des concepts des mots, recherche d'information grâce aux poids des mots importants dans les requêtes et les documents disponibles pour en tirer les plus pertinents ou encore calculer la similarité avec le principe de la distribution des contextes dont les groupes de mots sont utilisés. LSA est aussi utilisée dans les espaces

sémantiques car elle représente un modèle de « sac de mots »[21]. C'est-à-dire que la matrice générée attribut à chaque mot du corpus un vecteur reflétant sa position par rapport aux contextes de l'espace vectoriel, pour y arriver, la matrice initial Mots*Documents sera décomposée en trois sous matrices dont le produit est égal à l'initial, avec la première matrice comportant les termes et leurs distributions selon les contextes, la figure suivante illustre le mécanisme d'LSA :

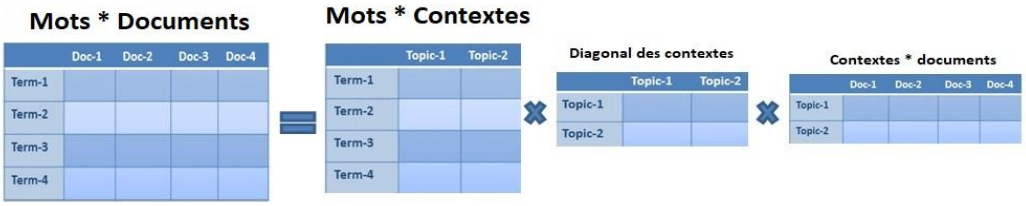

Figure6 : modèle représentatif d'LSA

6.2.3- PLSA (Probabilistic Latent semantic analysis) :

PLSA [80] est une variante de LSA, qui introduit le principe de probabilité. En principe, les documents d'une collection sont représentés comme étant un groupe de thème ayant chacun un modèle probabiliste propre, permettant de générer des mots précis (mots clés par exemple). Chaque document D[i] de la collection de documents Ds est alors supposé avoir été généré par un mélange pondéré des modèles latents des thèmes. Donc la probabilité qu'un document représenté par un groupe de mots C = {C1, …, Cn}.

6.2.4- COALS [81] :

Coals est un modèle de contexte qui parcourt les données mot par mots. Premièrement COALS utilise un schéma de pondération pour pondérer les cooccurrences en fonction de leur distance par rapport au mot cible dans le corpus, il supprime la distinction de cas des mots contextuels gauche / droite, traitant toutes les statistiques de cooccurrence dans la fenêtre de la même manière. Deuxièmement, afin de réduire l'influence de cooccurrences avec des mots de haute fréquence, COALS emploie une association lexicale qui est une fonction basée sur la corrélation de Pearson. Comme les poids négatifs contiennent peu d'informations sur la relation sémantique, les corrélations négatives sont mises à 0. Les corrélations positives sont ensuite transformées en appliquant la racine carrée. Les résultats démontrent que COALS fournit de meilleurs comptes rendus de similarité sémantique par rapport à LSA et WordNet.

La fonction d'association lexicale employée par COALS est particulièrement sensible à la covariation entre les mots liée à la sémantique similitude plutôt que des cooccurrences. Les performances de clustering observées [81] suggère qu'un modèle cognitif réussi devrait accorder une attention à des corrélations distributionnelles lors de l'apprentissage, et loin des cooccurrences fortuites et anti corrélations. De plus, le succès de l'algorithme COALS (au moins dans le domaine de catégorisation sémantique) contraint l'ensemble des mécanismes cognitifs plausibles qui peuvent être

[21] Un mot reflète plusieurs autres mots dans le même contexte, le groupe de mots est appelé sac de mots.

utilisé par les humains lors du codage des informations de distribution à partir de l'environnement linguistique.

6.2.5- Word Embedding [82] :

Ce modèle désigne un ensemble de techniques d'apprentissage automatique permettant de donner une représentation vectorielle aux mots d'un corpus, constituée de nombres réel. Ce modèle repose sur la théorie de distributions de contextes qui considère que les mots sont caractérisés par leurs contextes et ainsi la façon dont ils sont utilisés et donc des mots qui partagent des contextes similaires partagent également des significations similaires. Les algorithmes de Word Embedding sont le plus souvent employés pour décrire des mots à travers de vecteurs numériques, mais ils peuvent également être utilisés pour construire des représentations vectorielles de phrases entières, de données biologiques comme les séquence d'ADN, ou encore des réseaux représentés comme des graphes.

Il existe plusieurs approches de Word Embedding. Les premières remontent aux années 1960et reposent sur des méthodes de réduction de dimensionnalité. Plus récemment, de nouvelles techniques basées sur des modèles probabilistes et des réseaux de neurones, comme Word2Vec[83] de google ont permis d'obtenir de meilleures performances.

L'idée générale est de projeter un ensemble de mots d'un vocabulaire de taille « V » dans un espace vectoriel continu où les vecteurs de ces mots ont une taille « n » relativement petite. De plus, on veut trouver une représentation vectorielle de chaque mot « V » de façon que les mots aux représentations voisines apparaissent dans des contextes similaires. Mikolov[84] propose deux approches à cette méthode qui se base sur un entrainement d'un raison de neurone à partir de corpus assez volumineux :

- **SkipGram :** Cette approche consiste à projeter un vecteur d'un mot donné dans l'espace vectoriel et calculer l'ensemble des vecteurs dans cet espace dont la similarité est proche de ce dernier, c'est-à-dire que les mots retournés représentent les mots qui sont du même contexte que le mot d'entrée et donc ce modèle pourra prédire une phrase entière à partir d'un mot, la figure suivante représente un schéma explicatif :

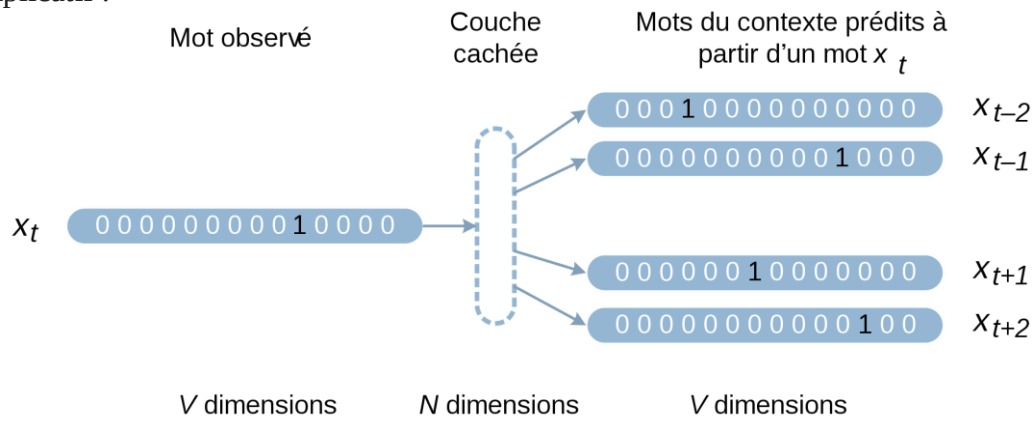

Figure7 : Modèle Skipgram

- **Continuous Bag of Words :** Cette approche est exactement l'inverse de SkipGram. Au lieu de prendre comme entrée un mot est prédire ses voisins dans le contexte, ce modèle un ensemble de vecteurs de mots dans un contexte et retourne un mot prédit à partir de ce dernier :

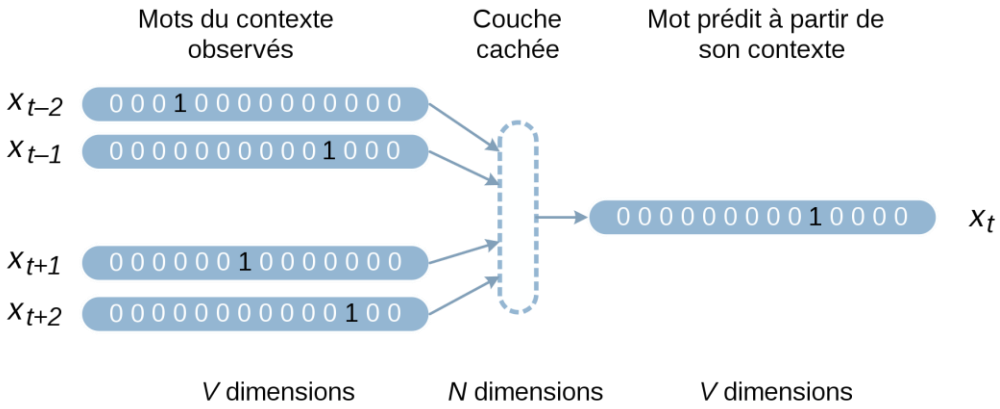

Figure8 : Modèle Continuous bag of words

7- Conclusion

Ce que le domaine offre en termes de corpus et d'analyse textuelle est très vaste et montre le potentiel du TAL arabe et global. Nous avons donc fini d'étudier les différents aspects de notre problématique ainsi que les techniques utilisées. Nous avons aussi pus poser un cadre de référence pour procéder à la réalisation de notre plateforme. La recherche a permit de mieux cerner les chapitres et les utilisations.

Les espaces sémantiques présentent un outil très important de tout ce qui touche à la sémantique en TAL. En effet, l'utilisation de ces approches améliore considérablement le calcul des similarités, d'indexation ou encore la classification des documents, des tâches quotidiennes des moteurs de recherches par exemple. Ces approches basées sur des distributions des mots, permet d'analyser plus de langue comme l'arabe dont le WordNet n'est pas encore satisfaisant et c'est pour cette raison que nous l'intégrons dans notre projet.

Chapitre 2 :
Conception et Réalisation.

Ce chapitre représente notre conception de la plateforme ainsi que la réalisation et implémentation des différents fonctions :

- Introduction
- Spécification des besoins de la plateforme.
- Schéma générale de l'architecture fonctionnelle du système développé.
- Développement des Fonctionnalités retenues.
- Scénarios d'utilisation.
- Réalisation

1- Introduction

Ce chapitre présente l'architecture générale et la conception de notre plateforme. La recherche menée avant dans ce rapport nous a permis de mieux comprendre les plateformes du même genre que là notre ainsi que les utilisateurs qui peuvent s'y intéresser selon les besoins du TAL. Ce chapitre présentera notre schéma général de l'application ainsi que les différentes fonctions que nous avons choisis d'inclure.

2-Architecture fonctionnelle

Dans le cadre de notre travail, nous développons une plateforme qui consolidera les besoins du TAL dans un seul endroit et offrira aux utilisateurs les outils nécessaires en commençant par la base qui est la génération de corpus (collection de données textuelles) ainsi que les différents analyses et études des données.

2.1- Acteurs du système

- **Les utilisateurs :** L'utilisateur est l'acteur principale de notre plateforme, c'est l'acteur qui aura accès à la gestion des corpus et aux différentes fonctionnalisées
- **L'administrateur** : représente le rôle de gestionnaire globale de la plateforme et des utilisateurs ainsi que tout ce qui touche aux fonctionnalités de la plateforme.

2.2- Fonctionnalité retenues

Les fonctionnalités retenues dans notre plateforme sont les suivant :

- **Gestion des corpus :**
  1. Création : créer des corpus depuis des mots clés.
  2. Enrichissement : étendre les corpus avec plus de données.
  3. Exportation : exporter les corpus générés vers l'utilisateur.
  4. Importation : créer un corpus avec des fichiers textes locaux.
  5. Suppression : supprimer un corpus de la liste acquise par l'utilisateur.
- **Analyse textuelle :**
  1. Exploration : étudier les mots et leurs utilisations.
  2. Analyse morphologique : analyse des flexions des mots.
  3. Espace vectoriel : généré des représentations vectorielles.
  4. Similarité textuelle : Analyse de similarité entre des couples de textes.
  5. Exportation : chaque résultat obtenu est exportable.

Les fonctionnalités sont décrites dans le schéma suivant :

