الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جامعة عمار ثليجي بالاغواط
Université Ammar Telidji Laghouat
كلية العلوم
Faculté des Sciences
قسم الإعلام الآلي
Département d'Informatique

Mémoire de fin d'étude
pour l'obtention de diplôme de Master en informatique.

Domaine : Mathématiques et Informatique.
Filière : Informatiques.
Option : Réseaux, systèmes et applications réparties.

Réalisé par:

Ferhat Hamza

Thème

Réalisation d'une méthode de détection de plagiat au sein d'une plateforme dédiée

---

Soutenu le 29/09/2020, devant le jury composé de :

Mme. Hadda Cherroun	Prof.	Université Laghouat	`Rapporteur`
M. Younes Guellouma	MCA.	Université Laghouat	`Président`
M. Nehar Attia	MCB.	Université Djelfa	`Examinateur`

Nº d'ordre ..... Année universitaire 2019/2020

Remerciements

Tout d'abord, nous remercions notre Allah de nous avoir donné la force et le courage pour achever ce travail.

Le travail réalisé tout au long de ce mémoire n'a été possible qu'avec l'aide et la soutien de nombreuses personnes. Je profite de cette occasion qui m'est donnée pour leur exprimer toute ma gratitude.

Nous remercions le corps administratif de notre département et tous nos professeurs qui ont été la raison de notre succès.

Nous remercions tout particulièrement notre enseignante ***Mme. Hadda Cherroun*** pour avoir eu confiance en moi et pour m'avoir donné l'opportunité d'effectuer ce travail. Je les remercie pour leurs disponibilités, leurs patiences et leurs précieux conseils.

Ferhat Hamza

Dedication

À mes tres chers parents ...
Avec tout mon amour et tout mon effection, je vous remercie de m'avoir guidée, conseillée et cru en moi, et surtout d'avoir fait de moi ce que je suis devenu Ajourd'hui. Merci Papa, et merci maman pour l'aide et le soutien que tu m'as octroyés pour obtenir mon master, j'espére que ce travail vous rendra encore plus fiers.

À mon oncle ...
Je dédie ce travail à mon oncle ***Moubarek sellami*** la raison après Allah pour atteindre ce succès. Merci beaucoup.

À toute ma famille ...
À tous mes amis ...

Ferhat Hamza

Abstract

In this work, we are interested in the problem of plagiarism detection, which is continually worsening, due to the increasing use of modern communication technology. Fighting against such practices may seem to be a simple check of similarity between documents, something that machines are supposed to perform well. Indeed, plagiarism detection is a challenging natural language processing task.
The most recent achievements are systems able to detect the simple verbatim reproduction (copy-paste). On the other hand, plagiarism techniques are evolving to more complex forms such as synonyms substitution, text manipulation, paraphrasing, and even translation. This makes the plagiarism detection much more challenging than a simple comparison of pieces of text. While the issue is being heavily investigated in various languages, very few research works were conducted on Arabic.

In this work, we describe the conception and the realization that we have conducted into a large framework. this framework aims to provide a toolbox dedicated to the detection of plagiarism. it's final user are mainly researchers and academic actors who are interested to have such platform.

Our contribution is implementation of a recent plagiarism detection method [NCS18]. The implementation relies on object-oriented modeling paradigms while using open source tools and languages.

Keywords: Textual Plagiarism, Plagiarism Detection, Natural Language Processing, 2L-ADP, Toolbox.

ملخص

في هذا العمل نهتم بمشكلة كشف السرقة الأدبية والتي تتفاقم باستمرار بسبب الاستخدام المتزايد لتكنولوجيا الاتصالات الحديثة. قد يبدو أن محاربة مثل هذه الممارسات مجرد فحص للتشابه بين المستندات ، وهو أمر يفترض أن تؤديه الآلات بشكل جيد. في الواقع ، يعد اكتشاف الانتحال مهمة صعبة لمعالجة اللغة الطبيعية.

أحدث الإنجازات هي الأنظمة القادرة على اكتشاف النسخ الحرفي البسيط (نسخ ولصق). من ناحية أخرى ، تتطور تقنيات الانتحال إلى أشكال أكثر تعقيدا مثل استبدال المرادفات ، والتلاعب بالنص ، وإعادة الصياغة ، وحتى الترجمة. هذا يجعل اكتشاف السرقة الأدبية أكثر صعوبة من مقارنة بسيطة لأجزاء من النص. بينما يتم التحقيق في هذه القضية بشكل مكثف بلغات مختلفة ، تم إجراء عدد قليل جدا من الأبحاث حول اللغة العربية.

في هذا العمل ، نصف التصميم والإنشاء الذي أجريناه في إطار عمل كبير. يهدف طار عمل هذا إلى توفير صندوق أدوات مخصص لاكتشاف الانتحال. المستخدم النهائي هو في الأساس باحثون وفاعلون اكاديمون مهتمون بامتلاك مثل هذه المنصة.

ومساهمتنا هي تصميم طريقة حديثة للكشف عن سرقة أدبية. يعتمد التنفيذ على نماذج كائنية التوجه باستخدام الأدوات مفتوحة المصدر.

الكلمات المفتاحية : الانتحال، كشف السرقة الأدبية، المعالجة الآلية للغة الطبيعية، صندوق أدوات.

Résumé

Dans ce travail, nous nous intéressons au problème de la détection du plagiat, qui s'aggrave, en raison de l'utilisation croissante des technologies modernes de communication. La lutte contre ces pratiques peut sembler être une simple vérification de la similitude entre les documents, ce que les machines sont censées faire avec succès. En effet, la détection du plagiat est une tâche difficile de traitement du langage naturel. Les réalisations les plus récentes sont des systèmes capables de détecter la simple reproduction textuelle (copier-coller). D'autre part, les techniques de plagiat évoluent vers des formes plus complexes telles que la substitution de synonymes, la manipulation de texte, la paraphrase et même la traduction. Cela rend la détection du plagiat beaucoup plus difficile qu'une simple comparaison de morceaux de texte. Alors que la question est largement étudiée dans diverses langues, très peu de travaux de recherche ont été menés sur l'arabe.

Dans ce travail, nous décrivons la conception et la réalisation que nous avons menées dans un grand cadre de travail. Ce travail de cadre vise à fournir une boîte à outils dédiée à la détection du plagiat. ses utilisateurs finaux sont principalement des chercheurs et des acteurs universitaires qui sont intéressés à avoir une telle plateforme.

Une contribution est en cours de réalisation pour la mise en place d'une méthode récente de détection du plagiat[NCS18]. La mise en œuvre repose sur des paradigmes de modélisation orientés objet tout en utilisant des outils à code source ouvert.

Mot clés: Plagiat textuel, Détection du plagiat, Traitement Automatique de la Langue, 2L-APD ,Boîte à outils.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

2L-APD	Système de détection du plagiat arabe à deux niveaux
AEF	Automate à états finis
Aplag	International Standard Book Number
IR	Information retrieval
ISBN	International Standard Book Number
NLTK	Natural Language Toolkit
TALN	Traitement automatique du langage naturel
VScode	Visual Studio Code

Introduction générale

L'avènement et la prolifération des nouvelles technologies ont augmenté l'incidence du plagiat dans les milieux académique. En outre, ces dernières années, l'expansion d'Internet facilite également l'accès aux documents du monde entier (rédigés en langues étrangères) et à des outils de traduction automatique de plus en plus puissants, ce qui accentue la progression d'un nouveau type de plagiat : le plagiat translingue. Cela a incité de nombreux chercheurs à tenter de trouver une solution à ce problème.

Dérivé du latin "plagiarius" qui signifie "kidnappeur, séducteur, voleur littéraire" [MPB10]. Et aussi du mot anglais "plagiary" est "celui qui prend injustement les mots ou les idées de quelqu'un".

La détection automatique du plagiat a fait l'objet d'une attention particulière dans le cadre de la mise au point de systèmes de détection du plagiat à petite et à grande échelle comme contre-mesure possible. Dans le cas d'un document textuel, la tâche d'un système de détection de plagiat est de découvrir si le document est copié en tout ou en partie à partir d'autres documents sur le Web ou de tout autre dépôt [KSM15].

Dans la littérature dédiée à la détection du plagiat, il y a eu une panoplie de méthodes proposées ainsi que d'outils crées dans ce contexte. Mais parmi ceux-la, il y a un manque d'outils qui supportent la langue arabe.

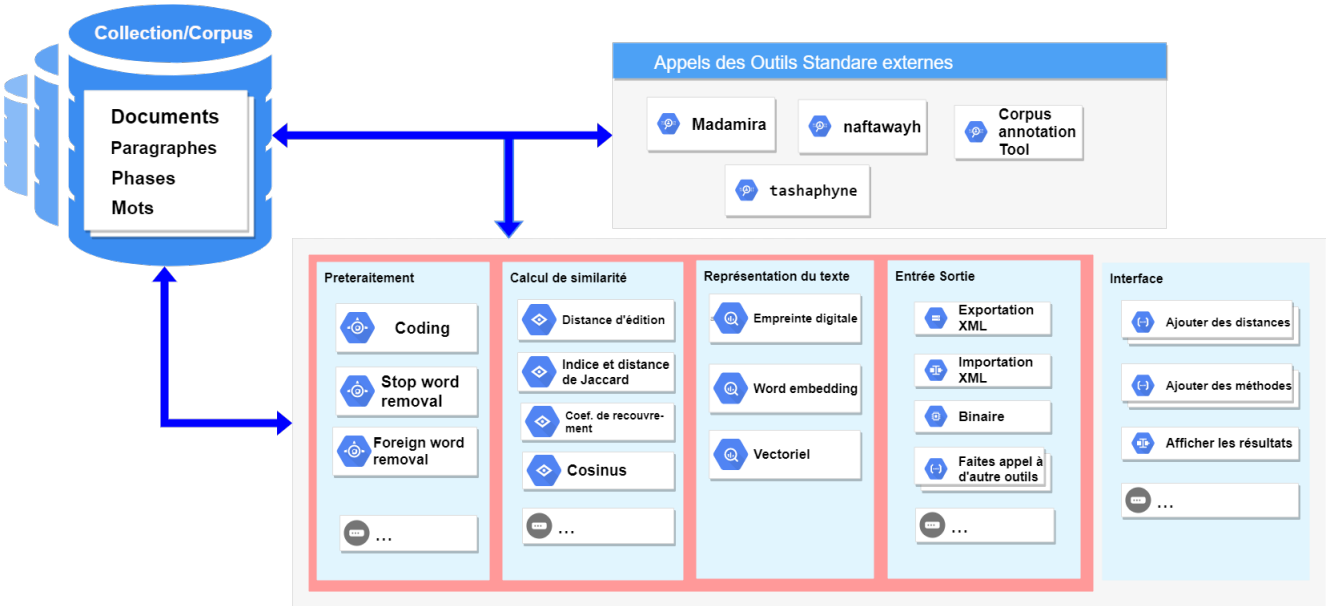

FIGURE 1 – Aperçu du projet global

Objectifs de la recherche

Notre travail de recherche rentre dans le cadre d'un projet plus grand qui est le développement d'une plateforme dédiée à la détection du plagiat. Les utilisateurs de cette plateforme peuvent être essentiellement des chercheurs, des étudiants et des professionnels du domaine académique.

Les buts principaux de ce projet global (voir Figure 1) sont nombreux :

— Introduire et présente des informations générales sur le plagiat et ses formes.

— Mettre en évidence le domaine de la détection automatique du plagiat et des méthodes utilisées, et le manque d'applications supportant l'arabe.

— Essentiellement concevoir et puis implémenter l'essentiel d'une boite à outil dédiée aux chercheurs s'intéressant à la détection de plagiat. Les fonctionnalités de cette boite à outil doivent être modulaires pour permettre aux chercheurs d'introduire de nouvelles.

