République Algérienne Démocratique et Populaire

Ministère de L'Enseignement Supérieur et De la Recherche Scientifique

Université Abderrahmane. Mira-Bejaia-

جامعة بجاية
Tasdawit n'Bgayet
Université de Béjaïa

Faculté des lettres et des langues

Département de langue et littérature françaises

Mémoire de Master

Option : Sciences du langage

Conception d'un logiciel de traitement automatique des néologismes dérivés d'anthroponymes.

Présenté par :

M. ABDELKRIM Mohamed Réda

M^lle^ ALLOUT Kenza

Membres de jury :

Dr BENNACER Mahmoud, directeur de recherche

M^lle^ NOUICER Amina examinatrice

M. SERIDJ Fouad, Président

Pr SADI Nabil, examinateur

Année universitaire 2023/2024

République Algérienne Démocratique et Populaire

Ministère de L'Enseignement Supérieur et De la Recherche Scientifique

Université Abderrahmane. Mira-Bejaia-

Faculté des lettres et des langues

Département de langue et littérature françaises

Mémoire de Master

Option : Sciences du langage

Conception d'un logiciel de traitement automatique des néologismes dérivés d'anthroponymes.

Présenté par :

M. ABDELKRIM Mohamed Réda

M^{lle} ALLOUT Kenza

Membres de jury :

Dr BENNACER Mahmoud, directeur de recherche

M^{lle} NOUICER Amina examinatrice

M. SERIDJ Fouad, Président

Pr SADI Nabil, examinateur

Année universitaire 2023/2024

Remerciements

Tout d'abord je remercie dieu le tout puissant de m'avoir donné la force et la volonté de finaliser ce travail de recherche.

Je tiens à exprimer ma gratitude envers mes chers parents, mes sœurs et mon frère, pour leur encouragement et leur soutien constant tout au long de mon parcours académique.

Je tiens à remercier mon directeur de recherche Dr M. BENNACER pour ses conseils avisés, la qualité de son encadrement exceptionnel, sa rigueur et sa disponibilité durant l'élaboration de ce mémoire de fin d'études.

Je remercie également tous nos professeurs, pour leurs enseignements inspirants et leur précieuse guidance tout au long de notre parcours universitaire.

Enfin, je remercie mon binôme pour sa collaboration, sa patience, son dévouement et son soutien tout au long de ce projet.

Je dédie ce modeste travail à toutes les personnes qui m'ont aidé et encouragé à la réalisation de ce mémoire.

M^{lle} ALLOUT Kenza

Remerciements

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce mémoire.

Tout d'abord, je remercie mes parents pour leur soutien indéfectible et leurs encouragements constants tout au long de mon parcours académique. Leur amour et leur confiance en moi ont été des piliers essentiels de ma réussite.

Je présente également mes remerciements à mon frère Mehdi, dont l'aide inestimable a été précieuse, à Loki, qui a été une source d'inspiration inépuisable. Son courage face à la maladie m'a motivé à persévérer malgré les obstacles. Un grand merci à ma binôme pour m'avoir aidé à accomplir ce mémoire avec son soutien et son implication constante.

Enfin, je tiens à remercier chaleureusement mon encadreur pour son temps, ses conseils avisés et son dévouement. Son accompagnement tout au long de ce travail a été d'une grande richesse et m'a permis de mener ce projet à bien.

À toutes et à tous, je vous adresse mes sincères remerciements.

M. ABDELKRIM Mohamed Réda

Table des matières

Introduction générale **8**
Chapitre 01 : Cadrage théorique **15**
1. Les ressources lexicales et les ontologies 18
1.1. La désambiguïsation sémantique 19
1.2. Les relations lexicales et structure morphologique 19
1.3. La contextualisation avec des ontologies 20
1.4. Amélioration de la recherche d'information 20
1.5. La gestion des ambiguïtés morphologiques 20
2. Les domaines spécifiques et approches spécialisées dans la dérivation de mots à partir de noms propres 21
3. Autour de la lexicologie, la morphologie, dérivation et les néologismes 23
4. La lexicologie 29
4.1. La lexicographie 30
4.2. Le lexique et la notion du mot 30
5. La morphologie 31
5.1. La morphologie dérivationnelle 32
5.2. Le morphème 32
5.2.1. Les morphèmes grammaticaux 34
5.2.2. Les lexèmes 35
6. Le radicale et la base d'un mot 35
7. Les mots simples 36
8. Mots communs et noms propres 36
9. Les mots dérivés 37
9.1. La dérivation 37
9.2. Les dérivés issus des noms propres 38
10. La créativité lexicale 39
11. La néologie 40
11.1 Les néologismes 41

Chapitre 02 : Considérations méthodologiques et téchniques 44
1. Procédés et méthodes suivis 45
1.1. Initialisation et découpage des données linguistiques 46
1.2. Définition des fonctions pour l'interaction avec les données..... 47
1.3. La manipulation et l'analyse des données linguistiques 47
1.4. Le raisonnement derrière l'utilisation de bases de données simplifiées 48
2. La définition des outils utilisés pour la création..... 50
2.1. Définition des langages de programmation 50
2.1.1. Les niveaux des langages de programmation 51
2.2. Le Python, définition et raisons d'utilisation 52
2.2.1. Les fonctions dans le Python..... 53
2.2.2. Les modules dans le Python..... 53
2.3. Les paramètres de fonctions en programmation..... 55
2.4. L'appels aux fonctions dans le Python..... 56
3. Le code..... 57
3.1. Définition des deux datasets utilisés dans le programme 57
3.1.1. Le dataset morphlex.txt..... 58
3.1.2. Le dataset noms.txt..... 58
3.2. Le principe du Python..... 59
4. Les futures éventualités du programme et fonctionnement de ses potentielles utilisations..... 60
4.1. Une intelligence artificielle (IA)..... 60
4.2. Une API (Interface de programmation d'application)..... 61
5. La récupération des commentaires des réseaux sociaux 61
6. L'analyse des sentiments..... 61
7. Les outils et bibliothèques utilisable en Python pour intégrer ces fonctions 61
8. Une parenthèse sur la potentielle future intégration d’un modèle d’apprentissage automatique 62
9. Le choix des noms à inclure 65
10. L’acquisition d’une base de données de noms de famille 66
11. Data.world..... 67

Chapitre 03 : Réalisation du programme 71
1. L'explication et schématisation du code 73
1.1. Importation d'une fonction 73
1.2. La création d'un dictionnaire à partir d'un fichier 73
1.3. L'extraction du préfixe 74
1.4. L'extraction du suffixe 75
1.5. L'assignation des noms de fichiers 76
1.6. La création des dictionnaires et listes à partir des fichiers 78
1.7. Le traitement d'un mot d'entrée 79
2. Exemples démontrant l'utilisation et flexibilité du programme 83
3. Les lacunes du logiciel et les potentielles extensions 91
3.1. Les difficultés du Projet 91
3.2. Les perspectives du projet 93
Conclusion générale 94
Références bibliographiques 104

Table des illustrations

Figure 1: Schématisation d'importation d'une fonction..... 73
Figure 2: Schématisation d'extraction du préfixe..... 75
Figure 3: Schématisation d'extraction du suffixe..... 76
Figure 4: Schématisation de L'assignation des noms de fichiers..... 77
Figure 5: Schématisation de La création des dictionnaires et listes à partir des fichiers..... 79
Figure 6: Schématisation du traitement d'un mot d'entrée..... 81
Figure 7: Exemple 1 : Utilisation Simple - "macroniste"..... 83
Figure 8: Exemple 2 : Utilisation Simple - "darwinisme"..... 84
Figure 9: Exemple 3 : Utilisation avec Dérivations Flexionnelles de Nombre - "marxiste »..... 85
Figure 10: Exemple 3 : Utilisation avec Dérivations Flexionnelles de Nombre - " marxistes"..... 85
Figure 11: Exemple 4 : Utilisation avec Dérivations Flexionnelles de Genre et de Nombre - "napoléonien/napoléoniennes"..... 86
Figure 12: Exemple 5 : Exception - "Trotski/trotskiste"..... 87
Figure 13: Exemple 6 : Exception - "hugo/hugolien"..... 88
Figure 14: Exemple 7 : Exception - "Staline/stalination"..... 90
Figure 15: Exemple 8 : Intégration d'un Préfixe - "detrumpation"..... 90

Introduction générale

Le traitement automatique du langage naturel (TALN) est une discipline qui s'intéresse à l'analyse et à la compréhension du langage humain par les machines. Il est aujourd'hui de plus en plus présent dans notre vie quotidienne, que ce soit pour la traduction automatique, la reconnaissance vocale ou l'analyse de sentiments sur les réseaux sociaux.