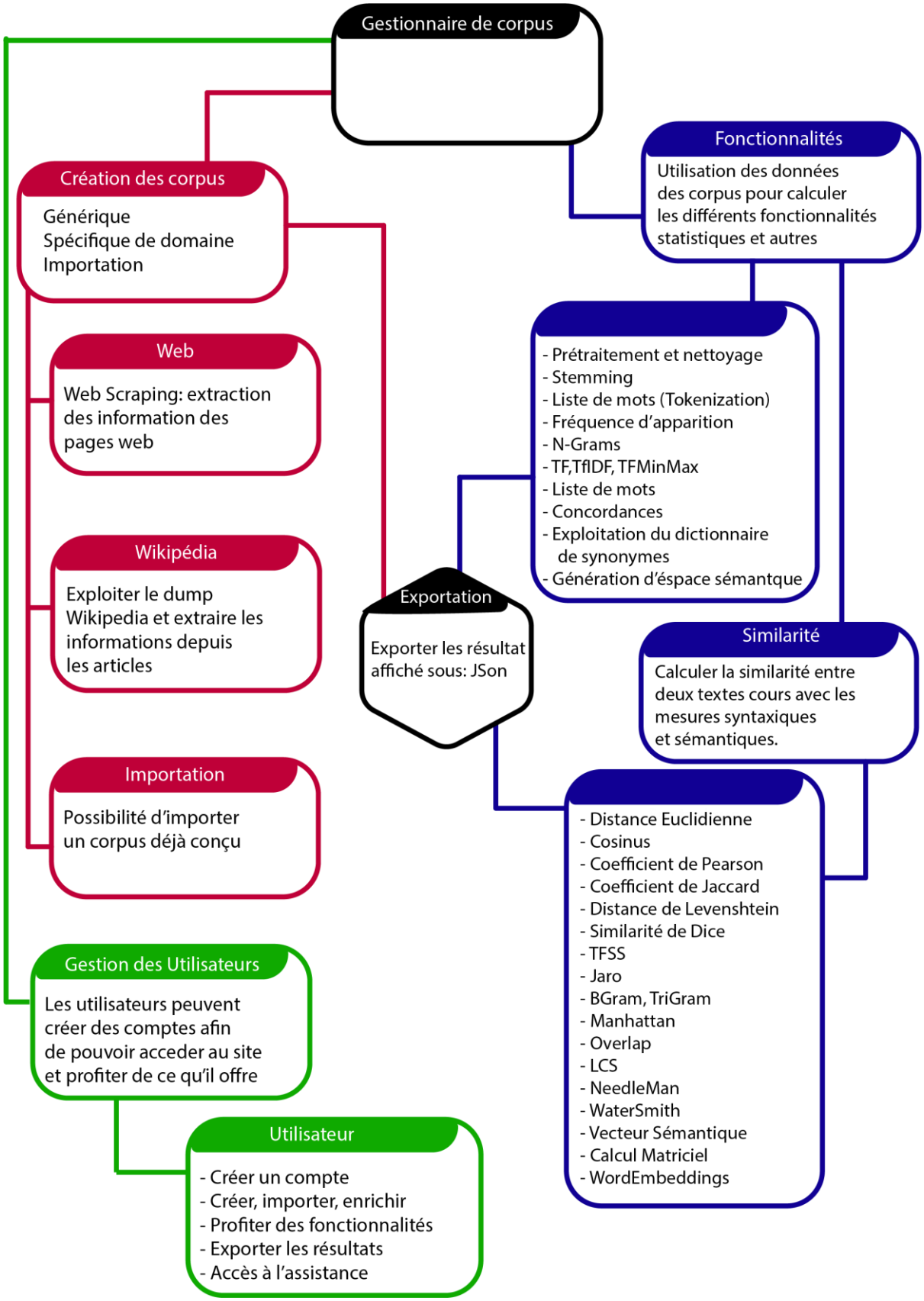

Figure9: Architecture fonctionnelle de la plateforme

3- Développement des Fonctionnalités retenues

3.1- Introduction

Le travail réalisé prend en considération tout ce qui a été abordé dans le deuxième chapitre « l'Etat de l'art ». Une consolidation des fonctionnalité et analyses textuelle du domaine du TAL représentant le « ready to use » de toutes les données souhaitées, c'est-à-dire que les utilisateurs pourront analyser la langue dans tous les côtés depuis la création des données textuelle jusqu'à l'exportation des résultats obtenus d'une panoplie de traitement offert, tout en respectant aussi la problématique ainsi que les objectifs posés au début du travail.

3.2- Gestion des Corpus

3.2.1- Création des corpus

Les corpus font partie des premiers point à réaliser dans notre travail, mais aussi le début de tous les ressources nécessaires dans le domaine du TAL. Nous avons pris en considération les types des corpus et les besoins des domaines suivants :

- **Spécifique :** Un corpus contenant une collection de documents dont la relation est proche d'un domaine décrit par un ensemble de mot clés par l'utilisateur au moment de la génération.
- **Multi-Domaine** : Un corpus contenant plusieurs corpus spécifiques dont les documents sont tirés à partir d'une liste de mots clés également mais cette avec une relation entre eux de n-gram, pour garantir l'intégrité du corpus en termes de diversité de documents.
- **Générique** : Ce type de corpus représente une collection de document qui n'a pas forcément de relation explicitement indiquée ou un domaine précis, c'est-à-dire que les corpus de ce type peuvent contenir tous les documents que le système croisera durant son processus de génération.

Pour la génération des corpus, nous avons exploité plusieurs approches pour les deux types. Les deux premières méthodes se penchent directement vers le web pour faire le « Scrapping[22] » et la langue du corpus à générer se fait à partir des mots, le système utilise un détecteur de langues intégré pour faciliter la tâche et rendre le processus mois compliquer à utiliser. La troisième utilise les fichiers locaux de l'utilisateur pour les importer et les utiliser comme corpus déjà construit manuellement ou simplement pour l'analyser à l'avenir. Les sources d'extraction de donnée sont comme suit :

- **Dump Wikipédia :** En utilisant les ressources de Wikipédia et leurs API[23] déjà prêt, nous avons construit les corpus à partir des articles rédigés. C'est-à-dire que notre corpus est constitué d'une collection d'articles relatifs au domaine décrit par les mots clés entrés par l'utilisateur. A partir de cet ensemble de mots clés (qui peut contenir un seul mot), nous utilisons la fonction incluse dans la package Wikipédia permettant de générer de nouvelles combinaisons de mots relative à celle d'entrée dont le nombre de résultats retourné est aussi paramétrable dans la même fonction,

[22] Extraction des données du web à partir des liens.
[23] Wikipedia Documentation — wikipedia 0.9 documentation

et donc avoir plus de résultats de recherche dans les serveurs. Cette méthode permet de parcourir plus d'articles et donc avoir plus de documents à extraire et à intégrer dans le corpus final. Pour les corpus multi domaine, chaque combinaison de mots servira d'entrée elle aussi pour cette méthode pour en extraire le plus de possibilités de recherches également. La « figure10 » est une illustration du processus :

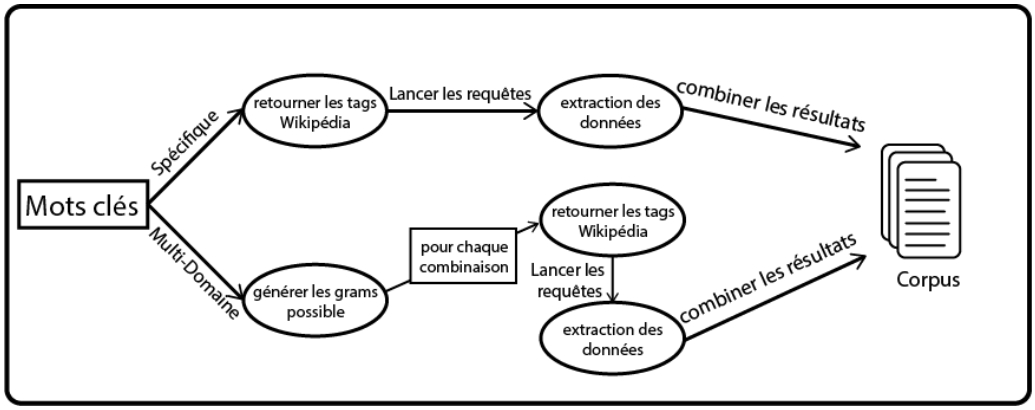

Figure10 : Processus de génération de corpus avec Wikipédia

- **Moteur de recherche Google[24] :** Google est un moteur de recherche très utilisé et offre un nombre de document pratiquement incalculable. En utilisant des méthodes de web scrapping disponible dans la librairie « Beautifulsoup[25] » de python qui permet de parcourir des page html depuis le web et d'extraire les donnée des balises comme les paragraphes dans notre cas <p>, nous lançons des requêtes de recherches depuis le moteur à partir de multiple combinaisons de trois mots pour chacune, réalisées avec la méthode des n-gram[26] de « NLTK[27] », pour retourner tous les liens proposés par google pour chaque combinaison et les utiliser comme source afin d'extraire leurs données textuelles. Pour les corpus multi domaine les requête sont lancé n fois pour n = nombre de gram disponible, tandis que pour les corpus spécifique la requête est lancée directement depuis le moteur de recherche comme entrée par l'utilisateur. Comme google n'a plus d'API disponible pour ce genre de processus, nous avons lancé les requêtes directement en modifiant la chaine de caractère avec les espaces remplacé par des « + », « la figure11 » explique le processus :

[24] https://www.google.com
[25] https://pypi.org/project/beautifulsoup4/
[26] Suites de mots de taille n
[27] https://www.nltk.org/

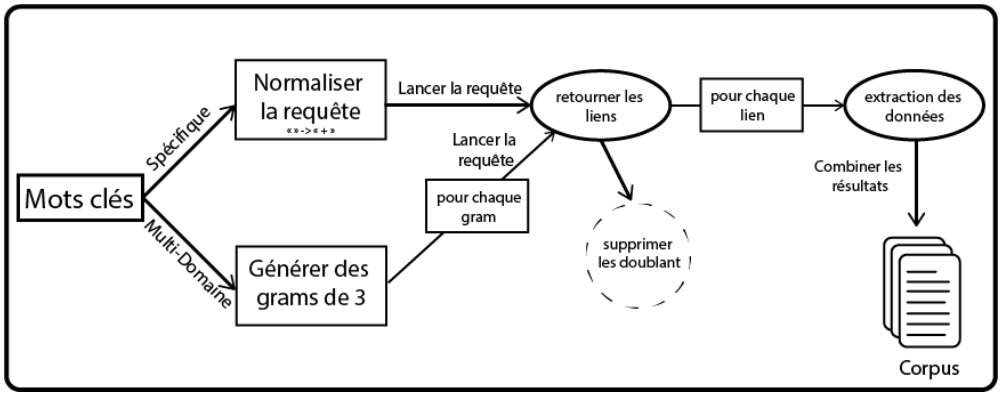

Figure11 : Processus de génération de corpus avec Google

- **Liens directs :** L'utilisateur peut également fournir des liens directs afin d'extraire les informations incluses pour construire son corpus. La façon dont notre système procède est la même que pour le corpus qui prend comme source le moteur de recherche google, c'est-à-dire que pour chaque lien nous lançons une requête vers ce lien depuis un script python et ensuite extraire toutes les données textuelles fournis par le lien grâce à la librairie de « Beautifulsoup ». Les liens vides qui n'ont pas retourner de résultats seront rejetés.

- **Fichiers locaux :** Concernant l'importation des données préalablement collecter par l'utilisateur, le système prend les fichiers tel qu'ils sont et supprime tout ce qui n'est pas du texte comme les balises des fichiers xml ou encore les accolade et autre annotations propres au fichier JSon par exemple. Les fichiers seront par la suite devisés en plusieurs parties selon leurs tailles individuelles pour alléger le système et le reste des processus. Les formats pris en charge sont : .txt, .json et .xml. Après avoir tout importer le corpus sera stocké dans la base de données comme les autres avec un « id » qui le relie à l'utilisateur.

3.2.2- Enrichissement :

Etant donné que les besoins peuvent varier et changer avec le temps, une façon qui permet d'anticiper ce genre de situation est d'offrir la possibilité d'extension des données d'un corpus déjà généré ou importé. C'est-à-dire que les corpus sont dynamiques et leur taille peut changer pour contenir plus de domaines ou plus données si l'utilisateur le juge nécessaire à son travail. Cette fonction sera très utile si par exemple on voudra combiner des données locales avec d'autres depuis le web et travailler avec des corpus plus génériques, ou encore si dans la génération initiale certains mot clé non pas était inclus.

3.2.3- Structure des Corpus :

Les corpus générés depuis notre plateforme sont tous représentatif de leur contenue grâce à des attribut qui donnent une sorte de pièce d'identité pour chaque corpus généré ou importé. Comme ça lors de l'utilisation dans l'analyse des documents, les utilisateurs pourront choisir le corpus le plus approprié à la recherche. Ces attributs sont des données calculées après la collecte de donnée pour donner une vue d'ensemble de la collection finale et organisé dans un dictionnaire qui est un type de variable dans python représentant des couple « clé : valeur », les attributs rajoutés sont les suivants :

- **Nombre de Tokens**[28] : représente le nombre d'articles total dans le corpus après le web scrapping ou comme dans le cas de l'importation le nombre de morceaux après la segmentation.
- **Nombre de Mots** : représente le nombre total de mots retourné dans el corpus de façon général (sans élimination des mots vides ou doublant)
- **Nombre de Mots différent** : Les mots différent sont aussi calculés à partir de la liste initiale tout en enlevant les doublant et garder que les mots uniques dont le nombre d'apparition est égale à 1 dans la nouvelle liste.
- **Type** : Spécifique, Multi-Domaine ou importé par l'utilisateurs.
- **Articles** : Cet attribut est un ensemble de couple de clés valeurs lui aussi et est celui qui contient à l'intérieur les données textuelles retournés.
- **Nom** : Le nom du corpus est le seul attribut qui doit être donnée par l'utilisateur et sera attribué à son corpus pour mieux le distinguer après.
- **Id**[29] : chaque corpus se verra attribué également un id unique à lui seul calculé de façon aléatoire afin de pouvoir le distinguer des autres corpus dans la base de données.

3.3- Analyse textuelle :

L'analyse des donnée générés (ou importés) représente le deuxième pilier de notre application. Après avoir étudié les autres plateformes de gestion de corpus afin de pouvoir poser un standard comme point départ pour notre développement et ensuite l'enrichir, nous avons catégorisé les différents fonctions et besoins en termes d'analyse en 4 groupes majeurs :

- **Exploration** : Cette catégorie embarque tout ce qui peut servir pour comprendre le comportement du corpus et comment les mots peuvent être utilisés comme la liste des mots, le nombre d'apparition ou encore les concordances et autres.
- **Source des mots** : Une collection de fonction permettant de retourner les radicales des mots.
- **Espace Vectoriel** ; Dans ce groupe il y a tout ce qui est en relation avec les représentations vectorielles des mots comme les espaces sémantiques et le calcul des poids avec TF-IDF, Tf etc.
- **Calcul de similarité** : Outil du calcul de similarité entre deux texte cours avec plusieurs mesures disponibles.

Les différentes fonctions présentées dans la suite du rapport, sont applicable sur les corpus de la plateforme ou autre texte que l'utilisateurs aura choisi.

3.3.1- Exploration :

Comme décrit avant, cette catégorie permet à l'utilisateur de comprendre comment le corpus a été conçus et de le décortiquer, de façon général les outils proposés dans cette catégorie représentent la base de tout traitement textuelle globale. Les outils inclus dans cette collection sont les suivants :

- **Liste des mots** : Générer à partir d'une texte la liste de mots avec lesquels il a été rédigé. Réalisé à partir de la fonction de tokenization de nltk « word_tokenize() » ou aussi « string.spli() » de python dans le cas où la source est un corpus car la

[28] Unités
[29] Identificateur

fonction split sur des textes brut peut retourner des résultats mois agréable avec les caractère spéciaux.