— Offrir aux chercheurs une collection de ressources textuelles pour pouvoir valider et tester leurs méthodes méthode.

— Implémenter et intégrer certaines méthodes de détection de plagiat de la littérature.

Notre contribution dans ce grand projet est la participation dans la conception de la boite à outils et plus spécialement le développement d'une méthode récente de détection de plagiat [NCS18]. En effet, cette méthode a fait ses preuves mais nécessite une implémentation plus efficace au sein d'une plateforme dédiée.

Organisation du mémoire

Nous avons divisé ce manuscrit en trois chapitres selon le plan méthodologique suivant :

Le premier chapitre donne un aperçu général sur le plagiat textuel, quelques définitions, les types de plagiat et la description de certains termes et concepts associés.

Le deuxième chapitre Dans ce chapitre on présente et on décrit certains logiciels et méthodes de plagiat existants dans la littérature dédiée à la détection de plagiat. Le but et de comprendre leurs fonctionnement afin d'en choisir ceux à rajouter à la plateforme.

Le troisième chapitre Dans ce chapitre, nous allons décrire une méthode récente de détection de plagiat [NCS18]. Cette méthode fait appel à beaucoup de concepts tel que le word embedding, les distances de similarité entre textes faisant appel aux informations sémantiques et lexicales.

Chapitre 1

Généralités sur le plagiat et les outils existants

Le plagiat et les facilités qu'il induise, tente une grande majorité d'étudiants, chercheurs et même professeurs [1], et avec la croissance de ce phénomène dans la communauté scientifique, de nombreuses questions se posent. Dans ce chapitre, nous répondrons aux trois questions les plus importantes :

Q'est-ce que le plagiat ?
Quels sont ses types ?
Quels sont les outils existant ?

1.1 Définitions et terminologies

A fin de bien illustrer le plagiat et ses variantes, nous présentons dans cette section ses définitions avec une description de certains des termes associés à ce domaine.

1.1.1 Plagiat

Le dictionnaire en ligne d'Oxford [2] définit le plagiat comme : "la pratique de prendre les idées ou le travail d'un autre et de les faire passer comme étant les siennes".

Le dictionnaire en ligne Larousse [3], définit également le plagiat comme :"Acte de quelqu'un qui, dans le domaine artistique ou littéraire, donne pour sien ce qu'il a pris à l'œuvre d'un autre."

De plus, selon Fishman [Fis09], le plagiat est défini comme suit : " le plagiat est l'utilisation des idées, des concepts, des mots ou des structures et les intégrer à son propre travail sans en mentionner la provenance ".

À partir des définitions ci-dessus, nous concluons que l'utilisation de tout travail, qu'il s'agisse de notre travaux précédent ou des travails d'autres personnes sans mentionner la source, est considérée comme du plagiat. Il est donc important de définir certains termes liés à ce domaine.

1. www.lexpress.fr (consulté le 16/04/2020)
2. www.lexico.com (consulté le 16/04/2020)
3. www.larousse.fr (consulté le 16/04/2020)

1.1.2 Référence

Selon Baudry [Fre18], une référence est une "action de se référer ou de renvoyer à un article, à un passage, à une chose ayant quelques rapports". Référencer une source dans un document représente un indicateur d'existence de source d'une information, texte, parole ou autre, ayant un rapport direct ou indirect avec ce document. La référence doit contenir suffisamment d'information pour identifier de façon unique les sources telles que : un article, un livre, ou tout autre document. Ainsi, une référence doit contenir : un numéro (comme [12] ou bien [Tarzi, 2018]), un nom du/des auteurs, un titre, le nom de l'éditeur, et la date de parution , et le numéro international normalisé du livre (ISBN) (dans le cas d'un livre). Dans le cas d'un article : le nom du/des auteurs, le titre de l'article, le titre de la revue, le volume, le numéro de l'édition, la date et la page [Nag17].

1.1.3 Citation

Millet'[Oli97], définit une citation comme étant : "un fait de parole (ou d'écriture), par définition individuelle et unique, qui est repris comme tel -cité- par un autre locuteur ou une infinité de locuteurs".

Par ailleurs, l'action cite est définie dans le dictionnaire Larousse comme : "Action de citer, de rapporter les mots ou les phrases de quelqu'un ; paroles, passage empruntés à un auteur ou à quelqu'un qui fait autorité." [4]. Donc, une citation est la reproduction d'un extrait d'un texte ou un écrit antérieur dans la rédaction d'un nouveau texte.

1.1.4 Bibliographie

La bibliographie est une liste structurée des sources citées (références), notamment des livres, des ouvrages, des articles de journaux, ou autres documents utilisés pour la préparation d'un document scientifique. La bibliographie se trouve généralement à la fin d'un article de journal, d'un livre ou d'un article d'encyclopédie [ATR06].

1.2 Plagiat textuel

Le phénomène du plagiat s'est accru dans la communauté universitaire et la recherche dans ce domaine a considérablement progressé, ce qui a donné lieu à de nombreux articles et recherches sur les définitions et les types de plagiat et leur prévention. Ce qui est important ici, c'est le plagiat du texte, et c'est le plagiat impliquant la réutilisation d'un ouvrage écrit (travail écrit) sans mention de la source.

Sur la base de ces recherches, le plagiat se présente sous de nombreuses formes, telles que [Nag17] :

— Présenter comme sien un travail original de quelqu'un d'autre.

— Intégrer dans un travail des passages de textes, des données, des résultats expérimentaux ou même des images provenant de sources externes sans en mentionner la source.

— Résumer une idée créative de quelqu'un d'autre en l'exprimant avec d'autres mots tout en omettant d'en mentionner la provenance.

4. www.larousse.fr (consulté le 16/04/2020)

1.3 Formes de plagiat textuel

Dans cette section, nous présentons différentes formes de plagiat textuel et quelques exemples pour chacune d'entre elles.

1.3.1 Duplication

La duplication est la copie directe de phrases ou de passages d'un texte publié sans citation, également appelée copier/coller [Clo03]. L'exemple suivant illustre un cas de duplication d'un passage *wikipedia* sans mention.

FIGURE 1.1 – Exemple de plagiat de duplication

1.3.2 Substitution par des synonymes

Remplacer des mots par des synonymes est une forme de plagiat. Et de son nom, elle a utilisé des synonymes pour paraphraser des expressions. La figure 1.2 montre la différence après avoir reformulé une phrase en arabe en utilisant des synonymes.

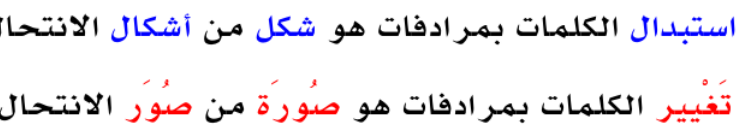

FIGURE 1.2 – Exemple de plagiat en remplaçant des mots par des synonymes

1.3.3 Reformulation paraphrastique

La paraphrase ou la reformulation paraphrastique, consiste à reprendre un texte original sans altération de son contenu en utilisant le changement de son vocabulaire par l'ajout, la suppression ou la substitution de mots par des synonymes [Nag17]. La figure 1.3 illustre la différence après avoir paraphrasé une phrase en arabe.

1.3.4 Traduction

Le plagiat par traduction aussi appelé plagiat translingue consiste à faire une transformation manuelle ou automatisée d'un texte original d'une langue à une autre sans mentionner la source [Deb10]. Par exemple traduire la définition de TALN [5] du français vers l'anglais comme l'illustre la figure 1.4 .

5. https://fr.wikipedia.org (consulté le 27/04/2020)

L'origine, par wikipedia	L'emprunt, par exemple
يمكن أن يكون البحث عن المعاني باللغة العربية مفيدًا لفهم السياق بطريقة فعالة. يمكنك الحصول على أكثر من ترجمة لكلمة واحدة باللغة العربية.	البحث عن معاني الكلمات بالغة العربية يساعد على فهم سياق ما فهما عميقا. يمكن ايجاد العديد من المعاني لكلمة واحدة بالغة العربية.

FIGURE 1.3 – Exemple de plagiat par reformulation paraphrastique

Définition originale en français	Définition traduite en anglais
Le traitement automatique du langage naturel, abrégé en TALN, est une discipline s'appliquant au domaine de l'informatique et du langage. Il est utilisé par exemple pour les traductions, la reconnaissance vocale ou encore les réponses automatiques aux questions[a].	The automatic processing of natural language, abbreviated to NLP, is a discipline that applies to the field of computer science and language. It is used, for example, for translations, voice recognition or automatic answers to questions.

a. fr.wikipedia.org. (consulté le 27/04/2020)

FIGURE 1.4 – Exemple de cas de plagiat translingue

1.3.5 Fantôme littéraire

Fantôme littéraire appelé aussi *ghostwriting* est une autre forme de fraude académique, souvent utilisée par les étudiants dans le domaine universitaire. L'étudiant paie une personne tierce ou une entreprise sur le net ou cybercafé pour lui faire ses devoirs écrits, réaliser son mémoire de master ou même sa thèse de doctorat [Nag17, FNBF17].

1.3.6 Assemblage

L'assemblage est une copie de passages provenant de sources multiples et leur mélange dans la nouvelle œuvre sans citation. Également connu sous le nom de "patchwork" [Nag17].

1.3.7 Auto-plagiat

Un auto-plagiat est, par définition, un emprunt à soi-même (de ses travaux antérieurs) sans y faire référence. La figure 1.5 illustre un auto-plagiat effectué par le docteur physicien Étienne Klein [6].

1.4 Logiciel de détection de plagiat

Dans cette section, nous listons quelques outils de détection automatique du plagiat, en nous concentrant sur ceux qui supportent la langue arabe. L'étude de ces outils repose

6. https://blogs.mediapart.fr (consulté le 27/04/2020)

L'origine, par Etienne Klein

Dans nos contrée, la métaphore du fleuve a accompagné presque toute l'histoire de la pensée du temps et continue d'irriguer notre façon de l'évoquer et de le représenter : ... au temps même les propriétés de la ligne par laquelle on le représente. Kant l'avait déjà vu :"Nous représentons la suite de temps par une ligne qui se ... second sont toujours successives.".[a]

a. "Le temps est-il affaire de conscience ?", Etienne klein, European Psychiatry, Novembre

L'emprunt, par Etienne Klein

la métaphore du fleuve a accompagné presque toute l'histoire de la pensée du temps - du moins en Occident - et continue d'irriguer notre façon de l'évoquer et de le représenter : ... au temps les propriétés de la ligne par laquelle on le représente. En d'autre termes, le fait de d'écrire le temps par une ligne lui confére i[sp facto des "probléme de ligne" Kant l'avait reléve :"Nous représentons la suite de temps par une ligne qui se ... second sont toujours successives.".[a]

a. "L'instant présent, unique mais banal",Etienne Klein,Pour La Science,octobre 2010.

FIGURE 1.5 – Un exemple d'un auto plagiat effectué par le docteur physicien Étienne Klein.

sur leur fonctionnement et leur popularité.

1.4.1 L'outil Aplag

Il s'agit d'une abréviation pour Aplag et est considéré comme un logiciel de détection du plagiat pour les textes en arabe. Il est publié par le département d'informatique de l'université du Roi-Saoud, année 2011. Il s'appuie sur la représentation logique des textes sous forme de paragraphes, de phrases et de mots afin que chaque phrase et chaque mot prenne des nombres entiers qui l'expriment dans l'ordre où ils apparaissent dans le texte [EBMB11].

1.4.2 L'outil Turnitin

Turnitin[7], l'un des plus anciens logiciels de détection de plagiat publié en 1998, par un groupe de chercheurs de l'université de Californie "UC Berkeley". Et il est devenu un programme affilié de *iParadigms* LLC(*Limited Liability Company*).

Turnitin maintient une base de données de comparaison qui comprend :

— Le contenu web actuel et archivé qui est accessible au public.