Dans ce contexte, la question de la prise en compte des néologismes dans le TALN se pose de manière de plus en plus pressante. Les néologismes sont des mots nouveaux ou des sens nouveaux donnés à des mots existants, qui apparaissent régulièrement dans les langues vivantes.

Parmi les néologismes, les mots dérivés de nom propres, représentent un enjeu particulier pour le TALN. L'utilisation de noms propres pour former des néologismes est une pratique courante dans la langue française. Ces nouveaux termes peuvent être créés pour diverses raisons, notamment pour exprimer des concepts nouveaux ou pour ajouter des nuances sémantiques à des termes existants.

Dans le contexte du traitement automatique du langage naturel (TALN), l'étude des néologismes mots dérivés de nom propres est d'autant plus importante que ces termes peuvent être difficiles à détecter et à analyser automatiquement

Les mots dérivés de nom propre peuvent ainsi apporter une richesse sémantique importante à la langue française, mais leur création pose des défis en matière de traitement automatique du langage naturel. En effet, les méthodes de détection et d'analyse des néologismes doivent être adaptées à la spécificité des mots dérivés de nom propres, qui se basent sur l'association d'idées et de connotations liées à un nom propre.

En effet, les mots dérivés de nom propres sont souvent liés à des phénomènes culturels ou sociaux, et leur traitement automatique pose des défis spécifiques pour les outils de TALN.

Dans ce mémoire, nous nous intéresserons à la prise en compte des mots dérivés de nom propres dans le TALN, en étudiant les méthodes et les outils existants, ainsi qu'en proposant de nouvelles approches pour améliorer leur traitement.

La conception d'un logiciel dédié à la définition de mots courants dérivés de noms propres, trouve sa source dans une compréhension profonde des défis linguistiques auxquels les utilisateurs contemporains font face. Cette initiative émane d'une volonté d'apporter une contribution significative à la compréhension du langage, en particulier dans un contexte où l'évolution rapide du lexique pose des défis constants à la communication et à la compréhension.

Plus concrètement, la première motivation découle de la reconnaissance des lacunes lexicales existantes dans les ressources linguistiques actuelles. Le langage évolue de manière dynamique, générant des mots dérivés de noms propres qui ne sont pas toujours inclus dans les dictionnaires traditionnels. La création de ce logiciel vise à colmater ces lacunes en fournissant des définitions précises et actualisées pour ces termes émergents.

Par ailleurs, la complexité croissante du langage contemporain, alimentée par des influences culturelles, sociales et technologiques, constitue une motivation essentielle. Ce logiciel aspire à démystifier cette complexité en fournissant des explications claires et contextualisées pour les termes dérivés, facilitant ainsi une compréhension approfondie dans un monde linguistique en constante évolution.

Une autre motivation sous-jacente réside aussi dans la volonté de faciliter la communication interpersonnelle. En rendant accessible la signification des mots dérivés de noms propres, le logiciel vise à créer un pont linguistique, favorisant une communication fluide et précise entre les individus, indépendamment de leur familiarité avec ces termes spécifiques.

La promotion de la maîtrise linguistique constitue également une motivation centrale. Ce logiciel ambitionne de servir comme outil éducatif, permettant aux utilisateurs d'approfondir leur compréhension du langage, d'élargir leur vocabulaire, et d'encourager une utilisation plus précise et nuancée des mots dans divers contextes.

En somme, la création de ce logiciel découle d'une vision holistique visant à enrichir la compréhension linguistique dans un paysage où la diversité des termes dérivés de noms propres pose des défis. Cette entreprise aspire à transcender les frontières lexicales conventionnelles pour offrir une ressource linguistique contemporaine, éducatrice et orientée vers la facilitation de la communication interpersonnelle.

Ce programme s'inscrit dans une approche analytique, visant à offrir une vision claire du langage en constante évolution. Décortiquons les utilités pratiques d'un logiciel dédié à la compréhension et à la définition de termes émergents dérivés de noms propres, mettant en avant la fonctionnalité précise que peut apporter une telle technologie.

Le suivi des tendances linguistiques par le biais de notre logiciel constitue une analyse approfondie des changements dans l'utilisation du langage au fil du temps. Voici quelques aspects ou notre technologie serai utile :

- Dynamique des nouveaux termes : Notre logiciel est équipé pour identifier et suivre la popularité des termes émergents dérivés de noms propres. Cette fonctionnalité offre une compréhension précise de la manière dont la langue intègre de nouveaux concepts, idées, ou personnalités.
- Fréquence d'apparition : Grâce à une analyse de la fréquence d'apparition des termes dérivés, notre logiciel met en évidence ceux qui gagnent en popularité, fournissant ainsi des indications sur les sujets ou influences captivant l'attention du public.

- Évolution des significations : Le suivi des tendances linguistiques englobe également la compréhension de l'évolution des significations des termes au fil du temps. Notre logiciel offre une perspective sur la manière dont ces mots acquièrent des nuances changeantes en fonction des contextes culturels et sociaux.
- Identification de mots-clés : Conçu avec précision, notre logiciel identifie efficacement les mots-clés émergents liés à des personnalités ou événements spécifiques, offrant ainsi une vision instantanée des sujets d'actualité et des préoccupations culturelles.
- Analyse de la popularité sociale : L'intégration habile de données issues des médias sociaux et d'autres plateformes en ligne permet à notre logiciel de fournir des informations précieuses sur la popularité sociale des termes, reflétant ainsi les tendances des conversations en cours.

Dans son ensemble, le suivi des tendances linguistiques par le biais de notre logiciel offre une compréhension de l'évolution du langage. Autrement dit, mettre en évidence les nouveaux termes en ascension et les ajustements dans la communication et la compréhension langagières.

L'analyse des médias sociaux via notre logiciel représente une exploration méthodique des informations générées sur ces plateformes. Elle permet une compréhension approfondie des tendances émergentes, en se concentrant sur les caractéristiques de notre technologie, examinons quelques points clés :

- Identification des tendances émergentes : En analysant les données des médias sociaux, notre logiciel peut identifier les tendances émergentes, soulignant les sujets, les hashtags ou les expressions qui gagnent en popularité.
- Analyse des sentiments : Grâce à des algorithmes sophistiqués, notre logiciel peut effectuer une analyse des sentiments, permettant de comprendre les réactions émotionnelles associées à des sujets spécifiques sur les médias sociaux.

- Cartographie des influences : En identifiant les comptes sociaux influents et les interactions, notre logiciel offre une cartographie des influences, révélant ainsi les acteurs clés qui contribuent à la diffusion de certains contenus.
- Détection d'événements : Notre logiciel peut détecter la montée d'événements ou de sujets spécifiques sur les médias sociaux, facilitant ainsi la réponse rapide aux développements en cours.