- **Apparition des mots** : Cette fonction permet de calculer le nombre d'occurrences d'un mot dans un corpus ou texte, ou de calculer tous les mots avec leurs nombres d'apparitions, en commençant par calculer la liste des mots globales ensuite nous procédant au calcul des occurrences. Afin de limiter la répétition de recalculer les apparitions d'un mot (puis qu'il peut exister dans plusieurs positions dans la liste) nous sauvegardons les mots déjà traité dans une autre liste qui sera par la suite notre repère pour éviter ce genre d'erreurs.
- **Concordance** : Les concordances sont des passages de texte, permettant de voir les différentes utilisations d'un mot données en entrée dans des phrases. Nous cherchons dans la liste globale des mots le mot souhaité, ensuite le programme prend un segment de 5 mots des deux extrémités du mots d'entrée ce qui donne une chaine de caractère de 11 mots en générale (si le mot se trouve à la fin il se peut que la chaines soit d'une taille plus petite).
- **Mots vide (Stopwords) :** Supprimer tous les mots vides d'un corpus grâce à une liste prédéfinie ou un seuil d'apparition choisis par l'utilisateur et retourner les mots qui en été éliminé ainsi que le nouveau texte en procédant par un mécanisme de recherche et suppression si selon la condition il se trouve considérer comme étant un mot vide.
- **N-Gram :** Permet de générer la liste des n-gram de tous les mots ou d'un mot spécifique, encore une fois nous avons exploité la librairie nltk pour faire ce traitement. Si l'utilisateur cherche uniquement les suites dont un mot existe, le programme repassera en revue les suites et supprimera celles dont le mot ne fait pas partie.
- **Nettoyage :** Propose la possibilité de supprimer les caractères spéciaux ou les chiffres des textes d'entrée.

Afin de mieux comprendre cette catégorie nous avons schématisé les différents processus dans la figure suivante :

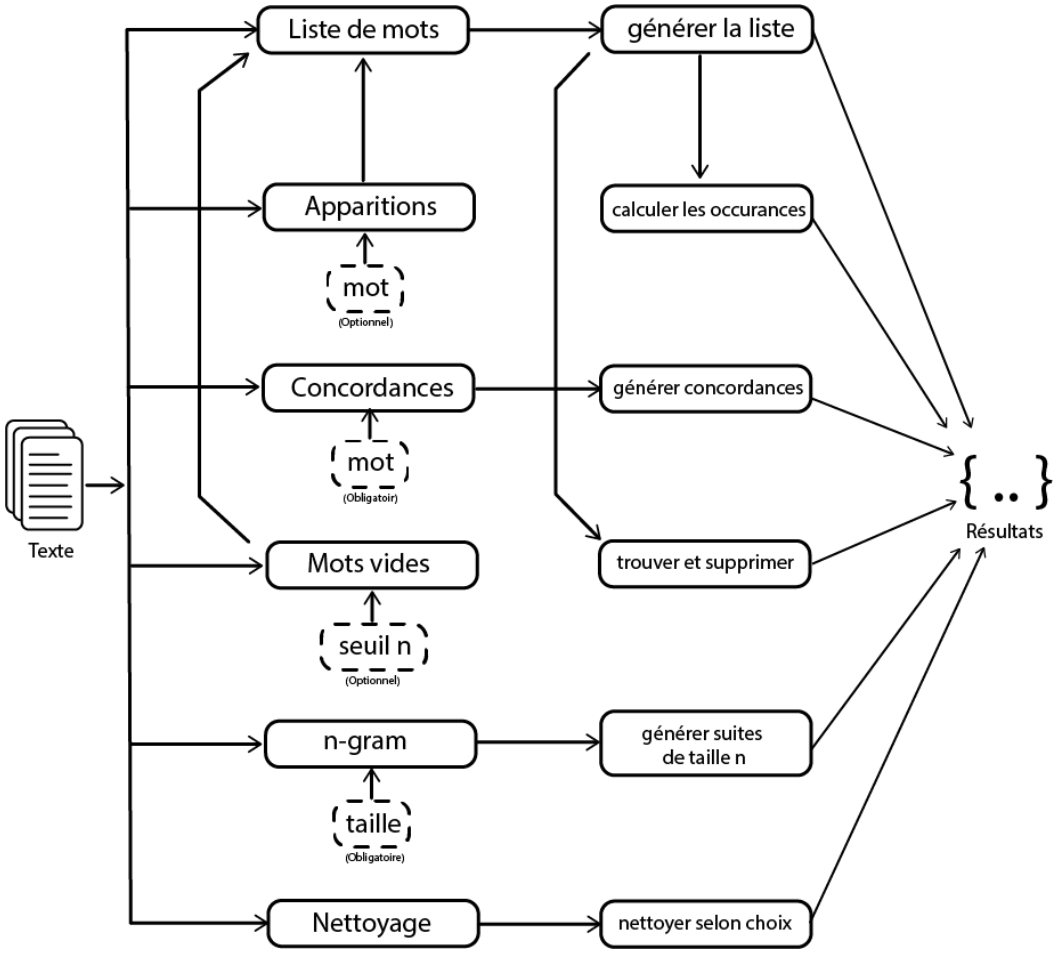

Figure12 : Schéma représentatif des fonctions d'exploration du texte.

3.3.2- Source des mots :

La plateforme propose aussi la possibilité de générer les stems des mots souhaités en spécifiant la méthode à utiliser selon le type de stem. Durant le développement nous avons implémenté trois méthodes de Stemming dont deux sont consacré au Stemming lourd. Les trois méthodes sont comme suit :

- **Tashaphyne: «** Tasha arabic light stemmer[30] » est une librairie python permettant de calculer le stem léger d'un mot donné en entré que nous avions utilité pour développer cet outil, il propose deux phases de calcul permettant de générer le stem léger du mot ou d'avoir son radical directement et c'est la fonction que nous avons implémentée pour développer les deux types de stemmers (léger et lourd) avec une seule librairie.
- **ISri[31] :** C'est une classe d'objet pour le calcul de stem pour la langue arabe incluse dans « nltk.stem » qui a été récemment mis à jour pour garantir un meilleur fonctionnement et des résultats plus cohérents notamment l'ajout de plus de stopwords ainsi que des flexions de mots.

[30] https://pypi.org/project/Tashaphyne/
[31] https://www.nltk.org/_modules/nltk/stem/isri.html

3.3.3- Espaces Vectoriels :

Afin d'enrichir la plateforme encore plus, nous avons aussi implémenté des fonctions relatives au représentations vectoriel des mots qui seront par la suite utilisé dans d'autre recherches de plusieurs domaines du TAL. Nous avons intégré les représentations des poids des mots comme : tf, tf-min_max, tf-idf, nous avons aussi inclus la génération des espaces sémantique présentée dans l'état de l'art, avec deux approches disponibles :

- **COALS :** Cette approche a été présenté dans l'état de l'art, et a été implémenté lors d'une autre recherche présentée par Atoub et Benayad[85]. La fonction prend en entré un texte (une seule unité) et calcul ainsi les poids des mots, et leurs corrélations ainsi que la liste des mots utilisés, en sortie le programme retourne deux fichiers textes. Le premier contient la liste des mots dont les vecteurs ont été calculés et le deuxième représente les vecteurs générés dont chaque ligne représente un vecteur d'un mot. Cette approche nous a aidée à intégrer cet outil mais nous avons quand même apporté des modifications. L'accès aux vecteurs nécessité un traitement préalable étant donné que les mots et leurs représentations vectoriels sont dans des fichiers différents. Notre programme prend donc la même méthode de calcul, mais au moment de la sauvegarde, nous utilisons des dictionnaires python avec des clés valeurs : « mot : vecteur », cette représentation nous permet de générer un seul fichier « .json » exportable, et garantir un accès direct et sur tout facile au mots souhaités sans devoir passer par un parcours de deux fichiers.
- **Word Embeddings :** L'approche consiste à exploiter les word-embeddings inclus dans un fichier traité par Zahran[86] qui contient plus de 6millions de représentations vectoriel de taille 300 milles de mots en langue arabe. Ce modèle déjà entrainé et disponible avec Skip-Gram ou CBOW mais nous avons utilisé uniquement le modèle Skip-Gram car c'est ce qui nous intéresse dans cette étude. La méthode consiste à prendre le fichier des WE et le parcourir afin de trouver la représentation d'un mot souhaité, et ce pour tout le texte étudié afin de générer les vecteurs sémantiques pour étudier les relations. Une étape de normalisation est nécessaire avant de pouvoir lancer la recherche des mots, car les mots inclus dans le fichier respectent quelque normes d'écriture comme la lettre « ۃ » qui s'écrit « ہ » et d'autres, du coup une autre fonction devra normaliser le texte en passant par les caractères de façon individuelle et voir le changement à apporter pour que les mots étudiés et ceux du fichier peuvent avoir une correspondance maximale et être sûr que le mot testé aura une représentation vectorielle. En revanche, Il se peut que le programme ne parvienne pas à trouver certains mots, dans ce cas les vecteurs seront des vecteurs de taille n avec une valeur de 0 pour toutes les colonnes. Encore une fois un dictionnaire sera retourné avec la même structure qu'avec l'approche COALS.

L'illustration suivante montre un schéma du fonctionnement des deux approches ainsi que les changements nécessaires à apporter avant de pouvoir exploiter le fichier des WE :

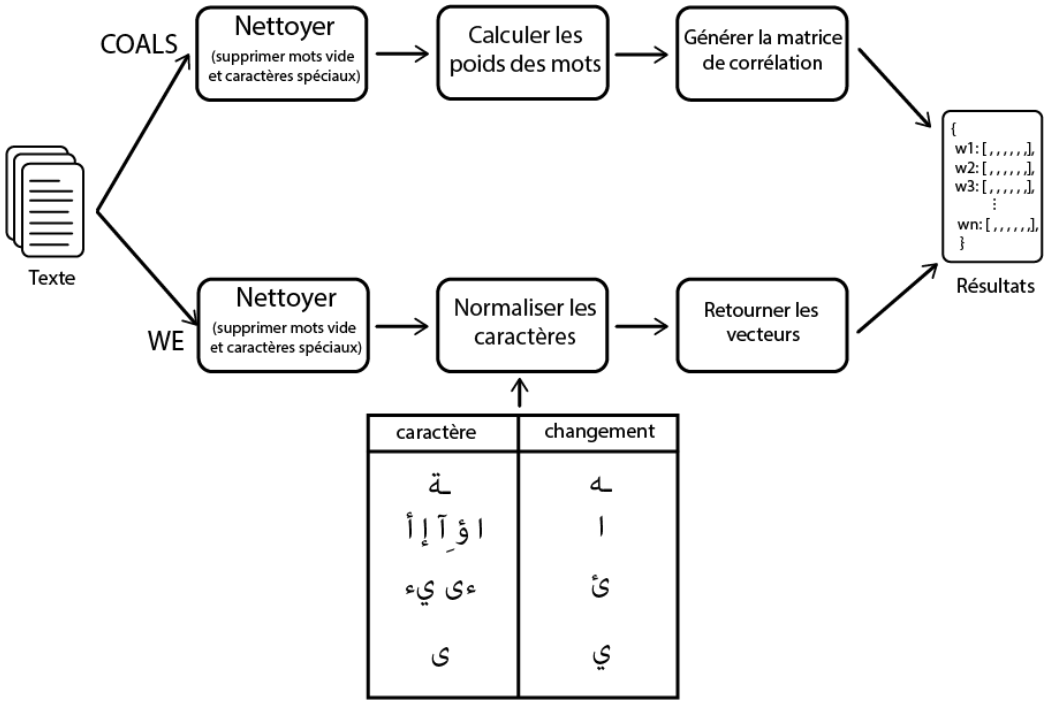

Figure13 : Schéma représentatif de COALS et WE.

3.3.4- Calcul de Similarité :

Concernant le calcul de similarité implémenté dans la plateforme, nous avons pris en considération les différents besoins d'analyse afin de proposer un outil plus avancé, et donc nous avons opté pour deux modes de calcul :

- **Texte vs Texte :** Ce mode propose le calcul de similarité entre deux texte cours avec le choix des mesures, une seule ou une combinaison de plusieurs. Le programme passera ensuite au teste des deux textes et retournera un dictionnaire contenant les différents scores pour chaque mesure ainsi que son type et son nom. Il y a aussi un score moyen retourné.
- **Dataset :** Il est aussi possible d'importer un dataset contenant des couple « texte1 » et « text2 » ensuite choisir les mesures à calculer afin de pouvoir générer un modèle d'entrainement pour l'apprentissage automatique. Le programme procède par couple de texte et calcule la similarité, les données de sortie seront représentées sous forme de Dataframe qui est un type d'objet de la librairie pandas[32] de python. L'avantage de ce type de donnée c'est qu'il pourra facilement être transformer en matrice, fichier « .csv » ou encore dictionnaire. La matrice pourra ensuite servir comme modèle pour d'autre recherches.

Il faut aussi mentionner quelque point important lors de l'utilisation de cet outil de la plateforme :

- Les textes devront être nettoyer en éliminant tout caractère spécial ou chiffres.
- Les mesures de calcul de similarité sémantique qui utilise le WordNet comme base de calcul ne seront pas inclus car l'Arabic WordNet n'est pas encore au point, nous cherchons à avoir des consistant.

[32] https://pandas.pydata.org/docs/

- Afin de pouvoir utiliser l'outil pour calculer la matrice d'entrainement à partir d'un Dataset, il est primordial d'importer des fichiers d'une forme assez spécifique. L'utilisateur devra importer deux fichiers texte dont les couples seront divisés avec chaque ligne représentant une partie du couple à étudier. Nous avons implémenté cette fonction de cette manière afin de garantir le bon fonctionnement de l'outil car les fichiers peuvent être assez imprévisible et leur structure varie entre chaque utilisateur.
- Les résultats seront toujours des dictionnaires. Le fichier « .json » contenant la matrice généré à partir d'un dataset auras en plus une commande python qui facilitera à l'utilisateur la conversion de son fichier en Dataframe ou csv.