— Livres, journaux et revues (grâce à ses partenariats avec des éditeurs, des bases de données des bibliothèques, des collections de référence numériques et des publications par abonnement).

— Documents d'étudiants envoyés à Turnitin.

Turnitin utilise un algorithme de comparaison pour trouver des chaînes de mots identiques à celles de son entrepôt de données. Cela signifie que le jugement humain, basé sur divers facteurs et considérations, est nécessaire pour déterminer si un cas de plagiat s'est

7. www.turnitin.com

produit. Turnitin ne peut donc créer que des rapports d'originalité qui montrent le degré de similarité entre un travail soumis et les sources de contenu de la base de données [8].

1.4.3 L'outil Urkund

Selon le site web d'Urkund : "Urkund est un système entièrement automatique d'apprentissage automatique de reconnaissance de texte conçu pour détecter, prévenir et gérer le plagiat, quelle que soit la langue dans laquelle vous écrivez".

Urkund analyse les documents envoyés à l'aide de l'apprentissage automatique, et détecte non seulement les similitudes avec d'autres sources, mais aussi l'utilisation de paraphrases et de substitutions par des synonymes [9].

1.4.4 L'outil PlagScan

PlagScan est un programme commercial (2000 premiers mots gratuits) de détection du plagiat de texte traite toute langue qui utilise le système de codage scientifique UTF8, donc il traite des textes arabes. Il est principalement basé sur la recherche sur Internet (Microsoft Bing) et dans les bases de données des éditeurs (articles scientifiques, revues académiques), et en option il peut rechercher dans n'importe quelle base de données définie par l'utilisateur [10].

1.4.5 L'outil Plagiarisma

Plagiarisma est la plus simple application web de détection de plagiat. La version gratuite comporte un nombre limité de vérifications de plagiat, et plus de 190 langues sont supportées entre elles l'arabe. Il vous permet de le faire : Copier/Coller ou type texte dans le champ approprié afin qu'il soit vérifié, vérifier l'URL fournie, ou le fichier téléchargé de votre ordinateur [11].

Plagiarisma recherche les doublons de votre texte dans Google, Yahoo, Bing, Scholar et Books. Si vous n'utilisez aucun de ces moyens, il est impossible pour Plagiarisma de savoir si vous êtes en train de plagier.

1.4.6 L'outil Copyleaks

Copyleaks est un service de détection de plagiat en ligne. vérifier le texte et le contenu de sites depuis une URL. il fournit un ensemble d'algorithmes de détection et utilise l'API Web de Google pour enrichir ses recherches [CB18].

— Ne Possède pas sa propre base de données.

— C'est un outil payent pour la vérification textuelle et URL mais qu'il dispose d'une version limitée gratuite pour la vérification de 9 essais.

8. www.algonquincollege.com (consulté le 06/05/2020)
9. www.urkund.com(consulté le 06/05/2020)
10. www.plagscan.com(consulté le 06/05/2020)
11. Site de Plagiarisma(consulté le 11/03/2020)

1.4.7 L'outil PlagAware

PlagAware est un moteur de recherche sur le plagiat. Il utilise le moteur de recherche classique (googl, bing ...) pour détecter et scanner le plagiat, et fournit différents types de rapports qui aident l'utilisateur ou le propriétaire du document à décider si son document a été plagié ou non.

1.4.8 Discussion

Nous avons présenté dans la section précédent les différents systèmes de la littérature autour de la détection de plagiat dans les documents en langue arabe. Le tableau 1.1 présente une synthèse de ces systèmes de détection du plagiat extrinsèques décrits en fonction des critères : la technique utilisée, le niveau de comparaison et leur efficacité dans la détection de différents types de plagiat.

D'après le tableau 1.1, on constate qu'il y a un manque de systèmes qui traitent les problèmes de détection du plagiat avec reformulation.

	supporte l'arabe	Web/ Desk./ LC	Perfermance	Gratuit/payant
Turnitin	Oui*	App. web	●●○○○**	-
Urkund	indé.de la langue*	App. web	●●●○○**	-
PlagScan	Oui*	App. web	●●○○○**	8 Pages gratuites
Plagiarisma	Oui*	Web / Desk	●●○○○**	Vér. limitées / jour
Copyleaks	Oui*	App. web	●●○○○**	10 pages gratuites
PlagAware	Oui*	App. web	●●○○○**	1365 mots gratuits

TABLE 1.1 – Systèmes de détection du plagiat extrinsèque pour les documents en Arabe.

* *: Ils ne disposent pas d'un corpus arabe ; ils vérifient le document arabe en utilisant un moteur de recherche (Google, Bing ..).*

** *: Nous avons essayé la version gratuite et la version payante est peut-être plus performante.*

1.5 Conclusion

Dans ce chapitre, nous avons recensé un ensemble de définitions liées aux plagiat textuel ainsi que certains termes et concepts associés. De plus nous avons rapportés certaines de ses formes importantes et certains systèmes plus connus. Enfin, de nombreuses recherches ont été menées sur le plagiat, le définissant et le classant chacun à sa manière, mais avec la large diffusion de ce phénomène dans le domaine scientifique, il est nécessaire de trouver des moyens de le prévenir et de le contrôler et surtout de le détecter.

Chapitre 2

Détection de plagiat dans les documents arabes

Dans ce chapitre, nous allons commencer par recenser, parmi les plus récentes, les méthodes de détections de plagiat. Nous nous concentrerons en suite sur la description des méthodes qui ont considérées la langue arabe et ses particularités.

L'objectif essentiel est de comprendre le fonctionnement de ces méthodes et choisir celles qui méritent d'être implémenter et intégrer dans notre boite à outils.

2.1 Introduction

De nombreuses entreprises spécialisées dans le domaine de l'informatique ont été capables de développer des logiciels et des applications web permettant de comparer des travaux scientifiques et d'enquêter sur leur authenticité, en interrogeant des bases de données et en parcourant divers sites Internet, dans le but de révéler des pratiques non éthiques dans le cadre de la recherche scientifique et de réduire les effets du vol scientifique (plagiat), mais la langue arabe est considérée comme l'une des langues les moins prises en charge dans les logiciels de détection de plagiat en raison de sa spécificité, tels que la grammaire, conjugaison et autres ; qui a rendu la question de sa numérisation très difficile.

Le manque et l'imprécision des outils de découverte du plagiat à l'appui de la langue arabe ont conduit à l'émergence de certaines initiatives arabes pour lutter contre le vol scientifique (plagiat) dans la recherche arabe. Dans ce chapitre, nous allons discuter de certaines de ces initiatives, en commençant par citer les méthodes de détection du plagiat en langue arabe.

2.2 Techniques de détection des plagiats de textes

On distingue deux catégories de techniques de détection de plagiat : intrinsèque et Extrinsèque. La figures 2.1 illustre ces catégories.

La détection de plagiat intrinsèque consiste à examiner statistiquement les caractéristiques linguistiques d'un document par rapport à lui-même sans avoir besoin de sources externes. Il s'agit de trouver les changements dans les styles d'écriture, et les utilise comme indicateurs d'un plagiat potentiel.

La détection du plagiat extrinsèque, d'autre part, compare un document suspect à une collection de documents de référence comprenant un ensemble de sources probables

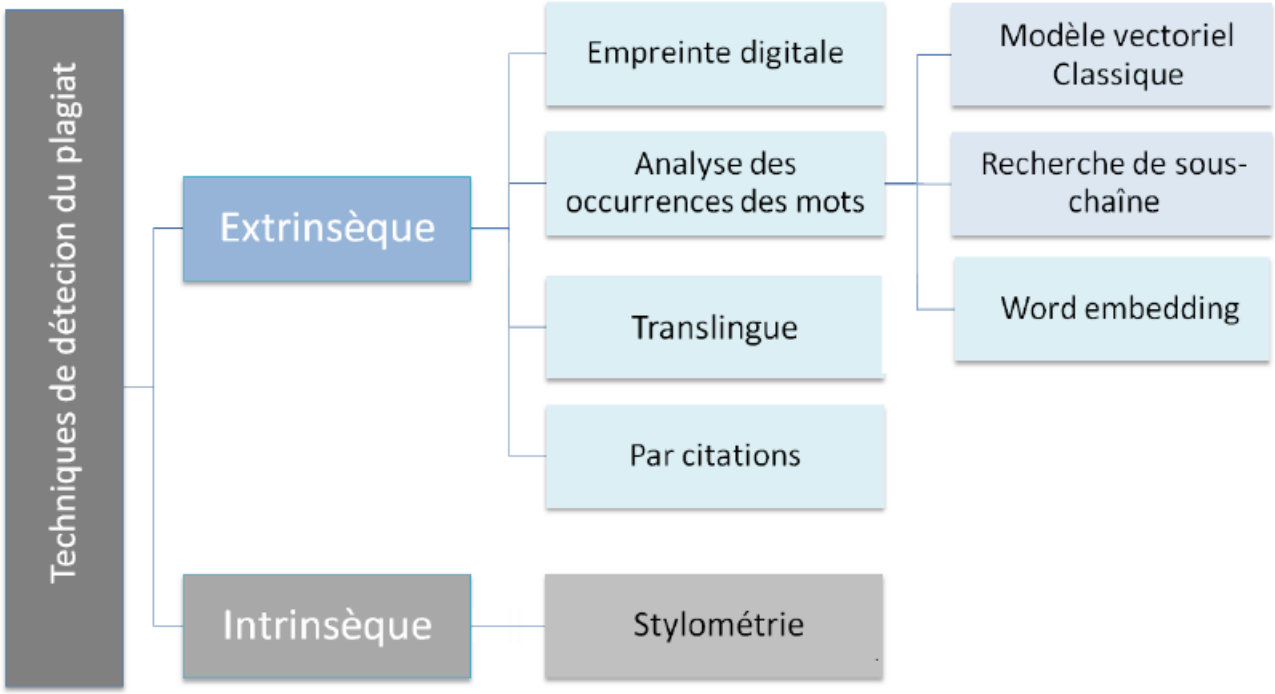

FIGURE 2.1 – Techniques de détection des plagiats de textes [Nag17]

de copie. L'objectif est de vérifier si certaines parties de cette collection sont similaires à des segments de texte du document suspect.

Dans notre travail, on s'intéresse qu'au deuxième type à savoir plagiat *Extrinsèque*.

2.2.1 Empreint Digitale

En informatique, un algorithme d'empreinte digitale (en anglais fingerprinting) est une procédure qui associe un élément de données arbitrairement grand (tel qu'un fichier informatique) à une chaîne de bits beaucoup plus courte, son empreinte digitale, qui identifie de façon unique les données originales, tout comme les empreintes digitales humaines identifient de façon unique les personnes. Cette empreinte digitale peut être utilisée à des fins de déduplication des données. C'est ce que l'on appelle également l'empreinte digitale de fichier, l'empreinte digitale de données ou l'empreinte digitale de texte.

Ces algorithmes sont utilisés dans la détection du plagiat, ils divisent généralement le document en séquences (appelés grammes) de longueur n, de sorte que les empreintes digitales de deux documents peuvent être comparées et les passages similaires (c'est-à-dire les grammes) identifiés comme des passages identiques dans les textes.

Iqtebas

Ameera et Elnagar [JE12] présentent un système de détection du plagiat textuel dans les textes arabes basé sur la technique d'empreinte digitale. Ce système est nommé Iqtebas qui vient du mot arabe إقتباس (citation). Iqtebas estime le taux de plagiat entre deux documents de la façon suivante :

1. Une première étape de pré-traitement des documents est faite. Elle inclut la suppression de mots vides et la racinisation des mots ;
2. La segmentation des documents (suspects et sources) en phrases et la génération des 3-grammes de mots pour chaque phrase ;
3. L'application de la fonction de hachage Karp-Rabi [KR87] sur les 3-grammes de mot ;
4. Afin de construire l'empreinte digitale, Iqtebas utilise l'algorithme winnowing [SSRA03] pour sélectionner un sous ensemble de hashes ;
5. La construction d'un index inversé [CDM08] pour accélérer le temps de calcul de similarité, de façon que le dictionnaire de l'index contient l'empreinte de chaque 3-gramme et la liste de positions contient tous les identifiants des phrases (phrases_{id}) qui contiennent ce 3-gramme ;
6. Enfin, Iqtebas utilise un Modèle de Fréquence Relative (MFR) [SGM95] et un Modèle de Proportion d'Inclusion (MPI) [JPBL03] pour estimer le degré de similarité entre l'empreinte digitale du document source (ED_{src}) et le document suspect (ED_{sus}).