Dans l'ensemble, l'analyse des médias sociaux via notre logiciel fournit une perspective approfondie des dynamiques en jeu sur ces plateformes. Elle permet aux utilisateurs de rester informés, d'anticiper les tendances émergentes, et de comprendre l'impact des contenus partagés au sein de la sphère sociale en ligne.

Pour les utilisateurs généraux, le logiciel offrirait la possibilité de comprendre immédiatement des termes spécifiques, même s'ils sont récemment créés. Notre logiciel se distingue par sa capacité à faciliter une compréhension rapide et précise des mots communs dérivés de noms propres. En explorant les caractéristiques spécifiques de notre technologie, examinons quelques points clés :

- Réponse rapide : Notre logiciel permet une réponse instantanée en fournissant des définitions claires et concises des termes dérivés, soutenant ainsi une communication réactive et efficace.
- Contextualisation immédiate : Grâce à sa fonction de contextualisation, le logiciel assure une compréhension instantanée en présentant les significations des termes dans des contextes pertinents, facilitant ainsi une utilisation appropriée dans la communication quotidienne.
- Actualisation dynamique : le logiciel actualise ses informations linguistiques pour s'adapter aux évolutions du langage, assurant une pertinence constante dans la compréhension instantanée des termes couramment utilisés.
- Facilitation de la communication fluide : La fonction "communication et compréhension instantanée" facilite une communication fluide en éliminant les barrières liées à la compréhension des mots dérivés de noms propres, encourageant ainsi des échanges clairs et efficaces.

En enrichissant continuellement sa base de données, le logiciel vise à améliorer la précision contextuelle, permettant une interprétation fine des termes, y compris ceux qui résultent de noms propres.

La fonction de "communication et compréhension instantanée" de notre logiciel, offre une expérience utilisateur dynamique en favorisant une compréhension rapide et précise des termes, soutenant ainsi une communication fluide et contextuellement pertinente.

L'adaptabilité aux évolutions culturelles signifie que le logiciel pourrait servir d'outil pour comprendre comment le langage évolue en tandem avec les changements culturels, politiques et sociaux.

Tout en identifiant les nouveaux termes qui émergent et en fournissant une perspective sur les tendances linguistiques liées à ces évolutions. Plus précisément ce qui est relatif à la réflexion des tendances culturelles, la sensibilité aux changements sociaux et le suivi des influences politiques.

Les termes dérivés de noms propres souvent reflètent des éléments culturels, tels que des mouvements sociaux, des personnalités influentes, ou des événements significatifs. En suivant ces termes, le logiciel pourrait fournir des renseignements sur les tendances culturelles en émergence.

Qui plus est, les mots qui émergent dans la langue peuvent refléter les préoccupations et les changements dans la société. Le logiciel pourrait ainsi aider à identifier comment le langage évolue pour aborder de nouveaux sujets ou concepts culturels.

De plus, certains termes dérivés de noms propres peuvent être liés à des personnalités politiques ou à des mouvements politiques. Le logiciel pourrait aider à suivre ces influences et à comprendre comment elles se manifestent dans le langage.

Chapitre 1

Cadrage théorique

Pour commencer, nous entamons notre chapitre en procédant à une revue de littérature qui présentera les fondements théoriques, les auteurs qui ont délimité les contours fondamentaux de notre champ d'étude. Cela consiste en la présentation des avancées conceptuelles ayant émergé dans le domaine de l'informatique, la lexicologie et de la morphologie, qui sont au cœur de notre étude. Par la suite, nous poursuivons notre recherche en définissant les concepts clés qui serviront de fondement théorique à notre étude, afin de clarifier les notions principales et établir un cadre conceptuel cohérent pour notre analyse.

L'état actuel de la question est marqué par des recherches indépendantes dans des domaines variés, chacune apportant des perspectives distinctes. La morphologie computationnelle explore la structure interne des mots, tandis que les ressources lexicales et ontologies fournissent des bases de données cruciales sur les relations lexicales. En plus de l'apprentissage automatique, notamment à travers des modèles de langage pré-entraînés, qui a considérablement impacté le paysage du TALN.

Cependant, une synthèse holistique de ces diverses approches est encore à réaliser. Nous visons ici à esquisser les principales avancées dans chaque domaine tout en soulignant la nécessité d'une approche intégrée pour aborder cette problématique émergente.

Comprendre la dynamique complexe des noms propres et des mots dérivés. Ce qui requiert une analyse approfondie des interactions lexicales, tout en explorant comment les modèles de TALN peuvent être adaptés pour capturer ces nuances subtiles. Ce travail cherche à jeter les bases d'une perspective unifiée, établissant ainsi un cadre conceptuel pour guider les futures recherches dans ce domaine prometteur.

Parmi ces disciplines nous avons cité la morphologie computationnelle qui constitue une branche essentielle de la linguistique computationnelle. Elle se penche sur l'analyse structurelle des mots et de leurs formes dans une langue donnée.

Elle offre un cadre méthodologique pour décomposer les mots en unités plus petites, appelées morphèmes, et examine comment ces unités interagissent pour former des mots complexes.

Dans le contexte du traitement automatique des mots dérivés à partir de noms propres, la morphologie computationnelle revêt une importance particulière en raison de son rôle dans la modélisation des processus morphologiques et de son impact sur la génération lexicale.

Pour cet aspect de notre travail, nous nous référons à l'ouvrage *« MorphologicalAnalysis and Generation:A First-Step in Natural LanguageProcessing »* de Kenneth R. Beesley qui constitue une référence classique dans le domaine de la linguistique computationnelle. Il offre une exploration approfondie des concepts fondamentaux liés à l'analyse et à la génération morphologiques, des sujets cruciaux pour la compréhension des structures lexicales dans les langues naturelles. L'auteur aborde les principes théoriques et pratiques de l'analyse morphologique, mettant en lumière les différentes approches et méthodologies utilisées pour décomposer et comprendre la structure interne des mots.

Kenneth R. Beesley explore également la génération morphologique, examinant comment les règles morphologiques peuvent être appliquées pour produire de nouveaux mots. L'ouvrage couvre diverses techniques, notamment l'utilisation de règles morphologiques, d'automates finis, et d'autres modèles formels. Il met l'accent sur la création de modèles linguistiques efficaces qui peuvent être utilisés dans des systèmes de traitement automatique du langage naturel.

La morphologie en TAL a aussi connu des avancées grâce aux travaux, notamment de Kenneth Beesley qui est un expert en linguistique computationnelle également l'un des auteurs du livre "Finite State Morphology". Il a travaillé sur des méthodes basées sur les automates finis pour modéliser la morphologie.

Un article paru en 2004 écrit par G. Dal, Nabil HATHOUT et Fiammetta NAMER nommé « Morphologie constructionnelle et traitement automatique des langues : le projet MORTAL » explore aussi la connexion entre la morphologie constructionnelle et le traitement automatique des langues (TAL).

Il est à souligner que l'objectif de leur travail n'est pas seulement théorique mais également applicatif. En référence à FRADIN (1994a), les auteurs indiquent que, bien qu'ils puissent utiliser des aspects théoriques, leur contribution dépasse le cadre purement conceptuel.

L'article se divise en deux sections. Dans la première, l'auteur offre une brève présentation de chaque discipline (morphologie constructionnelle et TAL), suivie d'un état des lieux sur les relations existantes entre ces deux domaines. La seconde section est consacrée à un projet en cours de construction de bases de données constructionnelle pour le TAL, appelé "MorTAL".

Cette partie détaillera rapidement le projet en question. Ensuite, l'article met en avant certains des avantages que le TAL peut tirer de ce projet, en se concentrant spécifiquement sur deux domaines d'application : la recherche d'information et la fouille de textes.