Autre partie de cette section concerne les méthodes utilisées et incluses dans le calcul. En effet, les fonctions permettant de calculer le score syntaxiquement parlant exposé dans l'état de l'art de notre rapport ont toutes été intégré. En revanche pour enrichir encore plus la plateforme, ce qui représente l'un de nos objectifs, nous avons aussi fait appel à d'autres recherches dans le cadre de ce genre de traitement textuelle pour aboutir à un bon nombre de mesures. Les recherche qui nous aidé sont les suivant :

- **Mesures Syntaxique** : Le travail de Garoudja et Abdallah [87] nous a permis d'enrichir notre outil avec plusieurs autres méthodes du type syntaxique qui ont été élaboré lors de leur recherche concernant les systèmes d'évaluation automatique de réponses courtes.
- **Calcul Matriciel** : L'outil développé dans [85] concernant les espaces sémantiques a aussi permis de développer une approche basée sur les matrices de corrélation dérivé de COALS mais qui prend en compte l'ordre des mots, leurs poids et ainsi l'importance et bon placement des unités lexicaux. Le processus consiste à construire une matrice à partir des mots des deux termes, retournée les vecteurs sémantiques ensuite procédé à l'élimination des mots commun ainsi que leur indexe pour ensuite les comparer avec une formule.
- **Word Embeddings** : Pour utiliser les représentations WEs, nous avons aussi étudié le travail de HANNOUFI et HENNICHE [88]. Dans une autre recherche de calcul de similarité cette fois pour un système d'évaluation de réponses courtes, les deux étudiant ont développé une méthode qui consiste à faire la somme des vecteurs représentant les mots des deux textes pour ensuite avoir deux vecteurs sémantiques. Après avoir calculer les deux vecteurs nous procéderons à un test de similarité classique avec Cosinus ou autre afin de calculer la distance entre les deux représentations des phrases dans l'espace vectoriel. Comme durant la génération des espaces sémantique, les mots dont la représentation n'est pas disponible parmi les 6 millions mots, son vecteur sera représenté par une suite de « 0 » indiquant qu'il n'a aucune valeur indispensable à l'analyse.

Les deux dernières approches [87 & 88] ont été développer et conçus dont le but d'être utilisé avec la langue arabe afin de répondre eux limites concernant les recherches et les ressources traitant la langue. Des approches comme le WordNet ont montrés leurs efficacités avec l'anglais, mais la langue arabe ne bénéficie pas d'autant de concentration même si le Arabic WordNet est en cours de développement et montre de plus en plus d'efficacité mais comme dit précédemment ça reste quand même assez difficile à implémenter et n'est pas aussi fiable, mais des approches basées sur les corpus, et des Word Embedding déjà conçu pourrait être exploité dans le calcul de similarité car ils ne nécessitent pas de connaissances de la langue. Dans ce sens, notre plateforme permet de consolider plusieurs outils déjà développés dans le même endroit.

3.3.5- Gestions des Utilisateurs :

Afin de pouvoir accéder à la plateforme pour bénéficier de ses fonctionnalités présentées avant, il faut s'inscrire pour avoir un compte. Un utilisateur représente l'acteur principale de notre plateforme, il pourra s'inscrire de façon très simple qui ne demande pas trop d'informations mais qui peuvent être étendu par la suite en accédant au paramètre du profil. Le profil est caractérisé par plusieurs attributs qui sont :

- **Email** : Elément indispensable et unique pour chaque utilisateur. Qui servira comme identificateur.
- **Nom et prénom** : Obligatoire lors de l'inscription.
- **Mot de passe :** Le mots de passe est une suite de caractère au choix mais qui suit un certain pattern. Un Mot de passe doit contenir un ou plusieurs chiffres, des lettres en majuscules et minuscule (au moins une variante existe) et des symboles. Pour taille finale de 8 caractères.
- **Profession :** La profession de l'utilisateur nous permettra de connaitre les utilisateurs afin de déceler les différentes utilisations de notre plateforme dans le futur pour plus de mise à jour et amélioration selon les priorités.
- **Pays :** Le pays de l'utilisateurs habite nous servira aussi de voir le déploiement de notre plateforme au niveau des pays.
- **Les corpus :** Afin de pouvoir repérer les corpus des utilisateurs, cet attribut contient une liste de « id » reflétant les corpus qu'il a généré ou importé dans la plateforme.
- **Dernier utilisé :** Une trace du dernier corpus utilisé pour faciliter la reprise des traitements directement. Contenant un seul ID (mis à jour à chaque fois qu'un corpus différents est utilisé)

Les Utilisateurs de la plateforme restent tous sous la supervision d'un administrateur qui devra gérer la plateforme et pourra voir les différents statistiques relatifs à l'utilisation.

3.3.6- Stockage :

La base de données de notre plateforme est constituée de deux classes :

- **Les utilisateurs :** Un fichier « json » contenant une liste des utilisateurs, chaque élément de la liste contient les caractéristiques présentées avant.
- **Les corpus :** Dans un autre fichier, nous avons mis les corpus générés par les utilisateurs, également une liste dont le « id » est un unique et représente un seul corpus. Nous optons vers une solution à distance ou de cloud afin de minimiser l'espace mémoire, mais pour le moment les corpus sont stockés localement.

Les différentes fonctions d'ajout, création ou suppression ont été implémenté en python dans un fichier contenant les différentes fonctions relatives aux utilisateurs. Voici un schéma représentant le diagramme de classes de la plateforme :

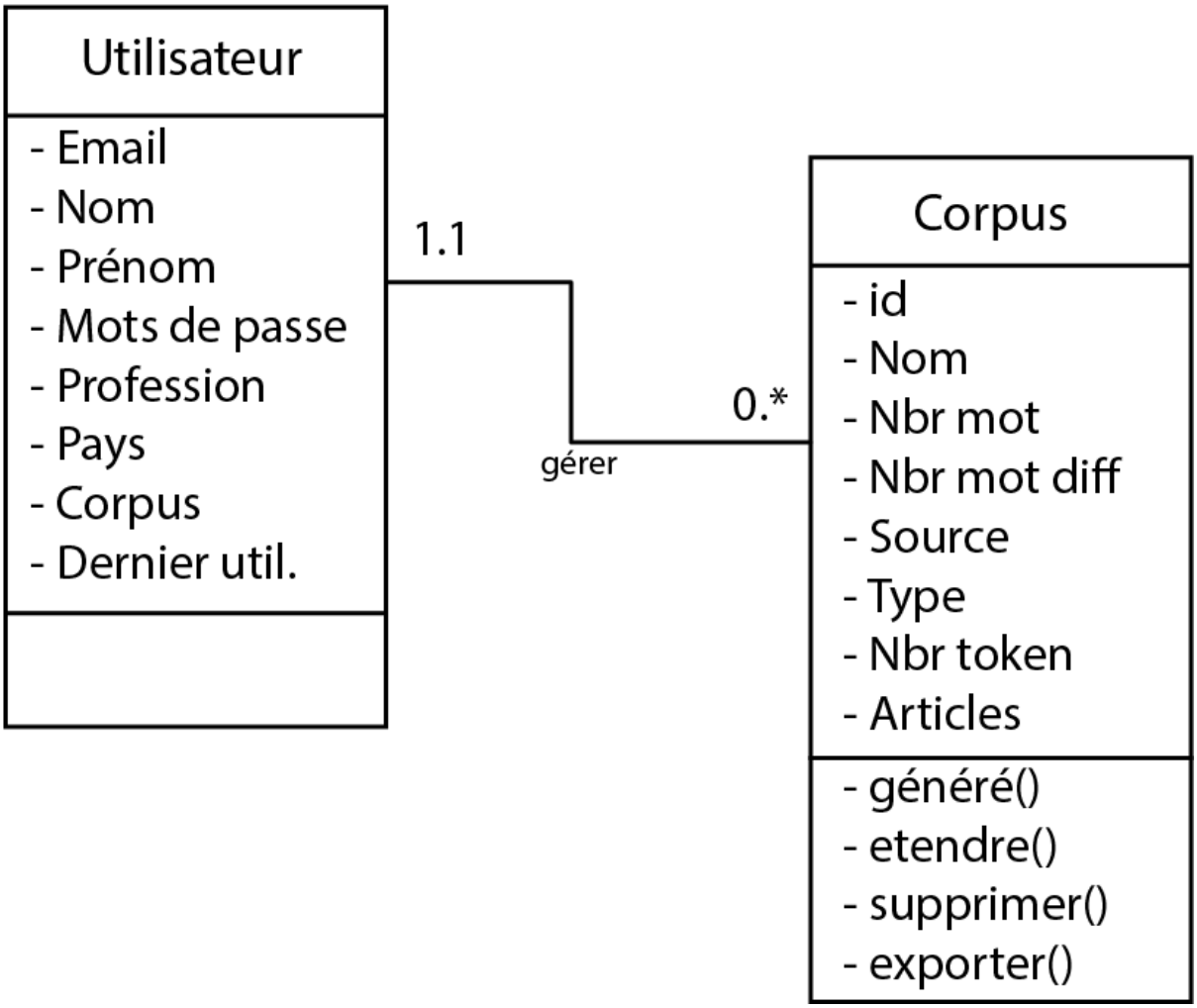

Figure14 : Diagrammes de classes.

Le reste des résultats des fonctions d'analyse ne seront pas sauvegardés, ils seront directement exportés vers l'utilisateur s'il le choisit.

3.4- Scénarios d'utilisation

Les scénarios d'utilisations sont décrits par un diagramme de séquence UML33, afin de bien comprendre comment la plateforme sera utilisé et définir les différents taches et navigations possible à inclure après l'étude du diagramme de cas d'utilisation dans « figure4 », voici le diagramme :

[33] Unifide Modleing Language

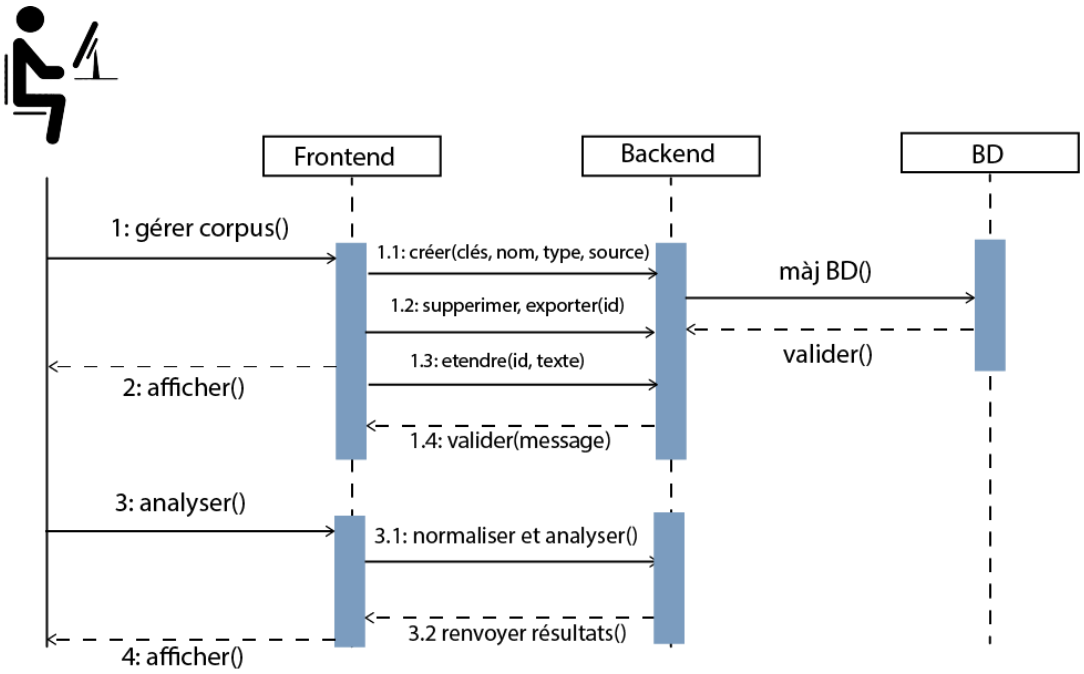

Figure15 : Diagramme de séquence.

4- Réalisation

4.1- Introduction

Maintenant que tout le contexte théorique lié à notre recherche a été introduit et la modélisation de la plateforme a été présentée dans le but de proposé un outil global pour personne qui s'intéresse au TAL. Ce chapitre présentera les différents outils utilisés ainsi que les besoins nécessaires en termes de matériel pour le bon fonctionnement de la plateforme.

4.2- Outils utilisés

- **Python**[34] **:** Python est une programmation interprétée, de haut niveau et est très polyvalent. Ce langage est dynamiquement typé et récupéré. Il prend en charge plusieurs paradigmes de programmation, y compris séquentielle, orientés objet et programmation fonctionnelle. Python est un langage qui présente une très grande bibliothèque, variée et robuste pour plusieurs types de développement. Pour le domaine du TAL, python représente un choix indiscutable avec tout ce qui offre comme librairies et facilité d'utilisation, la communauté lui accorde beaucoup de temps, c'est donc pour ça que nous avons choisis de l'utilisé comme outil principale du développement de notre backend. La version utilisée est python 3.8.3, mais nos codes ont été conçus de façon à fonctionner correctement avec les versions plus anciennes jusqu'au 2.7.
- **Django**[35] **:** Django est un Framework python open-source dédié au développement web. Il est orienté pour les développeurs ayant comme besoin de produire un projet solide rapidement et sans surprise. Django offre une liaison parfaire avec python. Il offre aussi des modèles déjà prêts à utiliser directement et/ou étendre pour satisfaire nos besoins.

[34] https://www.python.org/
[35] https://www.djangoproject.com/

- **JavaScript :** JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives mais aussi pour les serveurs avec l'utilisation de Node.js ou de Deno. Il s'exécute à l'aide d'un programme spécial appelé "Moteur Javascript". Nous l'avons utilisé pour proposer l'exportation des données d'analyse car il est plus simple de l'utiliser directement avec les données retournées par Django.
- **Visual Studio Code :** Pour écrire nos code python et page web, nous avons utilisé ce IDE[36] qui est très simple et léger qui ne consomme pas trop de mémoire. La communauté développe plusieurs outils comme « Kite[37] » qui est un système de prédiction intelligent pour les codes en python ce qui aide avec la productivité, un terminal puissant et une très bonne ergonomie.

4.3- Matériel du développement

Notre plateforme a été développer à l'aide de deux types de matériels. Nous avons utilisé notre machine de bureau personnelle physique ainsi que « Google colab38 » qui est une plateforme de google permettant d'accéder à plus de ressources de calcul, car certains testes de fonctionnalités exigent plus de mémoire vive par exemple.