Malgré le fait que le système *Iqtebas* a marqué un taux de succès intéressant dans la détection de duplication, il reste toutefois inefficace face aux reformulations, la substitution par des synonymes ainsi qu'à la suppression et l'ajout de mots.

Aplag

Parmi les outils de détection du plagiat dans les documents arabes et qui utilisent les empreintes digitales, on trouve (de l'anglais Arabic Plagiarism detection). Meni [Men12] présente ce système de détection afin d'améliorer la détection des plagiat cachés tels que la substitution par des synonymes et la modification de la structure des phrases par l'ajout et la suppression de mots, Aplag est basé sur la technique de l'empreinte digitale et il partage presque les mêmes étapes que Iqtebas à savoir : le pré-traitement, la génération des 3-grammes de mots et le hachage pour la construction des empreintes. Toutefois, Aplag utilise la base de données lexicales Arabic WordNet [BER+06] pour remplacer chaque mot dans le texte par le synonyme le plus fréquemment utilisé. De plus, Aplag propose aussi un modèle heuristique pour comparer le document suspect à différents niveaux hiérarchiques (phrase, paragraphe et document) afin d'éviter les comparaisons inutiles. Les résultats montrent que Aplag [Men12] donne de meilleurs scores dans la détection du plagiat dissimulé par rapport Iqtebas [JE12].

2.2.2 Ensembles Flous

La théorie des *ensembles flous* est une théorie mathématique du domaine de l'algèbre abstraite. Elle a été développée par Lotfi Zadeh [Zad65] en 1965 afin de représenter mathématiquement l'imprécision relative à certaines classes d'objets et sert de fondement à la logique floue.

FS-APD

FS-APD de l'anglais *Fuzzy Statement-based Arabic Plagiarism Detection system* est un système de détection du plagiat en langue arabe qui utilise les techniques des *ensembles flous*. Alzahrani et Salim [AS08] proposent ce système en 2009. Il est le fruit

des travaux visant à adapter le modèle IR flou pour l'utiliser avec la langue arabe afin de calculer le degré de similitude entre deux documents en arabe. Le degré de similarité est comparé à une valeur seuil et deux énoncés peuvent être traités comme égaux s'ils dépassent la valeur seuil. À partir de là, le nombre de énoncés similaires traités comme égaux est calculé, leurs valeurs de similarité sont agrégées, et enfin la valeur moyenne de similarité (AMS) est obtenue, ce qui peut définir la mesure dans laquelle deux documents sont similaires ou dissimilaires. En outre, l'AMS (valeur moyenne de similarité) peut être mise en correspondance avec un ou plusieurs des huit ensembles flous de similarité suivants : double, quasi double, très similaire, similaire, légèrement similaire, dissimilaire, très dissimilaire ou différent. Le processus de détection de plagiat passe par les étapes suivantes :

— Soient doc_i et doc_j un document original et un document suspect, respectivement. doc_i (original) : **السيارة هي إحدى وسائل الموصلات.** (La voiture est un moyen de transport).doc_j (suspect) : **وسائل المواصلات تشمل السيارة والطائرة.** (Les moyens de transport comprennent la voiture et l'avion). La première étape consiste à extraire toutes les paires de mots de doc_i et doc_j comme illustré par la figure 2.2.

Générés < w_i,w_j >Paires
<عربة ، سيارة >
<نقل، سيارة >
< طائرة، سيارة >
<منضدة، سيارة >
...

FIGURE 2.2 – Exemple d'extraction de toutes les paires (w_i,w_j)

— Ensuite, le FS-APD calcule le coefficient de corrélation C_{ij} pour chaque paire de mots. C_{ij} représente le degré de corrélation entre chaque paire de mots (corrélation mot à mot) à partir de la formule 2.1 :

$$C_{ij} = \frac{n_{ij}}{(n_i + n_j - n_{ij})} \tag{2.1}$$

où n_{ij} est le nombre de documents qui ont à la fois les deux termes et n_i, n_j représentent le nombre de documents qui ont w_i, w_j respectivement.Le résultat de cette étape est une matrice de corrélation mot-à-mot pour les mots dans les documents doc_i et doc_j comme illustré dans le tableau 2.1.

— Ensuite, FS-APD calcule le coefficient de corrélation $\mu_{w_i;doc_j}$ entre chaque mot w_i dans le document original doc_i et le document suspect doc_j (une corrélation mot-à-document). La corrélation μ_{ij} est obtenue comme suit :

$$\mu_{w_i,doc_j} = 1 - \prod_{w_k \in doc_j} (1 - C_{ik}) \tag{2.2}$$

— Le taux de similarité entre doc_i et doc_j est obtenu en appliquant la formule 2.3, où n est le nombre de mots dans doc_i :

$$Sim(doc_i, doc_j) = \frac{\sum_{k=1}^{n} \mu_{w_k,doc_j}}{n} \tag{2.3}$$

$<w_i, w_j>$	سيارة	عربة	نقل	طائرة	احدى
سيارة	1	1	0.7	0.85	0.001
عربة	-	1	0.5	0.6	0.002
نقل	-	-	1	0.7	0.003
طائرة	-	-	-	1	0.002
احدى	-	-	-	-	1
...	-	-	-	-	-

TABLE 2.1 – Matrice de corrélation

— Enfin, le taux de similarité $\text{Sim}(\text{doc}_i\,;\,\text{doc}_j)$ est comparé à une valeur de seuil fixe *SP* pour juger si doc_i et doc_j sont similaires ou non. Le seuil SP est calculé en utilisant la technique proposée par Yerra et Ng [RYK05].

2.2.3 SVD & LSA

En algèbre linéaire, la décomposition en valeurs singulières (SVD) d'une matrice est une factorisation de cette matrice en trois matrices [1].
L'analyse sémantique latente (*LSA*, de l'anglais : *Latent semantic analysis*) est un procédé de traitement des langues naturelles. Elle permet d'établir des relations entre un ensemble de documents et les termes qu'ils contiennent, en construisant des *concepts* liés aux documents et aux termes.

PDS-AD

Hussein [Hus15] propose un nouveau système de détection du plagiat pour les documents arabes nommé PDS-AD (de l'anglais A Plagiarism Detection System for Arabic Documents) basé sur la modélisation de la relation entre les textes et leurs n-grammes de phrases. Le système PDS-AD passe par une phase de pré-traitement qui comprend plusieurs étapes, y compris : la suppression de mots vides, l'étiquetage morphosyntaxique et la substitution par des synonymes. Ensuite, il introduit un algorithme heuristique de correspondance des paires de phrases afin de construire pour chaque document un modèle **tf-idf**(de l'anglais : *term frequency — inverse document frequency*) sous forme d'une matrice [SB88].
PDS-AD estime le taux de plagiat entre deux modèles **tf-idfs** par analyse sémantique latente (LSA, de l'anglais Latent Semantic Analysis) [SE06] et par décomposition en valeurs singulières (SVD, de l'anglais Singular Value Decomposition) [Ces08]. Le PDS-AD atteint un taux de rappel de 88% dans les cas de plagiat avec restructuration des phrases et de 86% pour la substitution par des synonymes respectivement.

2.2.4 Word Embedding

Le word embedding (*Plongement de mots* [VVC15] ou *plongement lexical* [Bra15] en français) est une méthode d'apprentissage d'une représentation de mots utilisée notamment en traitement automatique des langues (*TAL*). Elle permet de représenter les mots par des vecteurs dans un espace multidimensionnel, en prenant compte le contexte des mots dans le texte.

1. *wikipedia*: Définition de la décomposition en valeurs singulières (consulté le 02/8/2020)

2L-APD

Le système 2L-APD [NCS18] combine beaucoup de concepts et d'outils : la représentation avec *word embedding* , l'alignement des mots, le sac de sens, la pondération de la fréquence des termes et le balisage de la partie de la parole afin de créer un vecteur de représentation pour chaque phrase du texte sentences. Ensuite, une comparaison approfondie est effectuée entre les phrases suspectes S_{sus} et les phrases sources S_{src}.

La similarité entre S_{sus} et S_{src} est déterminée par les étapes suivantes :

— Tout d'abord, la 2L-APD a appliqué la segmentation par phrases, chaque document d_{src} en D et d_{sus} respectivement est divisé en phrases S_{sus} et S_{src} respectivement et normalise également ces phrases.

— Puis utiliser la technique d'alignement des mots proposée par Nagoudi et al[NCS18]. pour aligner les mots en fonction de leur similarité sémantique dans le modèle CBOW arabe de Zahran et al[MAZA15].

— Créer un *Bag-of-Meaning*(BoM_w) pour chaque mot donné w en utilisant le modèle arabe CBOW de Zahran et al[MAZA15].

— Calcule *Bag-of-Meaning* Similarité en calculant la similarité Jaccard entre chaque BoM_w et $BoM_{w'}$ (w, mots qui sont utilisés dans les mêmes contextes de w) comme suit :

$$Jacc(BoM_w; BoM_{w'}) = \frac{BoM_w \cap BoM_{w'}}{BoM_w \cup BoM_{w'}} \tag{2.4}$$

— Construire les vecteurs VR_{src}(Représentation vectorielle$_{src}$) et VR_{sus} pour S_{src} et S_{sus} respectivement, la valeur de l'entrée VR_{src} et VR_{sus} est fixée à Jacc(BoM_w ;$BoM_{w'}$).

— Nagoudi et Schwab [NS17] proposée une fonction de pondération *WT*, dans cette étape ils supposent que tous les mots n'ont pas la même importance pour le sens. Et créer *Vector Weighting* en utiliser la formule :

$$WT[w] = IDF(w) * POS(w) \tag{2.5}$$

— Calculer la similarité entre SS_{src} et S_{sus} par :

$$Sim(S_{src}; S_{sus}) = \frac{1}{2}\left(\frac{\sum_{w \in S_{src}} WT(w) * VR[w]}{\sum_{w \in S_{src}} WT(w)} + \frac{\sum_{w \in S_{sus}} WT(w) * VR[w']}{\sum_{w \in S_{sus}} WT(w)}\right) \tag{2.6}$$

2.3 Conclusion

Dans ce chapitre, nous définissons brièvement les deux types de techniques de détection du plagiat *Intrinsèque* et *Extrinsèque*. Nous avons présenté quelques techniques *Extrinsèque* utilisées pour la détection du plagiat textuel et nous prenons comme exemple les systèmes proposés pour chacun d'entre eux. Parmi ces systèmes, certains ne sont pas encore mis en œuvre d'un façon professionnelles pour facilite leur usage.
Dans le chapitre suivant principalement nous allons mettre en œuvre la méthode 2L-APD (de l'anglais *A Two-Level Arabic Plagiarism Detection System*). Un système de détection du plagiat à deux niveaux proposé par Nagoudi et al [NCS18].

Chapitre 3

Contribution

Dans ce chapitre, nous décrivons notre contribution au projet de boîte à outils pour la détection du plagiat. Cette contribution est divisée en deux parties.
Dans la première partie, nous présentons notre participation à la conception de la boîte à outils. Dans la deuxième partie, nous exposons notre implémentation de la méthode *2L-APD*. Nous commençons par décrire le processus de fonctionnement de la méthode *2L-APD* et ensuite nous présentons une brève description de la boîte à outils à laquelle nous avons participé à sa mise en place avec Chennouf [Che20]. Dans la dernière section, nous présentons l'ensemble des langages et des outils utilisés ainsi que nos différents script liées à 2L-APD.