Ajoutant à cela, Kimmo Koskenniemi, linguiste finlandais, dont le travail a influencé la morphologie computationnelle, en particulier dans le développement du système de désambiguïsation morphologique appelé *« Two-Level Morphology ».*

Il est important de souligner la liste des chercheurs nommés précédemment n'est pas exhaustive, et de nombreux chercheurs dans le domaine de la linguistique computationnelle peuvent également être impliqués dans des recherches liées à la morphologie computationnelle.

1. Les ressources lexicales et les ontologies

Représentent des piliers essentiels dans le domaine du Traitement Automatique du Langage Naturel (TALN), fournissant une infrastructure sémantique pour l'analyse et la compréhension du langage. Ces deux types de ressources, bien que distincts, convergent dans leur objectif de capturer la complexité et la richesse du sens linguistique.

Les ressources lexicales englobent diverses bases de données, dictionnaires, thésaurus et corpus linguistiques. Elles recueillent des informations sur les mots, allant des définitions et des synonymes aux catégories grammaticales et aux nuances d'utilisation. Ces ressources offrent une vue détaillée de la sémantique et de la structure morphologique des mots dans une langue.

Dans le contexte du TALN, les ressources lexicales sont essentielles pour des tâches telles que la lemmatisation, l'analyse de sentiment et la désambiguïsation sémantique. Elles fournissent un socle de connaissances pour enrichir la représentation lexicale, améliorant ainsi la précision des systèmes linguistiques automatisés.

Les ontologies, d'autre part, dépassent le cadre des ressources lexicales en offrant une représentation formelle et structurée des connaissances. Elles modélisent les entités et les relations dans un domaine spécifique. L'élaboration d'ontologies peut être manuelle, semi-automatique, ou automatique, impliquant la définition de classes, de propriétés et de relations.

Dans le TALN, les ontologies sont des outils puissants pour la sémantique lexicale, la résolution des références et la recherche d'information. Elles ajoutent une dimension contextuelle en décrivant comment les concepts sont liés, permettant ainsi une compréhension plus profonde du sens au-delà du niveau lexical.

Malgré leur utilité, l'intégration de ces ressources n'est pas sans défis. L'hétérogénéité des données, la nécessité de maintenir et mettre à jour ces ressources, ainsi que l'adaptation à des domaines spécifiques sont des enjeux constants. La dynamique évolutive des langues pose également des questions sur la pertinence et la représentativité continues de ces ressources.

En réunissant des ressources lexicales détaillées et des ontologies structurées, nous pouvons réaliser des avancées significatives dans la compréhension et la génération du langage naturel. La combinaison de ces deux types de ressources offre un potentiel considérable pour construire des systèmes linguistiques automatisés plus sophistiqués et contextuellement informés ce qui, dans le cadre d'un projet impliquant le traitement automatique des mots dérivés à partir de noms propres, l'utilisation de ressources lexicales et d'ontologies peut apporter plusieurs avantages significatifs.

Ces ressources peuvent contribuer à la réalisation et à l'amélioration de votre projet en aidant avec :

1.1. La désambiguïsation sémantique

Les ressources lexicales et les ontologies offrent une sémantique riche pour les mots, ce qui peut être crucial pour la désambiguïsation sémantique. Lorsque nous travaillons avec des mots dérivés de noms propres, la signification peut varier en fonction du contexte. Les informations sémantiques fournies par ces ressources peuvent aider à déterminer la signification appropriée dans un contexte particulier.

1.2. Les relations lexicales et structure morphologique

Les ressources lexicales détaillent les relations lexicales telles que les synonymes, les antonymes, et les termes associés. Ces informations peuvent être exploitées pour comprendre comment les mots dérivés sont liés aux noms propres d'origine. De plus, la structure morphologique fournie par ces ressources peut faciliter l'analyse des composants morphologiques des mots dérivés.

1.3. La contextualisation avec des ontologies

Les ontologies peuvent contextualiser les mots dérivés en décrivant comment ils s'inscrivent dans un ensemble plus vaste de connaissances. Cela peut être particulièrement utile lorsque les mots dérivés sont utilisés dans des domaines spécialisés où la terminologie peut être spécifique et nécessiter une compréhension approfondie du contexte.

1.4. Amélioration de la recherche d'information

Si à terme, le projet implique la recherche d'information, l'intégration d'ontologies peut améliorer la pertinence des résultats en fournissant une structure sémantique pour les requêtes et les documents. Cela peut contribuer à une recherche plus précise et à une récupération d'informations plus adaptée.

1.5. La gestion des ambiguïtés morphologiques

Les ressources lexicales peuvent aider à gérer les ambiguïtés morphologiques associées aux mots dérivés. En comprenant la morphologie des mots de manière détaillée, il devient possible de mieux traiter les variantes morphologiques et d'améliorer la robustesse des analyses.

En intégrant judicieusement ces ressources dans notre logiciel, nous pouvons renforcer la compréhension sémantique des mots dérivés à partir de noms propres, améliorer la qualité des analyses morphologiques, et potentiellement élargir le champ d'application de notre logiciel en le rendant plus adaptable à des contextes variés et spécialisés.

2. Les domaines spécifiques et approches spécialisées dans la dérivation de mots à partir de noms propres

La considération des domaines spécifiques est cruciale pour les recherches portant sur la dérivation de mots à partir de noms propres. Certains travaux de recherche se sont penchés sur des secteurs particuliers tels que la politique, la biologie, ou d'autres disciplines, où la dérivation de mots liés à des noms propres revêt une signification distinctive et est fréquemment utilisée.

Chaque domaine spécialisé introduit des nuances uniques dans le lexique et la sémantique. Par exemple, dans le domaine politique, les mots dérivés de noms propres peuvent encapsuler des idéologies politiques spécifiques, des mouvements sociaux ou des concepts propres à ce secteur.

Dans le domaine de la biologie, ces dérivations peuvent se référer à des classifications taxonomiques, des découvertes scientifiques ou des entités spécifiques au domaine biomédical.

La nécessité d'approches spécialisées découle de la diversité de ces contextes. Les approches génériques peuvent ne pas être suffisantes pour traiter efficacement la dérivation de mots, nécessitant ainsi des méthodologies adaptées à chaque domaine.

Cette adaptation peut impliquer la création de ressources lexicales spécialisées, l'ajustement de paramètres dans les modèles de traitement du langage naturel, ou le développement de règles morphologiques spécifiques.

L'objectif ultime de ces approches spécialisées est d'assurer une adéquation plus étroite avec les particularités morphologiques, sémantiques et contextuelles propres à chaque secteur. La précision contextuelle devient alors essentielle, surtout dans des domaines où la signification d'un mot dérivé peut varier considérablement selon le contexte.

Les applications pratiques de ces approches spécialisées sont visibles dans des secteurs tels que l'analyse politique, la recherche biomédicale, ou d'autres domaines d'expertise. Elles se traduisent par une meilleure extraction d'informations, la création de ressources lexicales adaptées, et des analyses sémantiques plus précises dans des contextes spécifiques.

Ainsi, l'exploration des domaines spécifiques dans la dérivation de mots à partir de noms propres nécessite une compréhension approfondie des particularités de chaque secteur, incitant au développement d'approches spécialisées pour répondre de manière précise et effective aux défis et aux exigences spécifiques de ces domaines.

Le langage, en tant que système complexe et évolutif, offre une riche palette d'unités linguistiques. Parmi celles-ci, les morphèmes dérivationnels jouent un rôle très important dans la formation de nouveaux mots et la création du sens. Ils élargissent l'extension du vocabulaire, en offrant une flexibilité dans l'expression linguistique.

Les morphèmes dérivationnels agissent comme des outils morphologiques puissants qui contribuent à la vitalité et à l'évolution d'une langue. Ils offrent une souplesse linguistique en permettant à la langue de s'adapter aux changements culturels, sociaux et technologiques. Ces unités minimales douées de sens facilitent la génération de nouveaux termes en réponse à de nouvelles réalités ou concepts.