4.3.1- PC personnel :

La configuration du PC personnel utilisée est la suivante :

- **CPU** : AMD Rayzen 3200g / 4 cœurs / 3.6ghz
- **RAM :** XPG d41 8gb 3000mhz
- **GPU :** AMD vega 8 graphics 4gb
- **OS :** Windows 10 Professionnel

4.3.2-Google colab :

Comme mentionné avant, la mémoire vive de notre pc n'est pas suffisante pour certaines taches de notre plateforme, comme Windows 10 a besoin de 4gb pour lui tout seul afin de fonctionner correctement, il nous restait uniquement la moitié de notre configuration disponible. C'est pour cela que nous avons exploité l'outil de google afin d'avoir plus de mémoire pour aboutir aux résultats. Avec les 24gb de mémoire, nous avons pu avancer mais ça reste quand même assez limité (notamment lors de la génération des espaces sémantique).

4.4- Présentation de la plateforme (Interfaces) :

Cette partie contient des captures d'écran des différentes pages et interfaces de notre plateforme.

4.4.1- Accueil, Inscription et authentification :

Dès qu'un utilisateur accède au lien de la plateforme, il sera directement face à une présentation du site qui par la suite l'invitera à se connecter ou créer un compte si ce n'est pas encore fait. Voici la page d'accueil :

[36] Integrated development environment
[37] https://www.kite.com/
[38] https://colab.sandbox.google.com/notebooks/welcome.ipynb?hl=fr

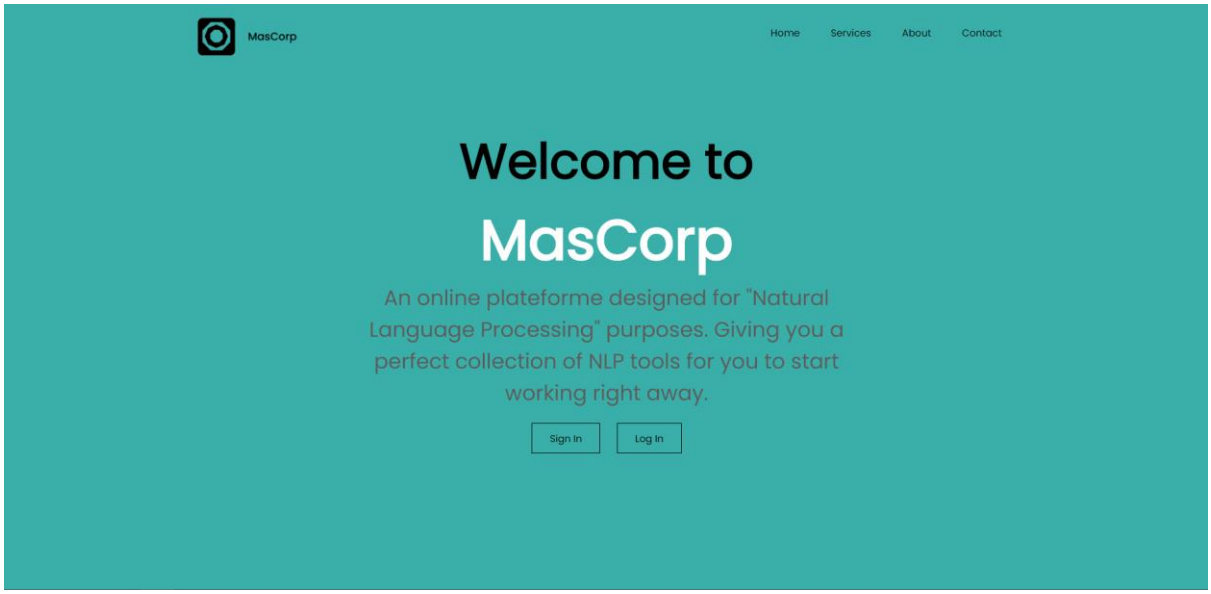

Figure16 : Page d'accueil de la plateforme.

Cette page offre également la possibilité de consulter les différents services proposés, de lire une présentation de la plateforme ou encore contacter directement l'administrateur.

Les pages de connexion et inscription sont tous les deux similaires et voici leur présentation en capture d'écran :

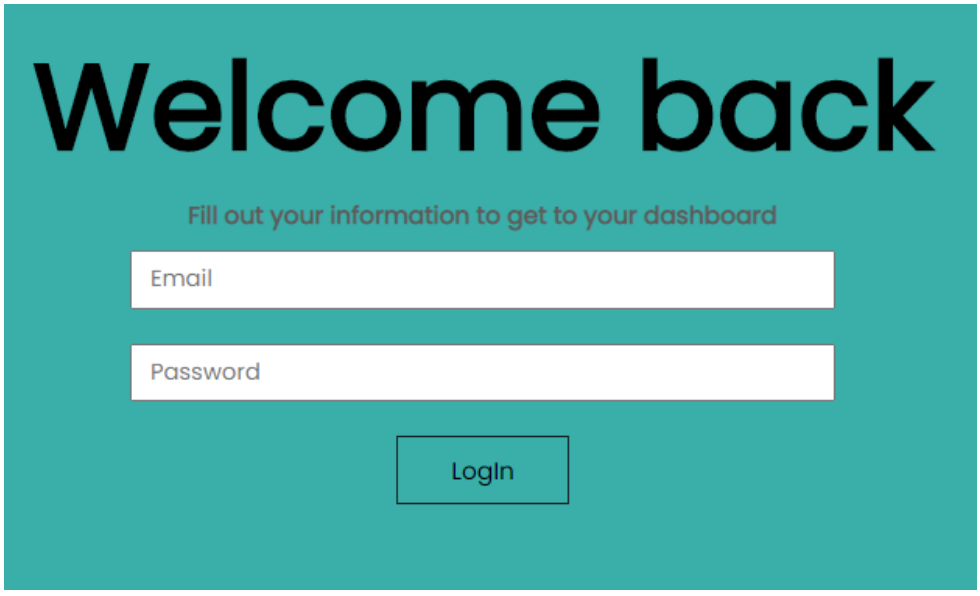

Figure17 : Page de connexion de la plateforme.

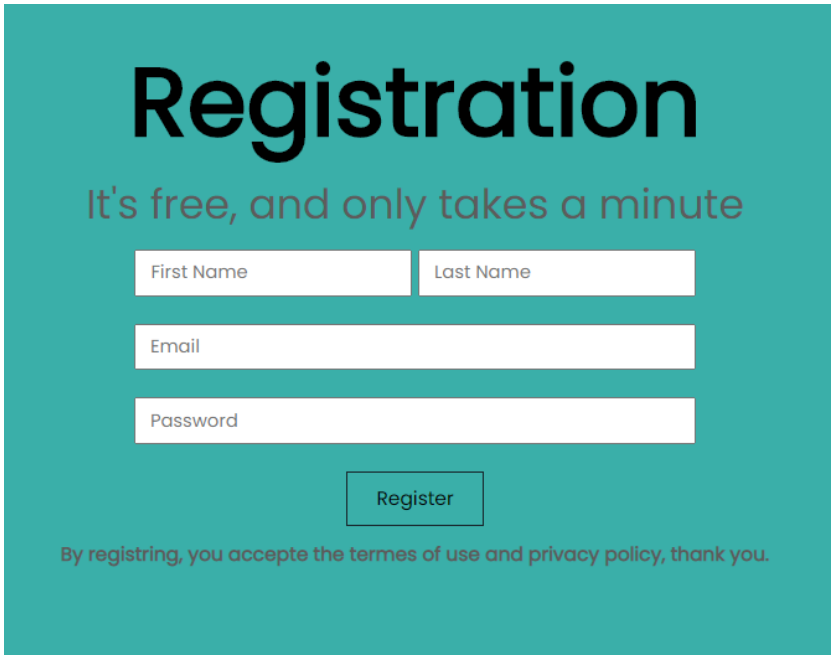

Figure18 : Page d'inscription de la plateforme.

4.4.2- Gestion des corpus :

Après la connexion, l'utilisateurs sera rediriger vers une page lui permettant de gérer ses corpus et en créer d'autres si besoins, ainsi que toutes les navigations possibles toujours présent, depuis cet écran il aura la liste de ses corpus, le dernier analysé ainsi que les entrées nécessaires pour un nouveau, la page « manage » lui offre la possibilité de supprimer, étendre ou exporter un corpus au choix :

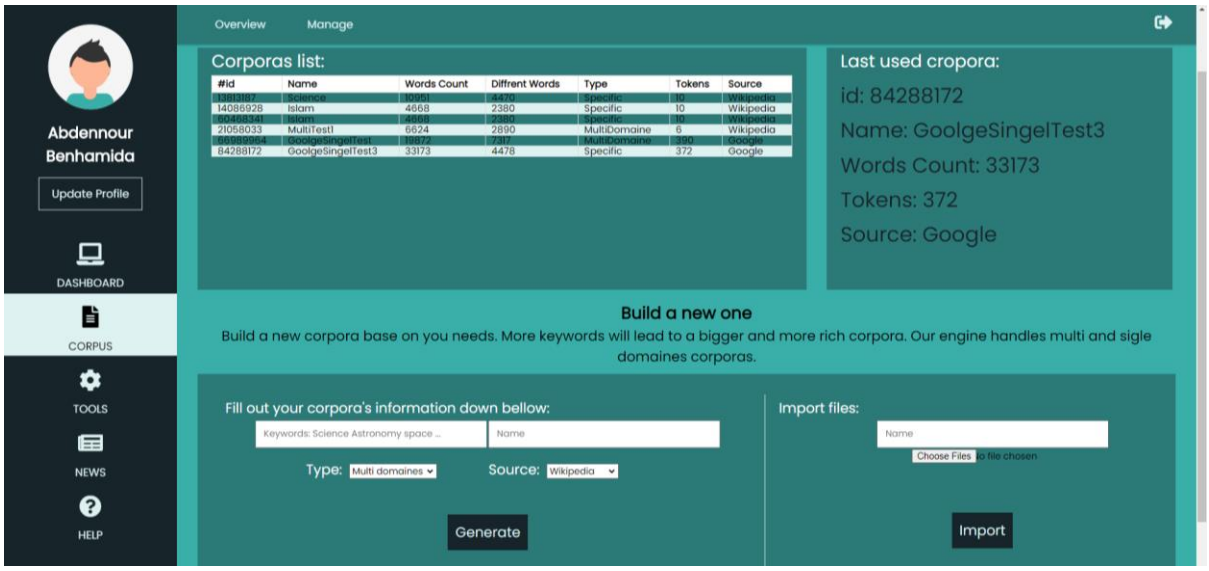

Figure19 : Vue d'ensemble sur les corpus et gestion.

4.4.3- Analyse textuelle :

Cet écran regroupe les différents outils d'analyse disponibles dans la plateforme, Chaque sous page contient plusieurs outils selon la catégorie, tous les outils qui seront présenter seront accessible depuis un corpus ou texte externe. Chaque page a une petite description des entrées et des sorties pour aider l'utilisateur à mieux exploiter l'outil, voici les captures d'écran pour illustrer les différents pages :

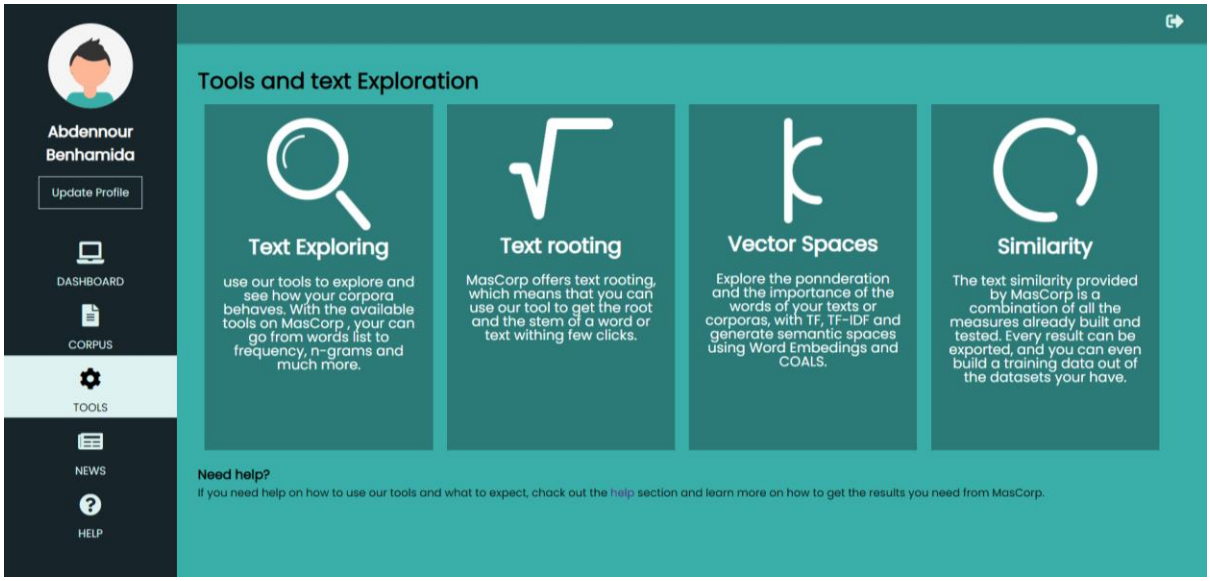

Figure20 : Page représentant les différentes catégories d'outils de la plateforme.