3.1 Description de la méthode *2L-APD*

2L-APD (Two-Level Arabic Plagiarism Detection System) [NCS18] offre une détection du plagiat extrinsèque à deux niveaux : lexical et sémantique. L'idée de base est basée sur la décomposition du problème de détection du plagiat en deux sous-problèmes : la détection du plagiat simple appelé plagiat littéral, puis dans une deuxième partie traitant du plagiat plus complexe.

La détection du plagiat simple (littéral) consiste à utiliser les empreintes digitales(module de détection basé sur les empreintes digitales) pour détecter le plagiat littéral, le copier/-coller, la restructuration des phrases, l'ajout de mots vides. Puis il utilisé un module de détection basé sur des word embeddings pour traiter les plagiats plus complexes (niveaux sémantiques) tels que la substitution par des synonymes et la reformulation de phrases.

Le problème de détection de plagiat prend en entrée une collection de documents $D = d_1, d_2, ..., d_n$ et un document suspect d_{sus}. Dans un premier temps, le passage suspect p_{sus} est scanné par le premier module ED_{mod}(module d'empreintes digitales), si aucun plagiat n'est détecté, alors p_{sus} est envoyé au deuxième module WE_{mod}(module de word embeddings) pour détecter le plagiat dissimulé s'il existe. La figure 3.1 illustre une vue d'ensemble sur le système 2L-APD.

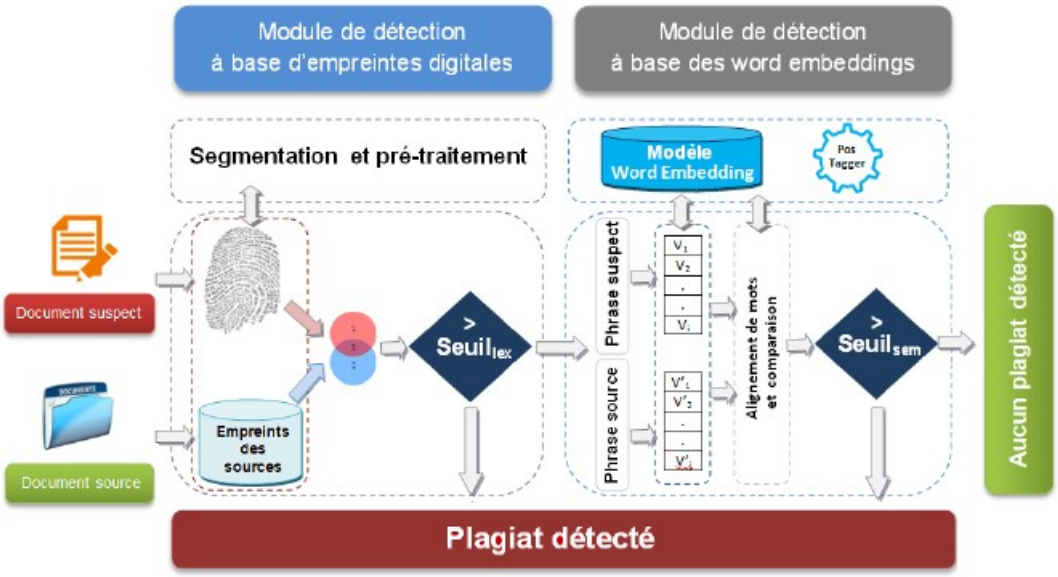

FIGURE 3.1 – Architecture du système 2L-APD [NCS18]

Avant de passer à la description des deux modèles ED_{mod} et WE_{mod}, il est nécessaire de présenter la phase de segmentation et de pré-traitement appliquée sur tous les documents à analyser par le système 2L-APD.

3.1.1 Segmentation et pré-traitement

Le système 2L-APD segmente les documents d_{sus} et d_{src} en phrases, et du fait que dans les documents arabes la phrase contient environ 35 mots par phrase [MLP06], Nagoudi et al [NCS18] ont choisi d'utiliser les signes de ponctuation (.), (,), (؛), (;), (:), (!) et (؟) comme points de segmentation, à condition que la longueur de la phrase soit inférieure à 35 mots. Ensuite, un ensemble d'étapes de pré-traitement sont appliquées afin de normaliser les phrases pour les modules de détection :

- Tokenisation : décomposer chaque phrase en un ensemble de mots.
- Suppression de mots vides.
- Suppression des signes de ponctuation, diacritiques et les caractères nonalphanumériques.
- Lemmatisation : l'outil *MADAMIRA* [PABD+14] est utilisé uniquement pour le module ED_{mod} afin de réduire les mots à leurs lemmes. Le module WD_{mod} utilise la forme normale des mots afin de préserver leurs propriétés sémantiques.

3.1.2 Module de détection avec empreintes digitales

La détection de plagiat entre les phrases de documents suspects d_{sus} et de source d_{src} dans ce module de détection est basée sur les empreintes digitales. Elle est effectuée comme suit :

1. ***Création d'empreintes digitales***, ce processus passe par les étapes suivantes :

— **Chunking**,Chaque phrase est représentée par un ensemble de n-grammes de caractères (chunks).

— **Sélection**, Dans le but de sélectionner un sous-ensemble de chunks, Nagoudi et al [NKC16] proposent de sélectionner uniquement les n-grammes (chunks) ayant une fréquence inférieure à un seuil de sélection T_{selec} donné.

2. ***Hachage***, La fonction de hachage de Brian Kernighan et Dennis Ritchie [ATR06] est appliquée aux n-grammes sélectionnés pour générer une empreinte digitale unique pour chaque phrase.

La figure 3.2 montre un exemple de détection du plagiat entre deux textes arabes avec le module ED_{mod}.

Texte source		**Texte suspect**	
ذَهَبَ يُوسفُ إلى الكُلية لِلحُضُور إلى النَدْوة العِلمية		يمضي يوسف مسرعا للجامعة من أجل الحضور للدورة العلمية	
Segmentation			
S_2: لِلحُضُور إلى النَدْوة العِلمية	S_1: ذَهَبَ يُوسفُ إلى الكُلية ،	S'_2: من أجل الحضورللدورة العلمية	S'_1: يمضي يوسف مسرعا للجامعة
Prétraitement: Tokenisation, Supprimer les signes diacritiques, les non-lettres et les mots-vides			
للحضور الندوة العلمية	ذهب يوسف الكلية	الحضور لدورة العلمية	يمضي يوسف مسرعا للجامعة
Lemmatisation			
حضور ندوة علمي	ذهب يوسف كلية	حضور دورة علمي	مضى يوسف مسرع جامعة
Extraction des N-grammes			
حضوضور ورن رند ندو دوة ةعل علم لمي	ذهب هبي بيو يوس وسف سفك كلي لية	حضوضور ورد ردو دور ورة ةعل علم لمي	سرع مضى بيو يوس وسف سفم مسر رعج عجا جام امع معة
Sélection des N-grammes			
ضور دوة ةعل لمي	هبي يوس سفك كلي	ضور ردو ورة ةعل لمي	يوس سفم رعج جام معة
Hachage avec la fonction BKDR			
258 101 211 189 : $f(S_2)$	160 236 203 209 : $f(S_1)$	258 101 136 293189: $f(S'_2)$	256 87 741 781 166 : $f(S'_1)$
Jaccard (S'_j, S'_j) > S_{lex}= 15% ?			
Jaccard $(f(S_1), f(S'_1)) < T_{lex}$	Envoyer les phrases au module WD_{mod} (niveau sémantique)		
Jaccard $(f(S_1), f(S'_2)) < T_{lex}$ Jaccard $(f(S_2), f(S'_1)) < T_{lex}$	Envoyer les phrases au module WD_{mod} Envoyer les phrases au module WD_{mod}		
Jaccard $(f(S_2), f(S'_2)) > T_{lex}$	**Un plagiat potentiel est détecté entre S_2 et S'_2**		

FIGURE 3.2 – Illustration de détection de plagiat par le module ED_{mod} [Nag17]

3.1.3 Module de détection avec word embeddings

Dans le module Word Embeddings (WD_{mod}) comme nous nous l'avons mentionné dans la section 2.2.4, 2L-APD utilise différentes techniques dont la première est la représentation de *word embedding* afin de créer un vecteur de représentation pour chaque phrase du texte. Ensuite, une comparaison approfondie est effectuée entre les phrases suspectes S_{sus} et les phrases sources S_{src} a travers cette représentation..

Par exemple, soit les deux phrases :
S_{src} ="ذهب يوسف إلى الكلية"(Joseph est allé à la faculté)
et S_{sus} ="يوسف يمضى مسرعا إلى للجامعة"(Joseph va rapidement à l'université).

La similarité entre S_{src} et S_{sus} est obtenue comme suit :

— **Calculer les vecteurs de phrases.** Pour calculer les vecteurs représentant les phrases V_{src} et V_{sus} nous effectuons la somme des vecteurs de leurs mots :

V_{src} = v(الكلية) + v(يوسف) + v(ذهب) (le mot vide إلى est supprimé)

V_{sus} = v(للجامعة) + v(مسرعا) + v(يمضى) + v(يوسف)

— **Mesure de similarité.** La similarité entre S_{src} et S_{sus} est obtenue en calculant la similarité cosinus entre V_{src} et V_{sus} : $Sim(V_{src}, V_{src}) = Cos(V_{src}, V_{src})$.

3.2 Description de la boîte à outils

Nous avons participé à l'élaboration et à la conception avec chennouf [Che20] d'une boîte à outils dédiée aux systèmes de détection du plagiat pour la langue arabe. Cette plateforme est baptisée راقب(en anglais "Rakib"). Ou l'interface (front-end) est développé en utilisant *Electron* et l'infrastructure (back-end) en *Python/Node.js*. Nous présentons la conception de cette boite à outils pour montrer où va s'intégrer notre développement de la 2L-APD.

3.2.1 Diagramme de classes

Un diagramme de classes représente la structure statique d'un système. Il contient principalement des classes, ainsi que leurs associations et on peut aussi y trouver des objets. Il représente des héritages, agrégations et compositions, le diagramme de classes peut être utilisé pour représenter les trois éléments clés d'un système d'informations (acteurs, informations, processus).
Le diagramme de classes comporte 6 concepts :

— **classe**.

— **attribut**.

— **identifiant**.

— **relation**.

— **opération**.

— **généralisation/spécialisation**.

Notion de classe

Une classe est une description abstraite (condensée) d'un ensemble d'objets du domaine de l'application : elle définit leur structure, leur comportement et leurs relations.

Notion de relation

S'il existe des liens entre objets, cela se traduit nécessairement par des relations qui existent entre leurs classes respectives. Les liens entre les objets doivent être considérés comme des instances de relations entre classes.
Il existe plusieurs types de relations entre classes :

— L'association.

— La généralisation/spécialisation

Nous contentons ici de présenter juste l'essentiel de la conception à savoir le diagramme de classe figure 3.3 et celui de séquence relative à l'ajout d'une méthode x figure 3.4.