Notre projet de recherche s'articule autour des ressources linguistiques exhaustives, en s'appuyant sur la collecte, de l'analyse, et de la classification qui vont permettre l'analyse et la compréhension des morphèmes dérivationnels dans différentes langues. Nous mettrons en usage un outil qui pourrait servir de référence ou une source pour les chercheurs, les linguistes, les étudiants et autres.

Le processus de création implique la sélection judicieuse de sources linguistiques variées, la mise en place d'une méthodologie rigoureuse de collecte de données. Ainsi que l'utilisation de techniques d'analyse morphologiques avancées.

Cet outil offre non seulement une référence clé pour l'exploration détaillée des mécanismes morphologiques, mais il joue également un rôle important dans des domaines tels que la lexicographie, la traduction, et la recherche linguistique comparative.

La collecte de données se fera à partir de sources lexicales variées, avec une attention particulière portée à la précision et à la représentativité. Les étapes clés du processus seront comme suit : la classification des morphèmes, la création d'une base de données structurée et la conception d'une interface intuitive.

Nous contribuons à constituer une sorte de dictionnaire de néologismes, organisé et accessible, capable de répondre aux besoins de la communauté linguistique académique. Nous espérons également de faire encourager de futures recherches dans ce domaine interdisciplinaires.

Notre étude englobe les domaines de la lexicologie, cette dernière inclut l'étude des mots et leur formation (la morphologie lexicale), l'étude de leur signification (la sémantique lexicale). Nous nous basons sur des ressources lexicologique et morphologiques, ces concepts linguistiques nous servent pour la création d'un logiciel qui sera apte d'interpréter les mots communs issus de noms propres de personnes.

3. Autour de la lexicologie, la morphologie, dérivation et les néologismes

La lexicologie, en tant que discipline linguistique, trouve ses fondements dans l'œuvre posthumes de Ferdinand De Saussure, notamment son ouvrage emblématique paru en 1916 nommé « Cours de linguistique générale ». Grace à cette œuvre majeure, Saussure entreprend une étude détaillée de la structure et de la nature du langage, avec une attention particulière portée à la signification des unités lexicales.

Sa réflexion sur la nature du signe linguistique, caractérisé par un signifiant et un signifié, a établi les bases pour une appréhension plus approfondie du rôle du lexique dans la communication linguistique.

En étudiant la relation entre le signifiant (la forme phonique ou graphique) et le signifié (le concept ou le sens), Saussure a ouvert la voie à une analyse plus systématique des unités lexicales et de leur fonctionnement dans le cadre linguistique. (Ferdinand de SAUSSURE, 2016)

En effet, la lexicologie a utilisé les concepts et les théories énoncés dans les travaux de Saussure, afin d'évoluer comme une discipline de premier plan au sein de la linguistique, tout en étudiant la formation des mots, leur signification et l'utilisation des mots dans divers contextes linguistiques et culturels.

La lexicologie a émergé comme une discipline autonome grâce aux travaux de Georges MATORÉ, en particulier avec la publication de son ouvrage majeur, « La méthode en lexicologie » en 1953. Ce texte fondateur s'est imposé comme une référence essentielle dans le domaine, cherchant à établir une approche méthodique et précise pour l'étude approfondie du lexique. (MATORÉGeorges,1953)

Alise. LEHMANN et Françoise. MARTIN BERTHET, dans leur ouvrage qui s'intitule « La lexicologie : sémantique, morphologie, lexicographie », publié en 2013, nous offre un large aperçu sur différentes dimensions du lexique dans la linguistique contemporaine.

A travers des différents chapitres les auteurs abordent de manière détaillée la sémantique des mots et les différentes relations qu'ils entretiennent, les différents processus de formation des mots. Ainsi que leurs différentes classes et leurs propriétés morphologiques et les méthodes utilisées en lexicographie pour organiser et élaborer le lexique d'une langue.

La lexicologie est selon Alise LEHMANN et Françoise. Martin BERTHET, une discipline qui étudie le lexique d'une langue, c'est-à-dire l'ensemble des mots et des unités lexicales qui la composent. Elle examine les relations entre les mots, leur formation, leur sens et leur évolution historique.

Igor A. MEL'CUK, André CLAS, Alain POLGUERE, dans leur ouvrage qui s'intitule *« Introduction à la lexicologie explicative et combinatoire »,* publié en 1995, présentent une approche théorique et pratique de l'analyse lexicale. Les auteurs étudient les différentes dimensions du lexique, les relations sémantiques et morphosyntaxiques qui sous-tendent la structure des mots et la combinaison de ces derniers dans le discours.

Georgette DAL linguiste et professeur, elle a publié un article en 2019 qui s'intitule « État actuel sur les études en morphologie en France et à l'international », où elle décrit la situation contemporaine de la morphologie de façon générale.

Elle a abordé les grandes questions qui traversent le champ de la discipline, ces questions englobent l'autonomie de la morphologie par rapport aux autres composantes de la grammaire, l'unité de compte de la morphologie ainsi que les différentes approches en concurrence avec cette dernière[1].

L'auteur définit la morphologie comme étant l'étude de la covariation récurrente de forme et du sens des lexèmes, des grammèmes et des formes qu'ils revêtent en contexte. Dans l'article, elle a parlé des fluctuations qu'elle a connu la morphologie au fil du temps, avant le structuralisme la morphologie occupait une position prédominante dans l'analyse linguistique. Cependant au début de la grammaire générative, elle a été un peu mise de côté au profit de la syntaxe et de la phonologie.

Dans le cadre linguistique, la morphologie a réalisé des améliorations grâce aux travaux de plusieurs linguistes comme Danielle CORBIN, linguiste et docteur ès lettres connue pour ses travaux sur la morphologie dérivationnelle. Dans son ouvrage nommé « Morphologie dérivationnelle et structuration du lexique », publié en 1987.

[1] https://hal.science/hal-03639345/document

Elle aborde la manière dont les mots dérivés contribuent à la structuration du lexique dans une langue, ainsi que l'importance du processus de dérivation dans la création et l'organisation du lexique. Elle explique comment les processus de dérivation influencent la formation de nouveaux mots et comment ces derniers s'intègrent dans le système lexical existant.

Jean MOLINO par son article qui s'intitule « Où en est la morphologie » publié en 1985, présente une large analyse de l'état actuel de la morphologie linguistique, en particulier dans le contexte de la morphologie dérivationnelle du français.

D'après Jean. MOLINO, la morphologie est une discipline de la linguistique qui étudie la structure des mots, en particulier les unités morphologiques, c'est-à-dire les mots et les formes grammaticales, ainsi que les règles qui régissent la formation des mots[2].

MOLINO dans son article parle des fondements théoriques qui animent la discipline, il souligne que la morphologie dérivationnelle a connu une certaine autonomie au cours des trente dernières années, mais qu'elle n'est pas toujours clairement délimitée par rapport à d'autres domaines de la linguistique, comme la lexicologie. Il est à préciser que MOLINO, insiste sur la prolongation de l'exploration dans ce domaine, tout en s'adaptant aux nouvelles questions et d'intégrer des avancées théoriques et méthodologiques pour mieux comprendre la structure morphologique des langues.

Bernard FRADIN, linguiste français reconnu grâce à ses travaux qui portent sur la morphologie dans le domaine de la linguistique, auteur de nombreux articles et ouvrages dont « Nouvelles approches en morphologie », paru en 2003, constitue une référence majeure dans le domaine.

[2] https://www.persee.fr/doc/lgge_0458-726x_1985_num_20_78_2462

FRADIN a travaillé sur l'étude des processus de formation des mots, particulièrement la dérivation et la composition, ainsi que l'analyse des interactions entre la morphologie et d'autres aspects du langage, comme la syntaxe et la sémantique[3].