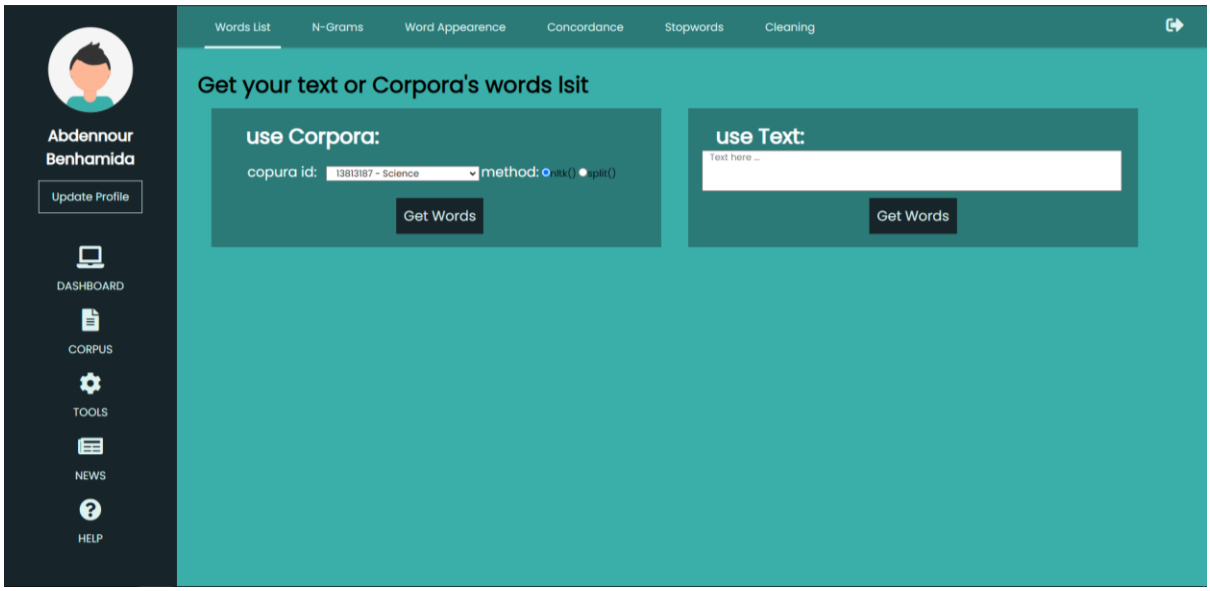

Figure21 : Analyse textuelle - Exploration.

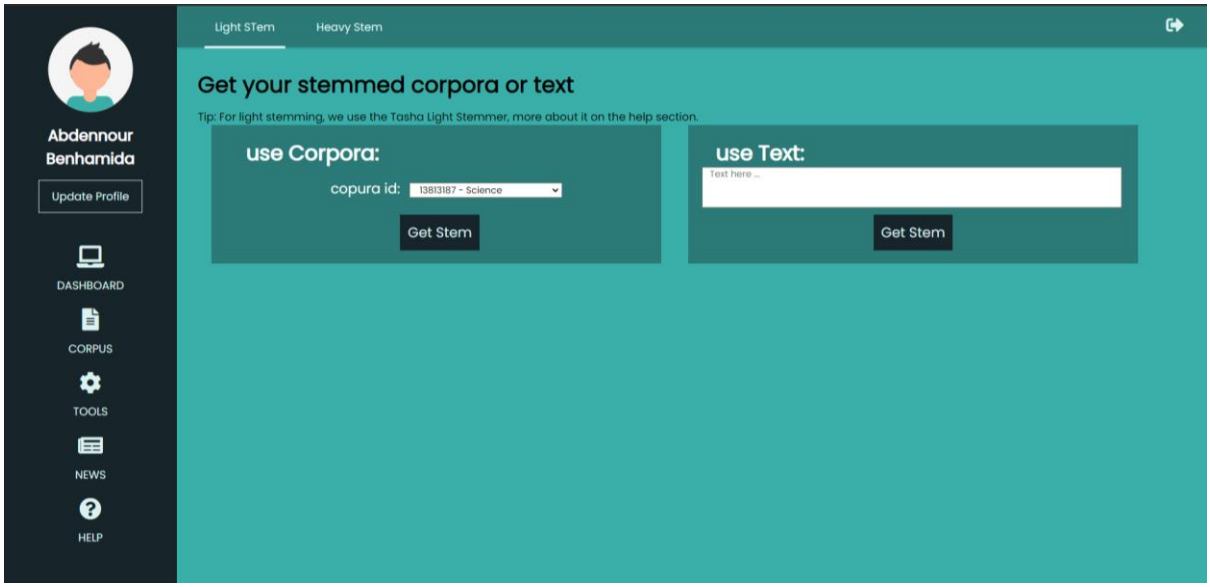

Figure22 : Analyse textuelle – Stemming

.

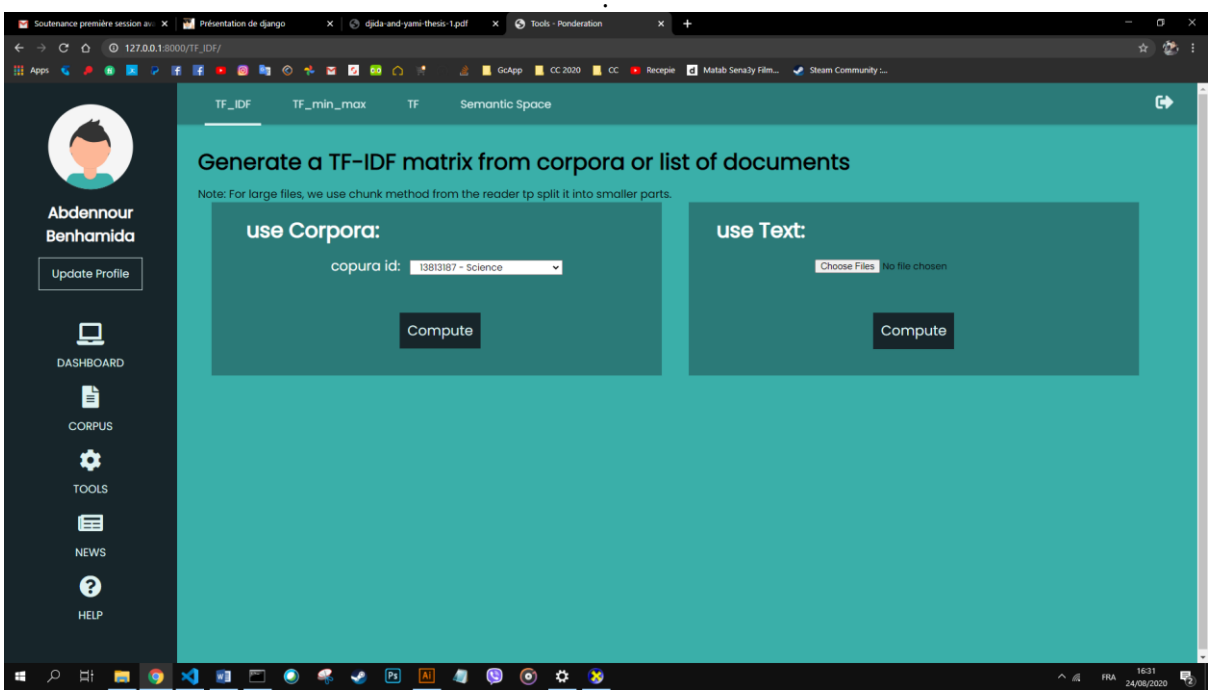

Figure23 : Analyse textuelle – Espaces vectoriels.

Figure24 : Analyse textuelle – Similarité.

Chaque page a sa propre signature, et tout est géré avec des formulaire Html basique et des views de Django très simplifié pour garantir une meilleure utilisation. Les données seront par la suite transférées vers python pour les traiter, et les résultats seront retournée sous forme de dictionnaire (seule forme de donnée disponible dans Django).

5- Difficultés rencontrées :

Durant notre travail, nous avons eus quelques obstacles dans le chemin de la réalisation, cette partie du rapport expose les différentes difficultés et problèmes rencontrés avant d'arriver à ce travail :

- Comme nos corpus sont générés à partir du web ou de Wikipédia, ils nécessitent une connexion internet fiable ce qui n'a pas toujours été le cas.
- Certains site internet limitent le nombre de fois qu'un utilisateur peut accéder à leurs contenus, par exemple si le site reçois trop de requêtes du même utilisateur il le bloque afin de ne pas tuer les serveurs, ce qui nous a conduit à rendre le processus de génération de corpus volontairement plus long.
- Au milieu de notre travail, Wikipédia avait abandonné le service qui nous permettait de retourner les articles depuis le dump, du coup il a fallu refaire le code et utilisé l'API Wikipédia et le tester à nouveau.
- La fonction qui permet de générer les Tags de Wikipédia retourne parfois des articles différents mais avec le même tag, ce qui mène le programme à la confusion comme il ne sait pas le quel choisir. Nous avons dû gérer cette exception en abandonnant toute sorte de confusion, même si dans la console il y avait des titres relatifs au domaine décrit par les mots clés.
- Certains sites retournés par des recherches googles n'ont aucun contenu textuel, comme des vidéos par exemple, du coup pour trouver plus de liens il faut des descriptions parfois plus approfondies depuis les mots clés.

6- Conclusion

Les fonctionnalités de la plateforme ont été implémentées comme décrit dans la conception, tout en essayant de garder les interfaces simples et représentatives le plus possible. La présentation des différentes pages montre aussi le coté visuel et devrait garantir une bonne interaction des utilisateurs avec les interfaces et les différents scénarios possibles.

La plateforme a été conçue de façon modulaire lui offrant beaucoup d'agilité pour des extensions futures. L'architecture utilisée durant la programmation rend les modules assez indépendants, et donc une mise à jour ne nécessite pas une modification globale.

Chapitre 3 : Performances & évaluation.

Ce chapitre représente les résultats obtenus grâce à notre plateforme en fonctions de ses outils, afin de pouvoir présenter les performances et procéder à une évaluation du travail :

- Introduction.
- Corpus générés.
- Résultats d’analyses.
- Comparaison avec les autres plateformes.
- Evaluation et Perspectives.

1- Introduction

Après avoir implémenté la plateforme dans tous ses aspects présentés dans la conception, il est nécessaire d'évaluer les performances des outils proposés. Le chapitre portera sur les corpus générés et que leurs caractéristiques ainsi que les résultats des fonctionnalités d'analyse textuelle.

2- Corpus générés

Pour évaluer les performances de notre générateur de corpus, nous l'avons utilisé pour voir les caractéristiques des corpus qu'on peut avoir, nous avons donc construit des corpus en utilisant plusieurs combinaisons de mots clés et aussi plusieurs sources afin de tester tous les aspects de l'outil, voici donc un tableau représentant les corpus générés :

Nom	Nbr Mots	Mots différents	Source	Tokens	Type	Temps
الاقتصاد	1,378,756	113,674	Wikipédia	615	Unique	1h10mi
كورونا	7,273	2,743	Google	302	Unique	7min
علوم	683,309	77,970	Wikipédia	479	Multi	52min
الجزائر	40,244	13,789	Wikipédia	384	Unique	37min
الجريمة الإلكترونية	1,121,548	127,364	Wikipédia	479	Unique	48min
الإعلام الآلي	6,583	3,134	Google	164	Multi	12min
الإعلام الآلي	61,968	7,913	Google	711	Multi	35min
الإعلام الآلي	2,331,856	152,489	Wikipédia	1279	Multi	1h40min
الإسلام	10,951	4,470	Liens	10	Unique	15min

Tableau12 : Exemple de Corpus générés par la plateforme.

Les corpus présentés dans le « tableau12 » ont tous été assemblés depuis notre plateforme en utilisant les méthodes exposées dans le chapitre traitant la conception et les algorithmes utilisés. Après la génération nous avons constaté les points suivant concernant nos corpus ainsi que l'outil :

- La taille des corpus représente un facteur majeur dans le domaine du TAL. Pour qu'un corpus soit utilisable il faut qu'il ait une certaine taille utilisable selon les besoins des utilisateurs, cet aspect de nos données de sortie varie énormément et n'est jamais stable car la taille est souvent liée à plusieurs autres éléments durant la description de nos corpus avec les mots clés ou encore le type et la source. Le tableau présente des corpus avec plus de 1 millions de mots et un autre avec plus de 2 millions, ces corpus ont été générés à partir de Wikipédia et leur description été assez détaillé et bien pensée, alors que le corpus utilisant les liens été plus petit comparé aux derniers car il n'y avait pas de dynamicité lors de la création (pas de génération de tag Wikipédia ou de ngram), les liens sont statiques et donc la taille est directement reliée au contenu et rien autre ne pouvais la changer, en revanche le corpus « كورونا » lui a été généré avec google et le contenu de ses tokens n'est très grand mais le sujet est d'actualité et fait partie des tendances du monde entier en ce moment, et donc plus de texte à retourner.
- Les corpus Wikipédia présentent beaucoup plus de données textuelles étant donné que le site est consacré à la rédaction et à la documentation, ce qui offre un avantage en termes de taille comparé aux autres sources.

- Nous avons aussi essayé de générer 3 corpus dont les attentes été les même mais avec plusieurs combinaisons. Le corpus « الإعلام الآلي » a été généré trois fois. Le premier été depuis google avec des mots clés basiques (4 mots au totale), en utilisant Google le corpus de sortie après 12min uniquement été déjà prêt et tous ses caractéristiques en été calculés. Le deuxième avait pris plus de temps, la même combinaison de mots a été utilisé mais cette fois en générant les tags Wikipédia du coup le système avait plus d'articles à décortiquer et donc la taille s'est multipliée par 10 mais le temps avait aussi augmenté. La troisième version été encore plus poussée, une combinaison encore plus large avec plus de mots clés traitant des domaines plus variés été notre corpus le plus large mais qui avait pris plus d'une heure pour se générer. Une autre fois, les corpus Wikipédia sont toujours plus grands, mais cette fois aussi il y'a le facteur de la description par mots clés qui intervient, une description plus détaillée mènera toujours à plus de données textuelles retournés.
- Les corpus Multi-Domaine sont aussi généralement plus grand que ceux qui ne traite qu'un seul domaine (encore une fois ça dépend de la description d'entrée).
- Le temps d'exécution est aussi concerné par la taille de la collection de données retournée, mais aussi à cause de notre traitement d'après qui calcule le reste des caractéristiques de l'objet notamment le calcul du nombre des mots et nombre des mots différents, ce processus pour 2 millions prendre plus de 20min.
- Nous tenons aussi à signaler qu'une connexion plus stable pourrai améliorer le temps d'exécution. Notre générateur permet de gérer les exceptions liées à la perte de connexion et les délais de connexions, se sont deux aspects dont la liaison internet est responsable de façon direct. Le programme prend du temps à essayer de garantir le lien s'il est perdu et attend 5 secondes à chaque fois que la connexion est perdue, ce qui lui rend plus lent dans certains cas, durant la génération du corpus « الجزائر » le système a perdu environ deux minutes au totale, comparé aux autres corpus plus larges cela pourra prendre encore plus de temps. Il y a aussi la gestion des ambigüités au niveau des tags Wikipédia (un tag peut avoir plusieurs articles dans plusieurs domaines) comme le système ne peut pas déterminer leur pertinence automatiquement, ça nous fait perdre en moyenne 8% des tags de départs même si certains peuvent avoir une certaine correspondance avec la description.
- Le corpus « الجريمة الإلكترونية » a été utilisé par Hadjersi et Benguergoura durant leur travail [89], notamment en générant un espace sémantique, ils l'ont comparé avec deux autres corpus du même domaine et les résultats sont assez comparables, ce qui nous donne un corpus qui est assez représentatif du sujet qu'il traite, l'un des caractéristiques principales d'un corpus (évoqué dans l'état de l'art).