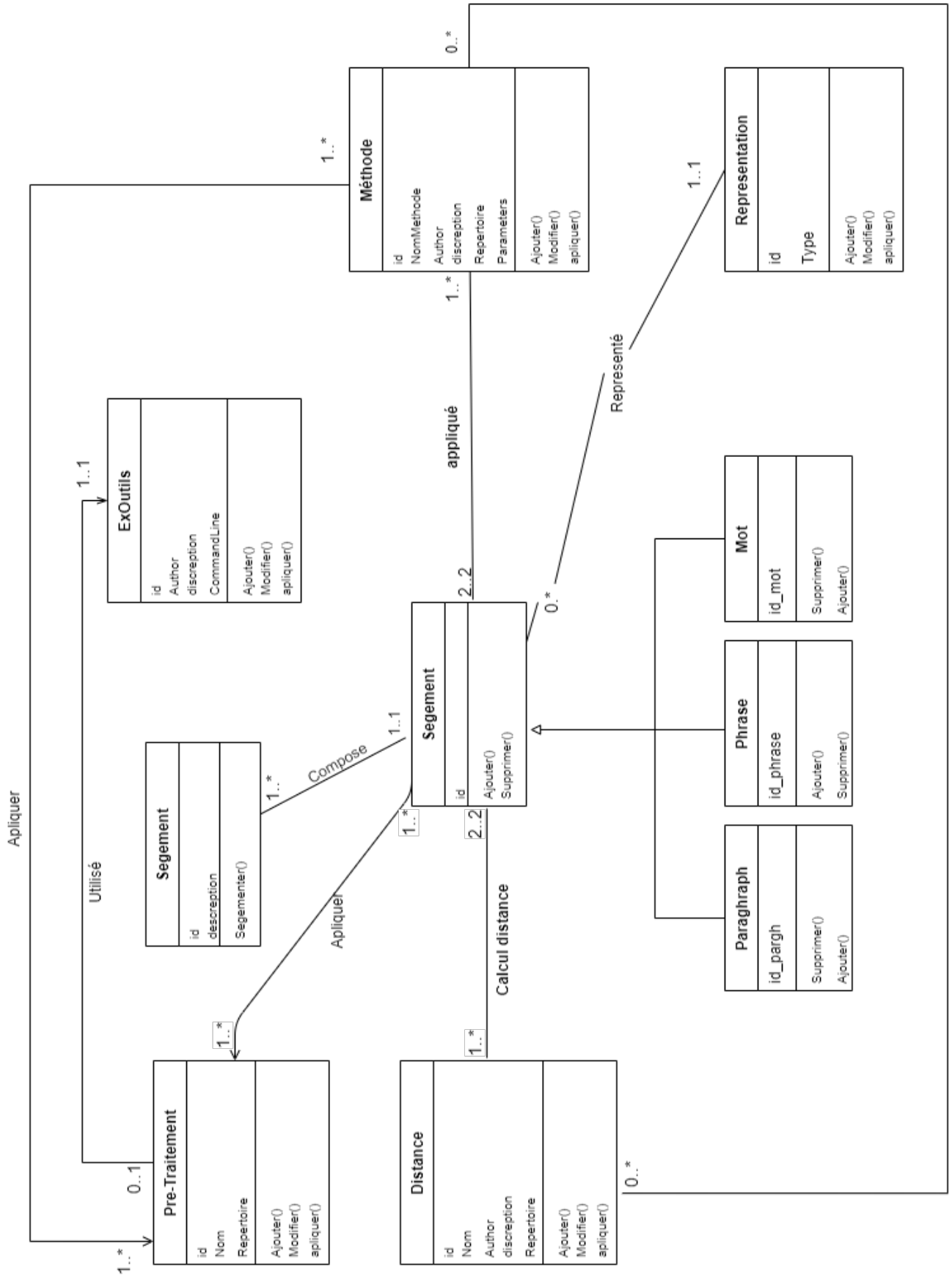

FIGURE 3.3 – Diagramme de classe de la boîte à outils

Essentiellement, notre diagramme (3.3) de classe repose sur le fait qu'un segment n'est qu'une partie d'un document et qu'il peut être soit une phrase, un paragraphe, un mot, sur lequel peut s'appliquer un *pré-traitement*, calcule un *distance*, applique une méthode, et peut représenter un segment soit avec un *word embedding* représentation, soit avec un *empreinte digitale*, etc.

3.2.2 Diagramme de séquence

Le diagramme de séquence décrit les interactions entre un groupe d'objets en montrant, de façon séquentielle, les envois de message qui interviennent entre les objets.

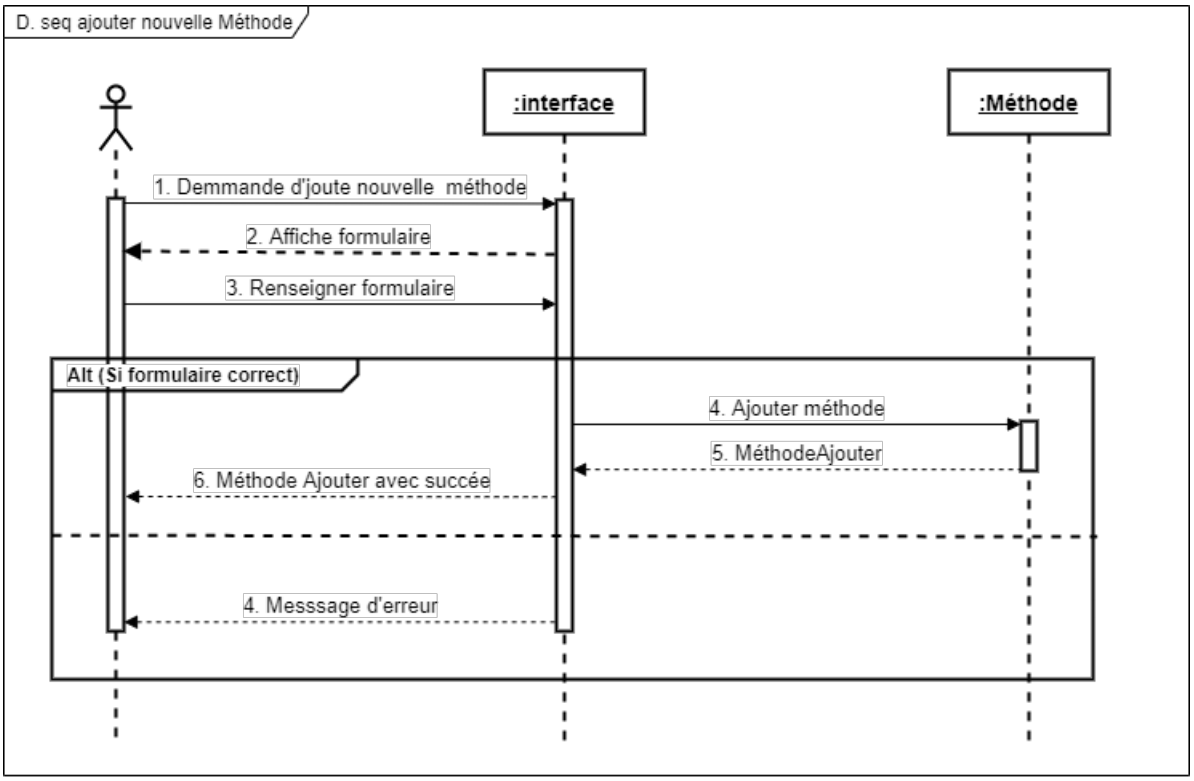

FIGURE 3.4 – Diagramme de séquence d'ajout d'une méthode

Ce diagramme (3.4), illustre que l'ajout d'une méthode à la boite à outils est réalisé suite à une demande de l utilisateur. En effet, il doit renseigner le système via un formulaire des informations sur cette méthode à savoir sa description, ses auteurs , le répertoire où se trouve le script de cette méthode ainsi que l'ensemble des pré-traitement qu'elle exige.

Les fonctionnalités de la boîte à outils

La figure 3.5 illustre l'architecture de la boîte à outils. Elle est conçue pour aider les chercheurs à tester et à comparer leur méthode de détection du plagiat avec d'autres méthodes. La principale fonction de cette boîte à outils est :

— **Analyser** les textes arabes (Tokenization , stop-words removal ..) voir l'annexe 4.1.

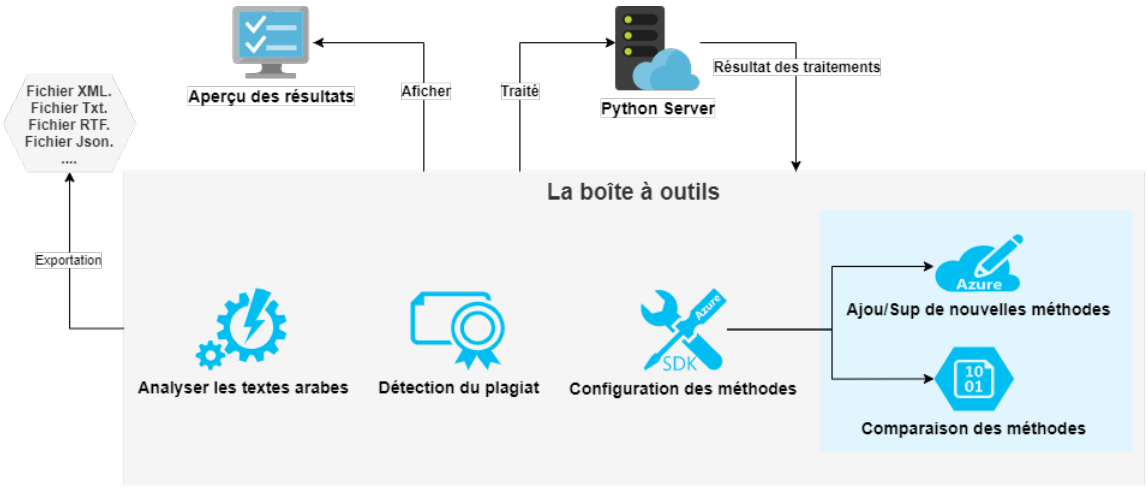

FIGURE 3.5 – Fonctionnalités de la boîte à outils

— ***Plagiat check***, L'interface de détection du plagiat donne à l'utilisateur la possibilité de copier/coller ou de charger des documents et de choisir la méthode de détection pour le traitement (voir l'annexe 4.2).

— ***Méthodes configuration***, L'interface de configuration des méthodes de détection du plagiat est divisée en deux parties(voir l'annexe 4.3) :

i) ***New Method***, L'interface permettant d'ajouter de nouvelles méthodes de détection en remplissant un formulaire.

ii) ***Comparaison of methods***, L'interface de comparaison des méthodes, il suffit de choisir deux méthodes déjà ajoutées dans la boîte à outils et d'effectuer la comparaison.

3.3 Implémentation de la méthode *2L-APD*

En ce qui concerne l'environnement de l'implémentation nous avons eu recours à certains logiciels qui permettent d'accomplir l'implémentation de la méthode *2L-APD*.

3.3.1 Environnement logiciel

Logiciels et bibliothèques déployés

Dans le cadre de l'implémentation, nous avons utilisé certains langages, bibliothèques et logiciels qui nous permettent de mettre en œuvre la méthode ainsi que les différentes interfaces de la boite à outils.

Python

Python est un langage de programmation interprété et multiplateforme. Il prend en charge la programmation structurée, fonctionnelle et orientée objet [1].
Nous avons choisi ce langage pour sa richesse en bibliothèques et sa puissance dans le TALN.

1. Définition de Python en *wikipedia*.

Visual Studio Code

VScode est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS, open source et gratuit, supportant une dizaine de langages. Il est basé sur **Electron**[2].
Nous avons choisi cet éditeur en raison de sa puissante fonctionnalité : de débogage, de mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la refactorisation du code et Git intégré, les raccourcis clavier, les préférences et la possibilité d'installer des extensions qui ajoutent des fonctionnalités supplémentaires(*live server*, ..).

Pyarabic

Pyarabic[3], une bibliothèque Python spécifique pour la langue arabe développé par *Taha Zerrouki* [Tah10b], fournit des fonctions de base pour manipuler les lettres et le texte arabes, comme la détection des lettres arabes, des groupes de lettres arabes et de leurs caractéristiques, la suppression des diacritiques, etc.

Naftawayh

Naftawayh[4] est une bibliothèque Python pour le marquage des mots arabes (classification des mots) en types (noms, verbes, mots vides) développé par *Taha Zerrouki* [Tah10a], ce qui est utile dans le traitement du langage, notamment pour l'exploration de textes. Naftawayh fonctionne selon la structure des mots arabes, et la capacité de deviner la classe de mots, à travers certains signes.