Par ailleurs, l'ouvrage *« La construction du lexique français »* de Denis APOTHELOZ, publié en 2002, fait partie de l'une des références usuelles exploité dans le domaine de la morphologie. Cet ouvrage représente un consensus assez général à propos du champ organisationnel de la morphologie, il a apporté des contributions notables à la compréhension des processus complexes impliqués dans la formation et l'évolution du vocabulaire de la langue française.

Denis APOTHELOZ, estime que la morphologie s'organise de la façon suivante : Tout d'abord, elle oppose la morphologie flexionnelle, dont le domaine est constitué par les variations formelles que subissent les mots en rapport avec leur fonction dans la phrase (conjugaison, déclinaison) et la morphologie lexicale, qui s'intéresse à la structure interne des unités lexicales.

Elle distingue la morphologie dérivationnelle (ou constructionnelle), qui s'intéresse à la construction des mots au moyen des affixes dérivationnels, et la morphonologie qui étudie les aspects phonologiques et prosodiques des morphèmes et des mots. (Denis. APOTHELOZ, 2002 P : 01)

La langue est un système vivant qui réfute toute conception statique ou immuable car elle se transforme continuellement en réponses aux besoins et aux évolutions de la société. L'évolution d'une langue est un phénomène intrinsèque à sa nature dynamique et à son utilisation par les locuteurs, ce qui en résulte plusieurs facteurs comme les changements sociaux, culturels, technologique et même idéologique et politique.

[3]https://www.cairn.info/nouvelles-approches-en-morphologie--9782130515487.htm

Ce phénomène se produit lorsque les individus d'une société se mettent à communiquer au moyen d'une langue donnée. Cette dernière subit inévitablement des transformations au fil du temps.

Comme le signale Jean PRUVOST et Jean-François SALLAYROLLES :

> « La communication entre les êtres humains passe en effet originellement par la création de mots pour désigner l'univers qu'ils perçoivent, les sentiments et les pensées qui les animent. Manifestation de l'activité symbolique de l'homme, les choses, les idées et els et les faits par des sons, des signes qui en sont les substituts. Quelles que soit l'interprétation métaphysique, biologique ou linguistique, le langage est toujours inscrit dans un processus langagier créatif et donc néologique. » (2003, P :04)

La néologie se présente comme un phénomène linguistique complexe, ce dernier nécessite à la fois des besoins évolutifs de la langue et des nuances sociétales inhérentes à son usage. Sa compréhension requiert une approche multidimensionnelle pleinement son impact et sa portée dans le cadre qui intègre des considérations lexicologiques et sociolinguistiques pour appréhender linguistique et social.

Selon, Jean –François SABLAROLLES *« la néologie est un processus par lequel le lexique d'une langue s'enrichit »* (2000, P :56). Il est à souligner que la néologie est un domaine interdisciplinaire qui fait appel à d'autres domaines comme la lexicologie, la morphologie, etc.

L'étude néologique a suscité l'intérêt de plusieurs chercheurs comme L. Guilbert qui est l'un des grands théoriciens. Dans son ouvrage *« Créativité lexicale »* publié en 1975, il a étudié les mécanismes de formation de nouveaux mots et de création lexicale. Il a abordé les processus cognitifs et linguistiques impliqués dans la création de néologismes, ainsi que l'influence des contextes socio-culturels sur l'émergence de nouveaux mots.

Jean-François SABLAYROLLES avec son ouvrage reconnu sous le titre « La néologie du français contemporain » où il étudie les différentes sources et processus de formation des néologismes, notamment l'innovation sémantique, morphologique et syntaxique. Ainsi que l'impact des domaines spécifiques comme la technologie, la culture populaire et les médias sur la création lexicale.

Pour Jean –François SABLAYROLLES, chaque unité lexicale et sens utilisé dans la langue pour la première fois est un néologisme. Ce phénomène découle sur des impératifs communicatifs divers, comme la nécessité de nommer des réalités, de nouveaux concepts, etc.

4. La lexicologie

La lexicologie est une branche de la linguistique, c'est une discipline descriptive. Elle décrit et étudie le lexique d'une langue donnée afin d'arriver à expliquer les formations, les constructions et le sens des mots de la langue.

Ce qui permet la compréhension de certain concepts basiques comme (morphème, racine, préfixe, suffixe…), qui sont utiles pour la conception de notre logiciel. La lexicologie désigne plus particulièrement la science qui étudie le lexique ou le vocabulaire d'une langue.

Pour LEHMANN Alise, M-BERTHET, (2013, P : 13)

> « La lexicologie a pour tâche d'inventorier les unités qui constituent le lexique, et de décrire les relations entre ces unités. Le lexique en effet n'est pas une simple liste qu'on ne pourrait ordonner que par l'ordre alphabétique ; il s'organise sur les deux plans du sens et de la forme :
>
> - La sémantique lexicale étudie l'organisation sémantique du lexique : elle analyse le sens des mots et les relations de sens qu'ils entretiennent entre eux.
> - La morphologie lexicale étudie l'organisation formelle du lexique : elle analyse la structure des mots et les relations de forme et de sens qui existent entre eux ».

La lexicologie se concentre sur l'analyse et la classification des unités lexicales d'une langue, qu'elles soient des mots simples ou construit. Son objectif principal est de décomposer le lexique en différentes unités et de comprendre les relations qui existent entre elles.

Pour effectuer une étude des unités lexicales ou du mot, la lexicologie s'organise de deux manières, en morphologie lexicale qui examine la structure interne des mots, et en sémantique lexicale qui explore les significations et les relations de sens entres les mots.

La forme d'un mot peut souvent donner des indices sur son sens et sa fonction dans une langue donnée, tandis que la sémantique, l'étude de la signification des mots, analyse les différentes nuances et connotations associées à ces mots.

Ainsi, pour comprendre pleinement un mot et son utilisation dans une langue, il est nécessaire d'étudier à la fois sa forme et sa signification, ce qui montre l'interdépendance étroite entre la lexicologie, la sémantique et la morphologie.

4.1. La lexicographie

Selon Igor A. MEL'CUK, André CLAS, Alain POLGUERE,

« La lexicographie est une discipline appliquée qui qui a pour objet l'élaboration de dictionnaire ; la lexicographie doit faire siens et utiliser en pratique les résultats théoriques dégagés par la lexicologie ». (1995, P : 26)

La lexicographie est donc l'ensemble des techniques qui visent à l'élaboration d'un dictionnaire. Elle est considérée à la fois comme une discipline scientifique et pratique, car elle aboutit à l'élaboration d'un objet commode qui a un visé pragmatique qui nous aide à comprendre la langue.

4.2. Le lexique et la notion du mot

Le mot peut être définis comme la forme linguistique la plus petite qui ait une autonomie. (NIKLAS-SALMIEN Aino, 1997, P : 20).

La notion de mot peut sembler simple et intuitive à première vue, mais elle suscite en réalité des débats et des réflexions approfondies. Nous avons besoin de chercher à définir et à clarifier cette notion, pour pouvoir préciser le sens du mot qu'on veut utiliser dans notre logiciel.

Pour André MARTINET, le mot, *« c'est un syntagme autonome formé de monèmes non séparable, le monème étant l'unité significative minimale que l'on peut dégager dans la chaîne parlée »* (François GAUDIN et Louis GUESPIN, 2000, P : 211).