Le « tableau11 » ne présente aucun corpus importé depuis des fichiers locaux car nous le considérons comme un autre type de données dont notre générateur n'intervient pas à la collection. Nous avons quand même voulu tester la possibilité de perte de texte ou de modification, 3 corpus de différentes tailles ont donc été exporté depuis la plateforme ensuite importer. Les trois corpus comparés aux originaux été 100% identique en termes de contenu textuelle, mais le nombre de Tokens a changé dans les 3 corpus. En effet, les corpus sont exportés dans un fichier json contenant tous les token dans l'attribut « Articles », mais quand le fichier est importé, le système le décompose en fonction de sa taille, s'il le juge suffisamment grand pour ralentir le système, il le décompose en plusieurs morceaux plus petits (2.5mb chacun).

Cette approche aide à faciliter la tâche et alléger le traitement, mais pourrai perturber les analyses dont le nombre de documents fait partie d'une équation donnée, comme le TF-IDF qui calcule le nombre de document dont un mot est utilisé. Ce qui nous mène donc à obliger

l'utilisateur à importer ses fichiers en s'assurant qu'ils sont correctement décomposés, si non le nombre de token pourrai changer selon les corpus.

3- Résultats d'analyses

Pour voir les résultats d'analyses textuelle de notre outil nous avons utilisé plusieurs types de donnée d'entrée représenté par des corpus et des textes bruts copié et ensuite collé sur l'entrée de l'interface, voici ce que nous avons pu constater :

- Les résultats d'analyse retournée par la plateforme donnent de bons résultats dans tous les aspects sauf au niveau du changement du nombre de token dans le cas des corpus importés, qui affecte les résultats du calcul du TF-IDF par exemple. Les trois couples de corpus ont donnée des résultats comparables mais pas à 100% dans toutes les caractéristiques. Par exemple, un corpus contient 3 documents, et après son importation le système l'a découpé en seulement deux. Prenons un mot qui existe dans deux documents du premier, mais dans le deuxième il existe uniquement un seul document contenant le mot (le système à fusionner les deux premiers documents dans ce cas), le IDF passe de « log (3 /2) ≈ 0.18 » à « log (2/1) ≈ 0.30 » ce qui est clairement différent. Dans le cas où l'utilisateurs a déjà son TF-IDF en cherchant juste à comparer ou vérifier, cela peut porter à confusion. Mais s'il le fait pour la première fois, le résultat aura un poids représentatif et ne posera pas de problème pour l'utiliser ailleurs.
- Concernant la génération des espaces sémantique, même avec les 24gb de « Google colab » la taille de la matrice générée par l'algorithme COALS ne dépassent pas les 30,000 mots car la mémoire été déjà saturée, mais même avec cette taille les résultats restent assez représentatifs et l'espace peut être exploiter dans les domaines concernés, mais ça nous donne une estimation de la taille de RAM minimal pour que le système puis fonctionner correctement. Ceci confirme le résultat [90] le fait que pour une taille de la matrice de l'espace sémantique comprise en 14000 et 100000, les résultats retournés restent comparables.
- Pour tester l'outil de calcul de similarités il nous fallait un dataset mais nous nous somme fiés aux résultats obtenus auparavant par les chercheurs qui ont traités le sujet plus en détail étant donné que ce n'est pas le but principal de notre recherche et que toutes les mesures de calcul incluses dans notre plateforme ont déjà été testés et évalués avec des dataset dans les normes.
- Au niveau des autres fonctionnalités d'analyses les résultats sont globalement consistant et donne des résultats réels.

4- Comparaison avec les autres plateformes

Après avoir développé notre plateforme et tester ses fonctionnalités, nous nous intéressons aussi à voir ce qu'elle vaut comparée aux autres plateformes car ça fait partie de nos objectifs, concevoir une application permettant de consolider le plus d'outil possible out en restant générique et modulaire, le tableau suivant présente donc notre plateforme face aux autres étudiés durant nos recherches :

- **Corpus :**

Gest./Fnct	Création	Enrichissement	Téléchargement	Prêt	Importation
SketchEngine	Oui (web)	Non	Oui	Oui	Oui
TXM	Oui(web)	Non	Non	Non	Oui
CQPweb	Non	Non	Non	Oui	Oui
Développée	Oui(web)	Oui	Oui	Oui	Oui

Tableau13 : Comparaison entre les corpus des plateformes étudiés et la plateforme développée.

Comme indiqué dans le tableau, nous avant pus atteindre nos objectifs en termes de gestion de corpus face aux autres gestionnaires, nous avons la possibilité d'exploiter tous les aspects possibles étudié avant. Tous les corpus générés par notre plateforme avant seront disponible pour chaque tous les utilisateurs afin e travailler directement si le besoin n'est pas spécifié.

- **Fonctionnalités :**

Fonctions	**SketchEngine**	**TXM**	**CQPweb**	**Développée**
Tokenization	Oui	Oui	Oui	Oui
Nettoyage	Non	Non	Non	Oui
Stemming	Oui	Oui	Oui	Oui
N-Gram	Oui	Oui	Non	Oui
Synonyme	Oui	Non	Non	Oui
Similarité	Non	Non	Non	Oui
Espaces Sémantiques	Non	Non	Non	Oui
Concordance	Oui	Oui	Oui	Oui
Nombre d'apparition	Oui (non direct)	Oui	Non	Oui(direct)
Traduction	Oui (quelque langues)	Non	Non	Non
Prise en charge de l'arabe	Oui	Oui	Oui	Oui
Plateforme	Web	Desktop	Web	Web
Moyen d'accès	Payant (après 30j)	Gratuit	Payant	Gratuit

Tableau14 : Comparaison des fonctionnalités des plateformes étudiés et la plateforme développée.

Le « tableau13 » présente donc les différents aspects étudiés dans le cadre référentiel posé au début de notre recherche. Notre plateforme présente une collection d'outil assez varié et très proche de SketchEngine qui le gestionnaire le plus complet des trois étudiés. Nous avons aussi pus enrichir la nôtre avec 3 autres fonctionnalités : Nettoyage des texte externes, similarité textuelle et espaces sémantiques. Ces trois outils d'analyse ne sont pas disponibles sur SketchEngine, d'ailleurs les espaces vectoriels ne sont pas du tout proposés, donc nous avons aussi un autre avantage de ce côté-là. Les fonctions ne sont pas présentées de la même façon dans 100% des cas, mais les résultats sont assez proches, SketchEngine est le gestionnaire qui propose le plus de personnalisation possible.

5- Evaluation et perspectives

Les résultats présentés sont des résultats dont la langue arabe est la langue la plus utilisés. En effet nous n'avons pas présenté de corpus d'autres langue même si ça reste possible, mais nous nous sommes beaucoup plus intéressés à l'Arabe étant donné que c'était notre première motivation en travaillant sur un gestionnaire de corpus, enrichir les outils de la langue. Nous pensons qu'une meilleure intégration des autres langues sera nécessaire pour que toutes les fonctionnalités soient disponibles.

En termes de génération de corpus l'utilisateurs n'aura pas à explicitement indiquer la langue comme nous avons implémenté un détecteur de langue, le système s'occupe de cette tâche, en revanche, la seule langue capable d'exploiter la génération des espaces sémantiques avec les WordEmbeddings est la langue arabe, COALS reste disponible pour toutes les langues.

Conclusion générale.

Nous avons développé une application purement destinée au domaine du TAL, qui offre aux utilisateurs une collection d'outils et de fonctionnalités très variée et assez polyvalente. La plateforme présente une contribution dans le domaine du TAL pour tous les chercheurs :

- Facilite l'accès et l'utilisation grâce à sa simplicité.
- Offre un outil global consistant mais surtout gratuit.
- Répond aux besoins des utilisateurs ainsi qu'aux limitations en termes de nombre de fonctionnalités incluses.
- Propose un outil d'origine arabe pour la langue arabe en premier et les autres globalement.
- Contribue à l'amélioration des ressources de la langue, notamment avec la création des corpus et l'analyse dans le même lieu.

Afin de mieux présenter la plateforme pour une bien meilleure contribution et valeur au futur, nous pensons aussi que ces améliorations pourront aider :

- Améliorer l'interface notamment au niveau de la langue utilisé, nous utilisons l'anglais parce que c'est la plus commune mais offrir le choix aux utilisateurs est aussi intéressant pour cibler plus d'utilisateurs.
- Améliorer le processus de génération pour s'exécuter en arrière-plan et notifier l'utilisateur quand il sera prêt afin de ne pas le faire trop attendre.
- Donner la possibilité de rendre des corpus publics après la génération, comme ça tout le monde pourra les utiliser si besoin.
- Intégrer la publication de document scientifique et outil du TAL pour contribuer à la plateforme.
- Améliorer les interactions des utilisateurs avec l'interface en fonction des avis et suggestion des utilisateurs.

  Les objectifs établis au début de notre travail ont tous été atteint sauf pour la combinaison de Machine Learning pour la similarité textuelle, mais comme ça représente une partie de notre travail nous nous sommes intéressés à l'outil lui-même et non pas ses améliorations.

Comparé à nos motivations concernant l'amélioration des ressources et outils arabe, la plateforme propose une bonne collection d'outils pour la langue arabe et les autres également. Nous pensons également que l'outil pourra contribuer au projet de « الذخيرة العربية » étant donné qu'il traite le même sujet et qu'il est orienté vers la langue arabe comme indiqué dans la problématique de notre thème.

Les corpus générés et les codes sont disponibles au téléchargement depuis un lien drive représentant un dossier contenant tous les outils nécessaires et les corpus utilisés.

Références

[1]- Debili, Fathi & Hadhémi, Achour & Souissi, Emna. (2002). La langue arabe et l'ordinateur de l'étiquetage grammatical à la voyellation automatique. Correspondances: bulletin de l'IRMC, ISSN 0330-7417, Nº 71, 2002, p. 10-26.

[2]- El Ouardi Ali Abdelwahab. University of Baghdad College of Education for Human Science / Ibn Rushd. Nominal state in (Anomalous Readings) book by Ibn Khalawayh (370 a. h) : semantic study. Vol. 1, Issue 220 (31 Mar. 2017), pp.165-182, 18 p.

[3]- Xiao, R., McEnery, A., Baker, P., and Hardie, A. (2004) Developing Asian language corpora: Standards and practice. In Proceedings of the Fourth Workshop nous Asian Language Resources, Sanya, Hainan. Island, pp. 1–8, March 25, 2004.

[4]- Hundt, M., Sand, A., and Skandera, P. (1999) Manual of Information to Accompany the Freiburg- Brown Corpus of American English ('Frown'). URL: http://khnt.hit.uib.no/icame/manuals/frown/ INDEX.HTM.

[5]- Spoor, J. (1996) The copyright approach to copying nous the Internet: (Over)stretching the reproduction right? In H. Hugenholtz (ed.), The Future of Copyright in a Digital Environment, pp. 67–80. Dordrecht, the Netherlands: Kluwer Law International.

[6]- Biber, D. (1993) Representativeness in corpus design. Literary and Linguistic Computing 8(4): 243–257.

[7]- Hunston, S. (2002) Corpora in Applied Linguistics. Cambridge, U.K.: Cambridge University Press. Ide, N. (1998) Corpus encoding standard: SGML guidelines for encoding linguistic corpora. In LREC-1998 . Proceedings, Granada, Spain, pp. 463–470.

[8]- Resnik P. (1998). Parallel strands: A preliminary investigation into mining the web for bilingual text. In conference of the association for machine translation in the Americas, 1998.

[9]- Ghani R. et Jones D. (2001). Mladenic, mining the web to create minority language corpora. CIKM 2001, 279-286.

[10]- Dhaou Ghoul (2014). « Construction d'un corpus arabe à partir du Web dans le but d'identifier les mots-outils ou tokens ».