Tashaphyne

Tashaphyne[5] est un stemmer et segmenteur arabe léger développé par *Taha Zerrouki* [Tah10c]. Il supporte principalement le light stemmming (suppression des préfixes et suffixes) et donne toutes les segmentations possibles. Il utilise un AEF modifié qui permet de générer toutes les segmentations.

NLTK

NLTK est une plate-forme permettant de construire des programmes Python pour travailler avec des données en langage humain[6]. Elle fournit des bibliothèques pour la classification, la tokenisation, le marquage, l'étiquetage, l'analyse et le raisonnement sémantique.

Adawat

Adawat[7] est un ensemble d'outils pour la langue arabe développé par *Taha Zerrouki* [Zer20], parmi ces outils *Tashkeel, arabize, romanize, etc.* Est écrire en langage Python.

2. VS Code description en *wikipedia*.
3. PyArabic: Python Library for Arabic's documentation.
4. Naftawayh: Arabic Word Tagger Site web.
5. Tashaphyne : Arabic light stemmer Site web.
6. Documentation de NLTK
7. Adawat: Arabic Language Toolkit

3.3.2 Implémentation de *2L-APD*

Afin d'implémenter la *2L-APD*, nous avons crée deux scripts python : *traitement.py*(voir Figure 3.6) et *2l_ apd.py* (voir Figure 3.7).
Le script *traitement.py* contient toutes les définitions de fonctions appelées par le script *2l_ apd.py*(dans la classe *Process*), donc dans cette classe on trouve toutes les fonctions Segmentation, Pré-traitement, Lemmatisation etc.

```
from pyarabic.araby import strip_harakat                # remove harakat
from pyarabic.araby import strip_tashkeel               # clean text
from pyarabic.araby import strip_tatweel, strip_shadda  # tatweel and remove shadda
from pyarabic.araby import normalize_ligature           # correction
from pyarabic import araby                              # tafri9 text words
import naftawayh.wordtag
from naftawayh.wordtag import WordTagger
from collections.abc import Iterable

import pyarabic.arabrepr
arepr = pyarabic.arabrepr.ArabicRepr()
repr = arepr.repr
from tashaphyne.stemming import ArabicLightStemmer


import sys
import re
import io
import codecs
import json
sys.path.append('Clean_and_Segmentation')


class Process: ···
```

FIGURE 3.6 – Code source *traitement.py*

Dans l'autre partie, le script *2l_ apd.py* contient une instance de la classe méthode *2l_ apd_ level1* qui appelle des fonctions de segmentation et de pré-traitement, etc. Afin de pré-traiter les segments suspects Sg_{sus} et les segments sources Sg_{src} . Enfin, calculer la sémilarité entre les deux (Sg_{sus}, Sg_{src}) avec la fonction *_ 2lapd_ level1*.

```
class _2l_apd:
    def __init__(self, tsrs, tsp): ···

    def Segmentation(self): ···

    def Pretraitement(self): ···

    def Lemmatisation(self): ···

    def convert(self, r): ···

    def trigram_srs(self): ···

    def _2lapd_level1(self): ···

    def _2lapd_level2(self): ···
    .
```

FIGURE 3.7 – Code source *2l_ apd.py*

Segmentation et pré-traitement

La partie de la segmentation est très importante dans la détection du plagiat, dans les documents arabes cette partie est très difficile à cause des caractéristique de la langue arabe.
En utilisant les bibliographies Python (3.3.1), le script *traitement.py* segmente ;par appel à la méthode *Segmentation* de la classe document ; le texte d'entrée en phrases (en utilisant la fonction *break_into_sentences()*)(voir figure 3.8), et le nettoyez en utilisant *removeHarakat(), removetashkeel(), removeTatweel()* de la méthode *Pré-traitement*.

```
def Segmentation(self):
    return self.break_into_sentences()

def Pretraitement(self):
    t1 = self.removeHarakat(self.text)
    t1 = self.removetashkeel(t1)
    t1 = self.removeTatweel(t1)
    print(t1)
    tagger = naftawayh.wordtag.WordTagger();
    sentences = self.Segmentation()
    words = []

    for sentence in sentences:
        words.append(araby.tokenize(sentence))

    print("words -- : " ,words)
    words_clear = []
    word = self.single_list(words)
    for w in word:
        if w != '،':
            if not tagger.is_stopword(w):
                words_clear.append(w)

    return words_clear;
```

FIGURE 3.8 – Segmentation et pré-traitement

3.3.3 Module de détection avec empreintes digitales

Apres cette segmentation et Pré-traitement, ce module extrait les N-grammes pour chaque phrase en utilisant la fonction *trigram()*(voir l'annexe 4.6), ces N-grammes seront sélectionnés et mis en cache avec la fonction BKDR [ATR06] et ensuite le calcul de la distance *Jaccard*[8] entre chaque deux phrases in (d_{sus}, d_{src}) : Jaccard((S_{sus1},S_{src1}) , Jaccard((S_{sus1},S_{src2}) etc.

3.3.4 Module de détection avec word embedding

Concernant ce module de la méthode 2l-APD, le travail est en cours et n'a pas été achevé à cause du temps pris pour la préparation des données pour extraire le modèle word embedding nécessaire.

8. Indice et distance de Jaccard

3.4 Conclusion

Dans ce chapitre, nous avons décrit notre contribution au projet d'implémentation d'une boîte à outils dédiée aux systèmes de détection de plagiat et montré où va s'intégrer notre développement de la 2L-APD et nous avons présenté les étapes de l'implémentation de la méthode 2L-APD.

Conclusion et Perspectives

Dans ce travail, nous avons identifié la conclusion de définitions relatives au plagiat textuel ainsi que certains termes et concepts associés. En outre, nous avons signalé certaines de ses formes importantes et certains de ses systèmes les plus connus. Nous avons mené de nombreuses recherches sur le plagiat, en le définissant et en le classant chacun à sa façon.

Nous avons défini ensuite brièvement les deux types de techniques de détection du plagiat : *Intrinsèque* et *Extrinsèque*. Nous avons présenté quelques-unes des techniques *Extrinsèque* utilisées pour la détection du plagiat textuel et nous prenons comme exemple les systèmes proposés pour chacune d'entre elles. Enfin, nous avons décrit notre contribution au projet de mise en œuvre d'une boîte à outils dédiée aux systèmes de détection du plagiat et avons montré où notre développement de la 2L-APD s'inscrira et nous avons présenté les étapes de la mise en œuvre de la 2L-APD.

Notre façon d'implémenter est basée sur des paradigmes orientés objets modularité, ce qui rend la maintenance très facile, tout comme le langage Python rend notre implémentation 2L-APD plus intégrable et plus facile à maintenir. Dans l'autre partie, la boîte à outils également basée sur l'orientés objets , ce qui le rend personnalisable et plus dynamique.

Notre travail est une étape pionnière dans l'aboutissement d'un plus grand projet, qui ouvre le champ à de nombreuses perspectives, dont certaines des plus intéressantes que nous proposons :

— Nous envisageons d'améliorer notre implémentation de module de détection par *fingerprinting*.

— Nous projetons de finaliser l'implémentation du second module de détection par *word embedding*.

— En ce qui concerne la boite à outils, nous prévoyant continuer le développement de la base de données en intégrant un corpus arabe. Pour l'interface, on pourra envisager la personnalisation des activités.

Bibliographie

[AS08] Salha Mohammed Alzahrani and Naomie Salim. Plagiarism detection in arabic scripts using fuzzy information retrieval. page 281–285, 2008.

[ATR06] Ritchie Anna, Simone Teufel, and Stephen Robertson. How to find better index terms through citations. In *Proceedings of the workshop on how can computational linguistics improve information retrieval ?*, pages 25–32, 2006.

[BER+06] William Black, Sabri Elkateb, Horacio Rodriguez, Musa Alkhalifa, Piek Vossen, Adam Pease, and Christiane Fellbaum. Introducing the arabic wordnet project. page 295–300, 2006.

[Bra15] Chloé Braud. *Identification automatique des relations discursives implicites à partir de corpus annotés et de données brutes*. Theses, Universite Paris Diderot-Paris VII, December 2015.

[CB18] Hussain A. Chowdhury and Dhruba K. Bhattacharyya. Plagiarism : Taxonomy, Tools and Detection Techniques. *arXiv :1801.06323 [cs]*, January 2018. arXiv : 1801.06323.

[CDM08] Hinrich Schütze et al Christopher D Manning, Prabhakar Raghavan. Introduction to information retrieval. 2008.

[Ces08] Zdenek Ceska. Plagiarism detection based on singular value decomposition. page 108–119, 2008.

[Che20] Mohammed Chennouf. Réalisation d'une boite à outils dédiée aux systèmes de détection de plagiat pour la langue arabe. 2020.

[Clo03] Paul Clough. *Old and new challenges in automatic plagiarism detection. National UK Plagiarism Advisory Service*. 2003.

[Deb10] Weber-Wulff Debora. Test cases for plagiarism detection software. *In Proceedings of the 4th International Plagiarism Conference*, 2010.

[EBMB11] Mohamed El Bachir Menai and Manar Bagais. APlag : A plagiarism checker for Arabic texts. In *2011 6th International Conference on Computer Science Education (ICCSE)*, pages 1379–1383, August 2011.

[Fis09] Teddi Fishman. We know it when we see it" is not good enough : toward a standard definition of plagiarism that transcends theft, fraud, and copyright. 2009.

[FNBF17] Patricia I Fusch, Lawrence R Ness, Janet M Booker, and Gene E Fusch. The ethical implications of plagiarism and ghostwriting in an open society. *Journal of Social Change*, 9(1) :4, 2017.

[Fre18] Baudry Frederic. *Grammaire comparée des langues classiques, contenant la théorie élémentaire de la formation des mots en sanscrit, en grec et en latin avec références aux langues germaniques*. L. Hachette et Cie, 2018.

[Hus15] Ashraf S Hussein. A plagiarism detection system for arabic documents. page 541–552, 2015.

[JE12] Ameera Jadalla and Ashraf Elnagar. A plagiarism detection system for arabic text-based documents. page 145–153, 2012.

[JPBL03] Xiao-Domg Liu Jun-Peng Bao, Jun-Yi Shen and Hai-Yan Liu. Quick asymmetric text similarity measures. in machine learning and cybernetics. page 374–379, 2003.

[KR87] Richard M Karp and Michael O Rabin. Efficient randomized pattern matching algorithms. page 249–260, 1987.

[KSM15] Imtiaz Hussain Khan, Muazzam Ahmed Siddiqui, and Kamal Mansoor. A framework for plagiarism detection in arabic documents. In *Proceedings of the Conference on Computer Science & Information Technology*, pages 1–9, 2015.

[MAZA15] A. Y. Mahgoub H. Raafat M. Rashwan M. A. Zahran, A. Magooda and A. Atyia. On the use of fuzzy information retrieval for gauging similarity of arabic documents. In *Word representations in vector space and their applications for arabic*, page 430–443, 2015.

[Men12] Mohamed El Bachir Menai. Detection of plagiarism in arabic documents. page 80, 2012.

[MLP06] Ryan McDonald, Kevin Lerman, and Fernando Pereira. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 216–220, New York City, June 2006. Association for Computational Linguistics.

[MPB10] Kashkur Maria, Serge Parshutin, and Arkady Borisov. Research into plagiarism cases and plagiarism detection methods. *Applied Computer Systems*, 42(1) :138–143, 2010.

[Nag17] El Moatez Billah Nagoudi. Détection automatique de plagiat. 2017.

[NCS18] El Moatez Billah Nagoudi, Hadda Cherroun, and Didier Schwab. A Two-Level Plagiarism Detection System for Arabic Documents. *Cybernetics and Information Technologies*, 18(1), February 2018.