Selon Jean DUBOIS, le mot est défini de la façon suivante *« En linguistique traditionnelle, le mot est un élément linguistique significatif composé d'un ou de plusieurs phonèmes. Cette séquence est susceptible d'une transcription écrite (idéogrammatique, syllabaire ou alphabétique) comprise entre deux blancs »* (Dubois et Al, 2012, P : 312)

Il y a quatre tentatives de définitions de la notion de mot, mais dans notre travail de recherche il y a seulement deux qui nous intéresse qui sont :

Le mot graphique : ce concept désigne une séquence de caractères alphabétiques délimitée par des espaces blancs. Cependant, une même séquence de caractères peut renvoyer à plusieurs mots grammaticalement distincts. Par exemple, le mot "chantais" peut indiquer soit une conjugaison à la première personne du singulier, soit à la deuxième personne du pluriel à l'imparfait.

Le mot sémantique : d'un point de vue sémantique, le mot représente une unité linguistique porteuse d'une signification qu'on peut identifier facilement au sein d'une phrase, cette unité sémantique devrait correspondre à une unité graphique. En revanche, il est possible qu'une séquence de plusieurs mots graphiques compose un seul mot sémantique, le cas des mots composés comme, "pomme de terre", "porte-avions" ou "qu'en-dira-t-on".

On déduit par ces tentatives de définition que dans la langue, le mot n'est pas la plus petite unité porteuse de signification, faisons référence à André Martinet, le premier qui a affirmé que le mot n'est pas la petite unité de signification. La plus petite unité douée de sens c'est le morphème, le morphème à son tour peut être un lexème comme il peut être un grammème.

5. La morphologie

La morphologie est une branche de la linguistique qui étudie la structure interne des mots, elle se concentre principalement sur la structure et l'organisation de ses derniers. Selon Jean Dubois (2012, P : 311)

> «La morphologie est la description des règles qui régissent la structure interne des mots c'est-à-dire les règles de la combinaison entre les morphèmes racines pour constituer des mots (règles de formation de mots, préfixation et suffixation) et la description des formes diverses que prennent ces mots selon la catégorie de nombre, du genre, de temps et de personne, et selon le cas (flexion nominale ou verbale), par opposition à la syntaxe qui décrit les règles de combinaison entre les morphèmes lexicaux (morphèmes, racines et mots) pour constituer des phrases. » (2012, P : 311)

La morphologie s'intéresse aux différentes parties d'un mot, appelées morphèmes, qui peuvent être des préfixes, des suffixes ou des racines.

L'objectif de la morphologie est d'analyser comment ces morphèmes se combinent pour former des mots et comment ils peuvent subir des modifications pour exprimer différentes significations grammaticales.

5.1. La morphologie dérivationnelle

Dans notre travail de recherche, la morphologie dérivationnelle c'est la branche qui nous intéresse le plus, vu qu'elle est considérée comme une partie de la linguistique qui prend en charge les nouveaux mots qui se forment à partir d'unités linguistique déjà existante. Elle étudie également les différents procédés de formation de nouveaux mots.

Selon GARDES-Tamine, Joelle, la morphologie dérivationnelle *: « Elle concerne la formation des mots et consiste à créer de nouvelles unités lexicales par l'adjonction d'un affixe à une base d'un affixe. »* (2008, P : 73)

5.2. Le morphème

Dans notre recherche nous avons choisi d'utiliser le terme morphème, principalement pour des raisons de clarté et de précision. Le morphème est considéré comme un terme standard utilisé par les linguistes.

Le morphème représente l'élément de base de la structure grammaticale d'un mot Pour Denis APOTHLÉOZ, le morphème est « *la plus petite unité formelle dotée d'un sens ; on dit aussi que c'est une unité significative minimale*. » (Denis APOTHLÉOZ, 2002, P :3).

La notion de morphème est fondamentale en linguistique pour comprendre la structure des mots et leur fonctionnement. Cette définition implique que chaque morphème contribue à la signification d'un mot ou d'une partie de mot.

L'analyse morphologique implique la décomposition des mots en leurs éléments constitutifs, les morphèmes, qui sont les unités minimales porteuses de sens.

Pour pouvoir identifier ces unités fondamentales de sens et de structure dans une langue, nous procédons à la segmentation qui consiste à identifier et isoler ces morphèmes au sein des mots.

En appliquant des tests grammaticaux comme la commutation et la distribution, nous pourrons déterminer le rôle et la fonction des morphèmes dans une construction phrastique.

Dans le contexte de notre recherche, pour savoir comment une machine peut séparer une phrase en mots il était nécessaire de savoir comment nous le faisons dans les langues naturelles.

Pour ce faire nous tenons compte de la théorie structuraliste de Leonard Bloomfield, qui considère le morphème comme la plus petite unité porteuse de sens, non segmentale. Pour qu'une partie d'un mot puisse être considérée comme un morphème, il est essentiel qu'elle puisse être substituée par un autre élément. Le morphème que ce soit un lexème ou un grammème, il vient apporter la notion de pertinence qui n'est pas apporté par le mot.

Les morphèmes se divisent en deux catégories principales : les morphèmes grammaticaux, également appelés grammèmes, et les morphèmes lexicaux, connus sous le nom de lexèmes. D'après Bernard POITIER, *« Les lexèmes ou morphèmes lexicaux s'opposent aux grammèmes ou morphèmes grammaticaux* » (François GAUDIN et Louis GUESPIN, 2000, P : 214).

5.2.1. Les morphèmes grammaticaux

Ils désignent des unités appartenant à une liste fermée de la grammaire, les morphèmes grammaticaux donnent des indices de catégorie grammaticale comme (le temps, le mode, la personne, le genre, le nombre). Comme elle affirme NIKLAS-SALMIEN Aino :

> « Les morphèmes grammaticaux jouent un rôle décisif dans l'organisation grammaticale de la phrase, qu'il s'agisse des marques morphosyntaxiques (nombre, genre, personne, temps et mode) ou des mots-outils qui marquent les relations entre mots et groupes de mots dans la structure phrastique (prépositions et conjonctions) ou qui assurent l'actualisation d'une autre partie du discours (les déterminants). Les morphèmes grammaticaux servent à transmettre des notions générales souvent axées sur la situation d'énonciation, contrairement aux morphèmes lexicaux qui véhiculent des concepts ayant leur spécificité propre. » (1997, P : 22).

Il existe deux types de morphèmes grammaticaux, dans notre cas nous nous intéressons aux morphèmes grammaticaux liés :

- **Les morphèmes grammaticaux liés**

Comprenant les désinences de nombre, de genre, de personne, de temps, ainsi que les préfixes et les suffixes. Ces des éléments linguistiques qui ne peuvent pas exister de manière autonome, mais qu'ils doivent être attachés à d'autres mots pour former un sens complet. Ces derniers sont répartis en deux sous-catégories distinctes :

- **Les morphèmes dérivationnels**

Cette classe désigne les morphèmes dérivationnels tels que les suffixes et les préfixes, ils servent à former de nouveaux mots qu'on appelle les dérivés. Les morphèmes dérivationnels modifient le sens du lexème auquel ils s'adjoignent, et peuvent également modifier sa catégorie grammaticale. La préfixation ne change pas la catégorie grammaticale de la base, mais la suffixation opère souvent le transfert de la catégorie.

Les préfixes sont des morphèmes ajoutés au début d'un lexème pour modifier son sens ou sa fonction. Les suffixes, quant à eux, sont des morphèmes ajoutés à la fin d'un lexème pour en modifier le sens ou la fonction.

Nous avons utilisé la signification des morphèmes dérivationnels dans notre logiciel pour faciliter l'interprétation des néologismes, (mots dérivés de noms propres de personnes)

- **Les morphèmes flexionnels**

Ce sont des unités porteuses de sens purement grammaticales, et ils ne créent pas de nouveaux mots mais des formes différentes d'un même mot. En français ils sont toujours postposés ils représentent les désinences et les marques du genre et du nombre. Nous avons utilisé les morphèmes flexionnels dans notre programme pour gérer les variations des morphèmes dérivationnelles.