[11]- https://www.mediawiki.org/wiki/API:Search

[12]- Wadjdi Zaghwani. Critical Survey of the Freely Available Arabic Corpora.(https://www.researchgate.net/publication/314092010_Critical_Survey_of_the_Freely_Available_Arabic_Corpora)

[13]- http://aracorpus.e3rab.com/argistestsrv.nmsu.edu/AraCorpus/

[14]- Alrabiah, M., Al-Salman, A. and Atwell, E. (2013). The design and construction of the 50 million words KSUCCA King Saud University Corpus of Classical Arabic. In Proceedings

of the Second Workshop on Arabic Corpus Linguistics (WACL-2), 22 Jul. 2013, Lancaster University, UK. (http://ksucorpus.ksu.edu.sa/?p=43)

[15]- https://sites.google.com/site/motazsite/arabic/osac

[16]- http://sourceforge.net/projects/arabiccorpus/

[17]- http://sourceforge.net/projects/arabiccorpus/

[18]- http://sourceforge.net/projects/tashkeela/

[19]- http://sourceforge.net/projects/arabiccorpus/

[20]- Al-Thubaity, A., Khan, M., Al-Mazrua, M., Al-Mousa, M. (2013). New Language Resources for Arabic: Corpus Containing More Than Two Million Words and a Corpus Processing Tool. In Proceedings of the Asian Language Processing (IALP) Conference, pp.67,70. (http://sourceforge.net/projects/kacst-acptool/files/?source=navbar)

[21]- http://sourceforge.net/projects/arabicwordcorpu/files/

[22]- http://www.comp.leeds.ac.uk/eric/latifa/research.htm

[23]- http://cri.kacst.edu.sa/Resources/TRN_DB.rar

[24]- http://l2arabiccorpus.cercll.arizona.edu/?q=allFiles

[25]- Rafalovitch, A. and Dale R. (2009). United Nations General Assembly Resolutions: A Six-Language Parallel Corpus. In Proceedings of the MT Summit XII, pages 292-299, Ottawa, Canada. (http://www.uncorpora.org/)

[26]- http://www.kunuz/

[27]- https://github.com/anastaw/Meedan-Memory

[28]- http://old-site.clsp.jhu.edu/ws99/projects/mt/toolkit/

[29]- Almeman K., M. Lee and A. Almiman. (2013). Multi Dialect Arabic Speech Parallel Corpora, In Proceedings of the First International Conference on Communications, Signal Processing, and their Applications (ICCSPA'13), Sharjah, UAE, 12-14 Feb. 2013.(http://www.cs.bham.ac.uk/~kaa846/arabic-multi-dialect-text-corpora.html)

[30]- Graja, M., Jaoua, M. and Hadrich Belguith, L. (2010) Lexical Study of A Spoken Dialogue Corpus in Tunisian Dialect. In Proceedings of the ACIT 2010: the International Arab

Conference on Information Technology, Benghazi, Libya, 14–16 December 2010.(https://sites.google.com/site/anlprg/outils-et-corpus-realises/TuDiCoIV1.xml?attredirects=0)

[31]- http://www.kacstac.org.sa/

[32]- http://smlc09.leeds.ac.uk/query-ar.html

[33]- Alansary, S., Nagi, M. and Adly, N. (2007). Building an International Corpus of Arabic (ICA): progress of compilation stage. In Proceedings of the 7th International Conference on Language Engineering, Cairo, Egypt, 5–6 December 2007. (http://www.bibalex.org/ica/en/About.aspx)

[34]- http://arabicorpus.byu.edu/

[35]- http://quranytopics.appspot.com/

[36]- http://textminingthequran.com/wiki/Main_Page

[37]- Steinberger, R., Pouliquen, B., Kabadjov, M., Belyaeva, J. and Van der Goot, E. (2011). JRC-Names: A freely available, highly multilingual named entity resource. In Proceedings of the 8th International Conference Recent Advances in Natural Language Processing (RANLP). Hissar, Bulgaria, 12-14 September 2011 (http://ipsc.jrc.ec.europa.eu/index.php?id=42#c2696)

[38]- http://www1.ccls.columbia.edu/~ybenajiba/downloads.html

[39]- Mohit, B., Schneider, N., Bhowmick, R., Oflazer, K and Smith, N.-A. (2012). Recall-Oriented Learning of Named Entities in Arabic Wikipedia. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pp. 162-173(http://www.ark.cs.cmu.edu/ArabicNER/)

[40]- Azab M. , Bouamor, H., Mohit, B. and Oflazer, K. (2013). Dudley North visits North London: Learning When to Transliterate to Arabic. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL/HLT 2013), Atlanta, USA, June 2013.

[41]- Attia, M., Toral, A., Tounsi, L., Monachini, M. and Van Genabith, J. (2010). An automatically built Named Entity lexicon for Arabic. In Proceedings of the Language Resources and Evaluation Conference (LREC) 2010. Valletta, Malta(https://sourceforge.net/projects/arabicnes/).

[42]- http://www1.ccls.columbia.edu/~ybenajiba/downloads.html

[43]- http://nlp.qatar.cmu.edu/qalb/

[44]- http://www.comp.leeds.ac.uk/scayga/alc/corpus%20files.html

[45]- Alkanhal, M.I., Al-Badrashiny, M.A., Alghamdi, M.M., Al-Qabbany, A.O. (2012). Automatic Stochastic Arabic Spelling Correction With Emphasis on Space Insertions and Deletions. In IEEE Transactions on Audio, Speech, and Language Processing. 20(7): 2111-2122, Sept. 2012(http://cri.kacst.edu.sa/Resources/TST_DB.rar).

[46]- Attia, M. Pecina, P., Tounsi, L., Toral, A., Van Genabith, J. (2011). A Lexical Database for Modern Standard Arabic Interoperable with a Finite State Morphological Transducer. In Mahlow, Cerstin; Piotrowski, Michael (Eds.) Systems and Frameworks for Computational Morphology. Second International Workshop, SFCM 2011, Zurich, Switzerland, August 26, 2011 (http://arabicwordcount.sourceforge.net/).

[47]- Attia, M., Pecina, P., Samih, Y., Shaalan, K., Van Genabith, J. (2012). Improved Spelling Error Detection and Correction for Arabic. In Proceedings of the COLING 2012, Bumbai, India (http://sourceforge.net/projects/arabic-wordlist/).

[48]- Attia, M., Toral, A., Tounsi, L., Monachini, M. and Van Genabith, J. (2010). An automatically built Named Entity lexicon for Arabic. In Proceedings of the Language Resources and Evaluation Conference (LREC) 2010. Valletta, Malta (https://sourceforge.net/projects/arabicmwes/).

[49]- Attia, M., Samih, Y., Shaalan, K., Van Genabith, J. (2012b). The Floating Arabic Dictionary: An Automatic Method for Updating a Lexical Database. In Proceedings of the COLING 2012, Bumbai, India (http://arabic-unknowns.sourceforge.net/).

[50]- http://sourceforge.net/projects/arabicstopwords/

[51]- Attia, M., Pecina, P., Tounsi, L., Toral, A. and Van Genabith, J. (2011c). Lexical Profiling for Arabic. 6 Electronic Lexicography in the 21st Century. Bled, Slovenia (http://obsoletearabic.sourceforge.net/).

[52]- Al-Maadeed, S., D. Elliman, C.A. Higgins. (2002). A database for Arabic handwritten text recognition research. In Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition. Niagara-on-the-Lake, Canada, 2002 (http://handwriting.qu.edu.qa/dataset/).

[53]- Hassaïne, A., Al-Maadeed, S., Alja'am, J.M., Jaoua, A., Bouridane, A. (2011). The ICDAR2011 Arabic Writer Identification Contest. In Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), pp.1470-1474, 18-21 Sept. 2011 (http://handwriting.qu.edu.qa/dataset/).

[54]- Al-Maadeed, S., W. Ayouby, A. Hassaïne, J. Alja'am. (2012). QUWI: An Arabic and English Handwriting Dataset for Offline Writer Identification. In Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition, Bari, Italy, pp. 742-747(http://handwriting.qu.edu.qa/dataset/).

[55]- Al-Maadeed, S., W. Ayouby, A. Hassaïne, J. Alja'am. (2012). QUWI: An Arabic and English Handwriting Dataset for Offline Writer Identification. In Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition, Bari, Italy, pp. 742-747 (http://handwriting.qu.edu.qa/dataset/).

[56]- Bensalem, I., Rosso, P., Chikhi, S. (2013). A New Corpus for the Evaluation of Arabic Intrinsic Plagiarism Detection. In: Forner, P., Müller, H., Paredes, R., Rosso, P., and Stein, B. (eds.) CLEF 2013, LNCS, vol. 8138. pp. 53–58. Springer, Heidelberg .

[57]- Adam Kilgarriff, Vít Baisa, Jan Bušta, Miloš Jakubíček, Vojtěch Kovář, Jan Michelfeit, Pavel Rychlý, Vít Suchomel. The Sketch Engine: ten years on. *Lexicography*, 1: 7-36, 2014.

[58]- Heiden, S., Magué, J-P., Pincemin, B. (2010a). TXM : Une plateforme logicielle open-source pour la textométrie – conception et développement. In Sergio Bolasco, Isabella Chiari, Luca Giuliano (Ed.), Proc. of 10th International Conference nous the Statistical Analysis of Textual Data - JADT 2010) (Vol. 2, p. 1021-1032). Edizioni Universitarie di Lettere Economia Diritto, Roma, Italy.

[59]- Hardie, Andrew. (2009). CQPweb - Combining power, flexibility and usability in a corpus analysis tool. International Journal of Corpus Linguistics. 17. 10.1075/ijcl.17.3.04har.

[60]- Baeza-Yates and Ribeiro-Neto, 1999

[61]- Baeza-Yates, R. A. and Ribeiro-Neto, B. (1999). Modern Information Retrie- val. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[62]- Karl Pearson (20 June 1895) "Notes nous regression and inheritance in the case of two parents," Proceedings of the Royal Society of London, 58 : 240–242.

[63]- Hamers, L., Hemeryck, Y., Herweyers, G., Janssen, M., Keters, H., Rousseau, R., et al. (1989). Similarity Measures in Scientometric Research: The Jaccard index versus Salton's cosine formula. Information Processing & Management, 25(3), 315-318.

[64]- Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady, 10(8) :707□710.

[65]- Thorvald Sørensen, « *A method of establishing groups of equal amplitude in plant sociology based nous similarity of species and its application to analyses of the vegetation nous Danish commons* », Biologiske Skrifter/Kongelige Danske Videnskabernes Selskab, vol. 5, nº 4, 1948, p. 1–34

[66]- Christiane Fellbaum (1998, ed.) WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.

[67]- Leacock, C. and Chodorow, M. (1998). Combining local context and wordnet simila- rity for word sense identi□cation. In Fellfaum, C., editor, MIT Press, pages 265□283, Cambridge, Massachusetts

[68]- Wu, Z. and Palmer, M. (1994). Verbs semantics and lexical selection. In Proceedings of the 32nd annual meeting nous Association for Computational Linguistics, ACL '94, pages 133□138, Stroudsburg, PA, USA. Association for Computational Linguistics.

[69]- Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In IJCAI 1995, pages 448□453.

[70]- Lin, D. (1998). An information-theoretic de□nition of similarity. In In Proceedings of the 15th International Conference nous Machine Learning, pages 296□304. Morgan Kaufmann.

[71]- Jiang, J. J. and Conrath, D. W. (1997). Semantic similarity based nous corpus statistics and lexical taxonomy. CoRR, cmp-lg/9709008.

[72]- Bisson, G. and Hussain, S. F. (2008). Chi-sim : A new similarity measure for the co-clustering task. In ICMLA, pages 211□217.

[73]- Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In In Proceedings of the 20th International Joint Conference nous Arti□cial Intelligence, pages 1606□1611.

[74]- H. Soori, M. Prilepok, J. Platos, E. Berhan and V. Snasel, "Text similarity based nous data compression in Arabic", Lecture Notes in Electrical Engineering , no. DOI: 10.1007/978-3-642-41968-3_22, October 2013

[75]- Mohammad A. Al-Ramahi, Suleiman H. Mustafa, "N-Gram-Based Techniques for Arabic Text Document Matching; Case Study: Courses Accreditation", Abhath AL-Yarmouk Basic Sci. & Eng, vol. 21, no. 1, pp. 85-105, 2012.

[76]- Ashraf S. Hussein, "Visualizing Document Similarity Using N-Grams and Latent Semantic Analysis", in Proceedings of SAI Computing Conference, London, UK, 2016, pp. 269-279.

[77]- H. Froud, R. Benslimane, A. Lachkar , A. Lachkar,S. A. Ouatik, "Stemming and Similarity Measures for Arabic Documents Clustering", in 2010 5th International Symposium nous I/V Communications and Mobile Network, Rabat, 2010, pp. 1-4

[78]- Ali Selamat, Hanadi Hassen Ismail, "Finding English and Translated Arabic Documents Similarities Using GHSOM", in Proceedings of the International Conference nous Computer and Communication Engineering 2008, Kuala Lumpur, Malaysia, 2008, pp. 460-465.

[79]- Susan T. Dumais (2005). "Latent Semantic Analysis". Annual Review of Information Science and Technology. 38: 188–230

[80]- Hofmann, T. (1999). Probabilistic latent semantic indexing. In Proceedings of the 22nd annual international ACM SIGIR conference nous Research and development in information retrieval, SIGIR '99, pages 50□57, New York, NY, USA. ACM.

[81]- Brian Riordan, Michael N. Jones. “Redundancy in Perceptual and Linguistic Experience: Comparing Feature-Based and Distributional Models of Semantic Representation” . pages 1-338 2009.

[82]- Belkacem, Thiziri & Dkaki, Taoufiq & Moreno, Jose & Mohand, —. (2017). Apprentissage de représentations de documents et leur exploitation en recherche d'information.

[83]- https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[84]- Mikolov T., Le Q. V., Sutskever I., « Exploiting similarities among languages for machine trans-lation », arXiv preprint arXiv :1309.4168, 2013

[85]- Atoub Yasmine, Benayad Asma. “Mesures de similarité sémantique pour un système d’évaluation automatique des réponses courte: Application à la langue arabe”. Mémoire master USDB1. 2017/2018

[86]- Zahran, Mohamed & Magooda, Ahmed & Mahgoub, Ashraf & Raafat, Hazem & Rashwan, Mohsen & Atyia, Amir. (2015). Word Representations in Vector Space and their Applications for Arabic. 10.1007/978-3-319-18111-0_32.

[87] - Abdallah Amina & Garoudja Khadidja. « Mesures de similarité syntaxique pour un système d’évaluation automatique des réponses courtes : Application à la langue arabe. », Mémoire master USDB1. 2017/2018

[88]- Hannoufi Mohammed Hamza, Henniche Adel Nassim. « Les Word-Embedding pour l’évaluation automatique des réponses courtes en apprentissage en ligne : Application à la langue arabe ». Mémoire master USDB1. 2017/2018

[89]- M. Hadjersi, O. Benguergoura, « Le machine Learning pour l’Evaluation Automatique des Réponses Courtes : Application à la Langue Arabe ». Mémoire master USDB 1. 2019/2020.

[90]- S. Abdellaoui, « Généralisation d’approches basées Espace Sémantique pour un système d’évaluation des réponses courtes adapté à la langue arabe ». Mémoire master USDB 1. 2018/2019.