[NKC16] El Moatez Billah Nagoudi, Ahmed Khorsi, and Hadda Cherroun. Efficient inverted index with n-gram sampling for string matching in arabic documents. In *13th IEEE/ACS International Conference of Computer Systems and Applications, AICCSA 2016, Agadir, Morocco, November 29 - December 2, 2016*, pages 1–7. IEEE Computer Society, 2016.

[NS17] El Moatez Billah Nagoudi and Didier Schwab. Semantic similarity of arabic sentences with word embeddings. 04 2017.

[Oli97] Millet Olivier. *Dictionnaire des citations*. Librairie générale française, 1997.

[PABD+14] Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. MADAMIRA : A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1094–1101, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[RYK05] Yerra Rajiv and Ng Yiu-Kai. A sentence-based copy detection approach for web documents. pages 557–570. Springer, 2005.

[SB88] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. page 513–523, 1988.

[SE06] Sabrina Simmons and Zachary Estes. Using latent semantic analysis to estimate similarity. 2006.

[SGM95] Narayanan Shivakumar and Hector Garcia-Molina. Scam : A copy detection mechanism for digital documents. 1995.

[SSRA03] Daniel S Wilkerson Saul Schleimer, Michael O Rabin, and Alex Aiken. Winnowing : local algorithms for document fingerprinting. page 76–85, 2003.

[Tah10a] Zerrouki Taha. Naftawayh : Arabic word tagger, 2010.

[Tah10b] Zerrouki Taha. pyarabic, an arabic language library for python, 2010.

[Tah10c] Zerrouki Taha. Tashaphyne : Arabic light stemmer, 2010.

[VVC15] Vukotic Vedran, Claveau Vincent, and Raymond Christian. Irisa at deft 2015 : Supervised and unsupervised methods in sentiment analysis. In *Actes de la 11e Défi Fouille de Texte*, pages 16–27, Caen, France, June 2015. Association pour le Traitement Automatique des Langues.

[Zad65] L.A. Zadeh. Fuzzy sets. *Information Control*, 8 :338–353, 1965.

[Zer20] Taha Zerrouki. Towards an open platform for arabic language processing, 2020.

Annexe

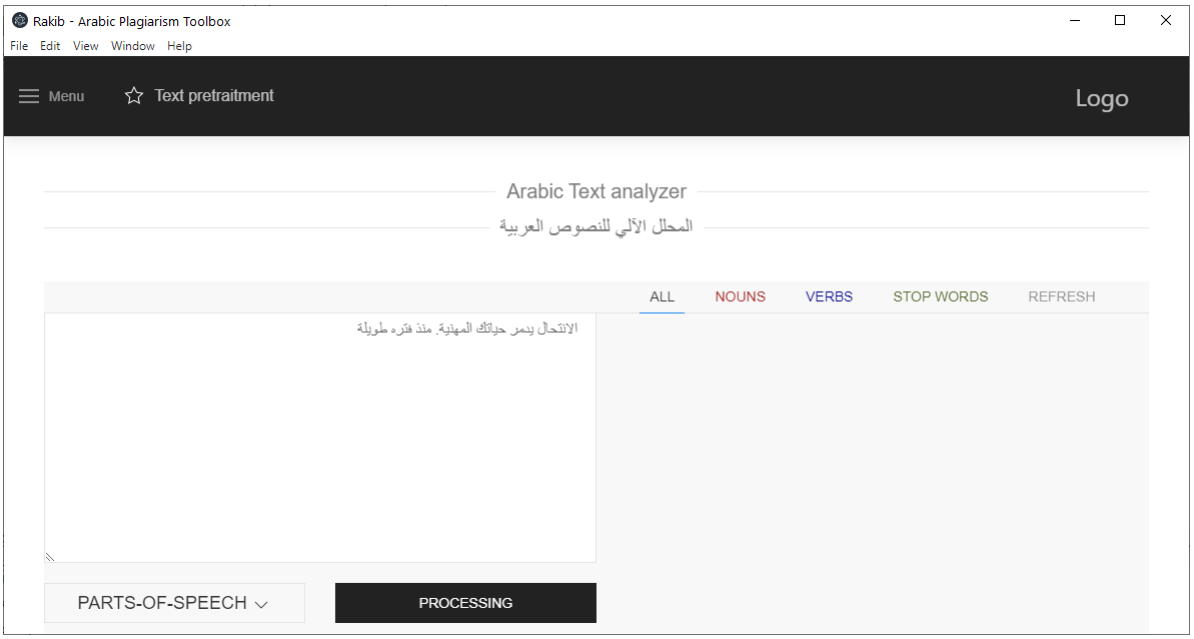

FIGURE 4.1 – Analyser les textes arabes [Che20]

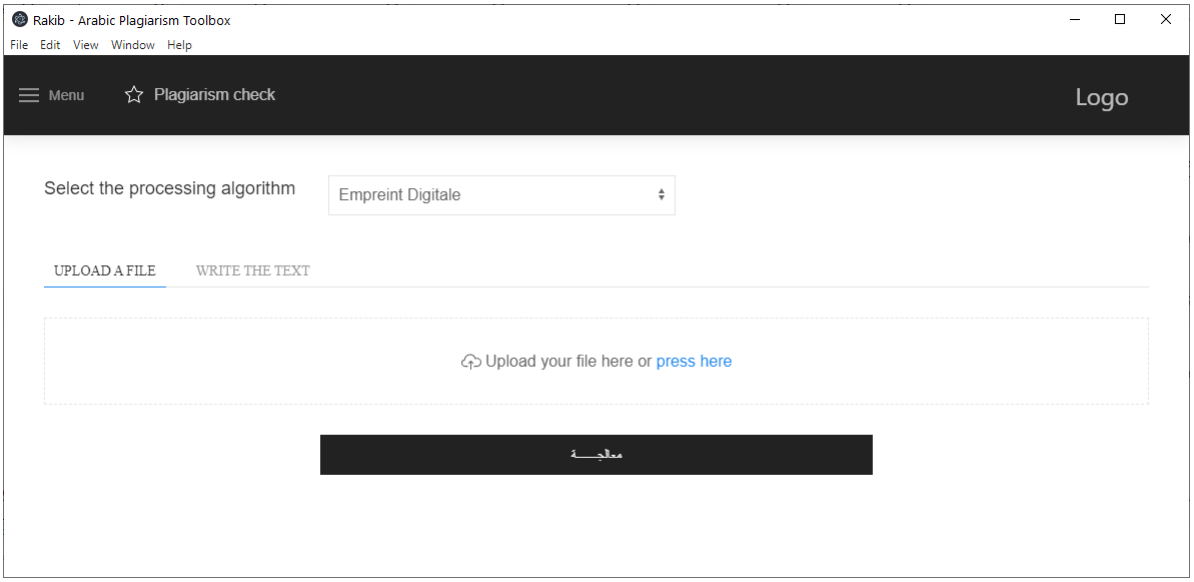

FIGURE 4.2 – L'interface de détection du plagiat [Che20]

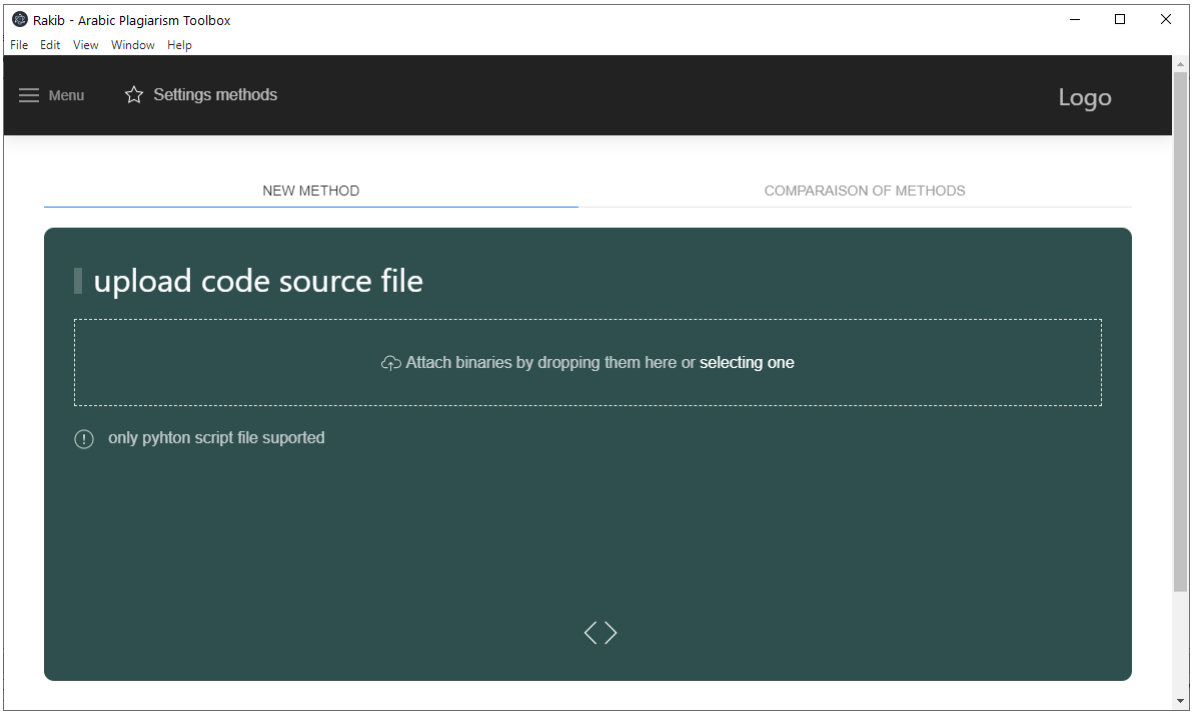

FIGURE 4.3 – L'interface de configuration des méthodes [Che20]

```
def Lemmatisation(self):
    tagger = naftawayh.wordtag.WordTagger();
    ws = self.Pretraitement()
    ArListem = ArabicLightStemmer()
    words_root = []
    words_all = {}
    words_all['words'] = []
    for w in ws:
        #if not tagger.is_noun(w):
        stem = ArListem.light_stem(w)
        ww = ArListem.get_prefix()+" " + "+ ArListem.get_stem() +
                                    " " + " + ArListem.get_suffix()
        words_all['words'].append(ww)
        words_root.append(ArListem.get_stem())
```

FIGURE 4.4 – Fonction *Lemmatisation()*

```
for ch in paragraph.strip():
    if ch in [u'؟', u'!', u'.', u':', u'؛',u'?']:
        Sentence_Size = 0
        flag = True
    elif flag:
        sentences.append(''.join(temp_sentence).strip())
        temp_sentence = []
        flag = False
    regex = re.compile(r'[ٱٱٱٱ]')
    ch = re.sub(regex, 'ا', ch)
    regex = re.compile(r'[ؤ]')
    ch = re.sub(regex, 'ؤ', ch)
    #remove_non_arabic_symbols fct
    ch = re.sub(r'[^\u0600-\u06FF]', ' ', ch)
    temp_sentence.append(ch)
    if ch.isspace():
        Sentence_Size = Sentence_Size + 1
    if Sentence_Size > Max_Size:
        Sentence_Size = 0
        flag = True
```

FIGURE 4.5 – Fonction *breakIntoSentences()*

```
    def trigram_srs(self):
        result = self.Lemmatisation()
        # return ngrams(result.get('srs'), 3)
        srs = self.convert(result.get('srs'))
        sup = self.convert(result.get('sup'))
        print("srs :",srs," - sup :",sup)
        srs_1 = ''
        for i, item in enumerate(srs, start=1):
            if i % 4 == 0 :
                srs_1 = srs_1 + ","
            else:
                srs_1 = srs_1 + item
        print('final :',srs_1)
```

FIGURE 4.6 – Fonction *trigram()*