5.2.2. Les lexèmes

Sont les formes de base des mots, ils trouvent leur place dans lexique, se sont eux qui apportent la signification des mots. Les morphèmes lexicaux n'ont pas toujours besoin de grammèmes pour exister dans la langue. En revanche, ont souvent besoin de grammèmes pour être actualisées dans une phrase.

6. Le radicale et la base d'un mot

Le radicale est la partie autonome d'un mot, qui peut se trouver dans le lexique qui contient son sens principal et qui peut être modifiée par l'ajout de préfixes, suffixes ou infixes pour former de nouveaux mots ou des variantes de sens.

Selon Denis APOTHELOZ *« Le radicale est le morphème lexical qui demeure lorsque tous les affixes, qu'ils soient dérivationnels ou flexionnels, ont été retirés d'un mot. Il constitue ainsi la base ou le noyau sémantique d'un mot. »* (2002, P : 16).

D'après Denis APOTHELOZ, la base est un élément qui peut comporter deux ou plusieurs morphèmes. Elle peut être un mot simple ou une partie d'un mot complexe sur laquelle des affixes peuvent être ajoutés pour créer de nouveaux mots.

La notion de radicale est essentielle en morphologie, car elle permet d'analyser la structure interne des mots et de comprendre comment les éléments constitutifs contribuent à leur sens et à leur fonctionnement grammatical.

7. Les mots simples

Un mot simple est un mot qui ne possède pas de morphèmes ajoutés (préfixes, suffixes ou infixes) pour former un nouveau mot. Il s'agit donc d'une unité lexicale qui ne peut pas être décomposée en parties plus petites ayant un sens ou une fonction distincte. *« Quand un mot est formé d'un seul morphème, il s'agit, d'après une terminologie assez courante, d'un mot simple ou (mot monomorphématique) ».* (NIKLAS-SALMIEN Aino, 1997, P : 20).

Les mots simples sont des unités lexicales indivisibles, ils concernent les unités lexicales qui ne peuvent pas être segmenter en unités minimales douées de sens.

8. Mots communs et noms propres

Les mots communs (noms communs), sont considérés comme des bases essentiels à des fins de notre communication quotidienne. Ce sont des mots qui désignent une catégorie d'éléments de manière générale (des personnes, des lieux, des concepts, des objets, des idées, des actions), sans se référer à une entité particulière.

Selon LEHMANN Alise et MARTIN-BERTHET, les noms propres*, « à la différence des noms communs et des autres mots de la langue, ils n'ont pas véritablement de sens, mais seulement un référent, qui est une entité individuelle. »*

Elles ajoutent *« Certains noms propres entrent dans le lexique par leurs dérivés, qu'il s'agisse de noms de lieux (français, parisiens, savoyard) ou de noms d'auteurs (cornélien, rabelaisien, marxisme) et de personnage (gargantuesque, ubuesque). »* (2013, P : 25).

Dans notre travail de recherche nous nous intéressons spécifiquement aux anthroponymes, qui sont des noms propres de personnes, comme les noms et les prénoms. Ils servent à identifier et à distinguer les individus.

Nous estimons que les noms communs se démarquent des noms propres, ces derniers sont intrinsèquement dénués de sens. Les noms propres portent un référent individuel et spécifique.

Par ailleurs, nous pouvons former à partir de ces noms propres des dérivés (mots nouveaux) qui peuvent être utilisés pour décrire des caractéristiques, des idées ou des concepts associés à ces entités spécifiques. Par exemple, des dérivés de noms de lieux, d'auteurs ou de personnages peuvent être employés pour élargir l'utilisation des entités spécifiques à des contextes plus généraux.

9. Les mots dérivés

Un mot dérivé est un mot formé en ajoutant un préfixe et/ou un suffixe à un mot préexistant, en remplaçant un suffixe sur ce mot, ou encore en changeant sa forme pour créer un nouveau mot avec une signification différente. On appelle un dérivé le lexème construit par opération d'affixation.

9.1. La dérivation

Selon Alise LEHMANN et F. MARTIN-BERTHET, *« La dérivation produit un mot nouveau à partir d'un seul mot préexistant en le modifiant »* (2013, P : 139).

C'est un processus de formation de mots nouveaux, il permet la création de mots dérivés qui peuvent changer la catégorie grammaticale du mot d'origine, passant des noms aux verbes, des verbes aux adjectifs, des noms aux adjectifs, des adjectifs aux adverbes, etc. Ainsi, un mot dérivé est obtenu en combinant le sens de sa base lexicale avec celui des préfixes ou suffixes qui le composent, donnant naissance à un mot avec une signification spécifique.

Parmi les processus de dérivation ce qui nous intéresse est la dérivation affixale qui est selon Denis. APOTHELOZ *« une opération qui consiste à composer une base et un affixe (préfixe ou suffixe), elle est l'un des procédés majeurs de formation de mots en français, à côté de la composition et la dérivation non affixale »*. (2002, P : 73).

La préfixation : C'est un processus de dérivation où un préfixe (morphème lié) qui est ajouté à une base lexicale pour former un nouveau mot. Les préfixes sont des affixes qui sont placés avant la base, et ils peuvent modifier le sens du mot d'origine ou lui donner une nuance particulière.

La suffixation : Les suffixes sont des affixes qui sont placés après une base lexicale pour former un nouveau mot, ils peuvent modifier le sens d'un mot ainsi que sa catégorie grammaticale.

Les affixes dérivationnels, qu'ils soient des préfixes ou des suffixes, ont principalement une fonction sémantique dans la formation de nouveaux mots. Leur rôle est de créer une nouvelle unité lexicale à partir d'un mot déjà existant.

9.2. Les dérivés issus des noms propres

Nous considérons les mots communs dérivés de noms propres comme étant l'objet d'étude et la cible de notre logiciel, en raison de la fréquence de leur utilisation et leur importance dans la langue, nous avons créé ce prototype pour les interpréter. Ces dérivés de noms propres sont souvent utilisés dans divers domaines, notamment la technologie, la médecine, la littérature, la cuisine, l'ingénierie et la culture populaire. J. François-SABLAROLLES indique :

> « Parmi les dérivés, certains linguistes font un sort particulier à ceux dont la base est un nom propre. Des anthroponymes, toponymes ou noms de marque peuvent en effet servir de base à des dérivés pour dénommer des objets, notions, comportements, concepts, qui évoquent un trait contenu dans la réalité nommée par le nom propre. » (2000, P : 218)

Exemple : chiraquette, ouzbékisation, macroniste, marxisme

10. La créativité lexicale

Nous avons décidé de créer notre logiciel pour répondre à une évolution constante et rapide du lexique grâce à un processus nommé la créativité lexicale. Elle est le processus par lequel de nouveaux mots sont créés dans une langue. Cette créativité peut prendre différentes formes, telles que la création de néologismes (nouveaux mots), la dérivation (formation de mots à partir de racines existantes), la composition (combinaison de mots existants pour en former de nouveaux mots), etc.

Ce processus fait référence à la capacité d'utiliser le langage de manière inventive et originale en créant de nouveaux mots selon les circonstances des locuteurs, en jouant avec les mots existants ou en inventant des expressions nouvelles.

Cela peut inclure des néologismes, des jeux de mots, des métaphores, et d'autres formes d'expression linguistique qui élargissent ou modifient le sens des mots traditionnels.

D'après Chomsky, la créativité lexicale est *« l'une des caractéristiques fondamentales de la compétence linguistique qui permet à n'importe quel individu qui connaît sa langue d'expliquer un nombre illimité de pensées nouvelles adoptées à des situations nouvelles ».* (MOUNIN Gorges, 1974 P : 91).

Elle est considérée comme une compétence linguistique, c'est la capacité d'une personne à exprimer une variété infinie d'idées nouvelles dans divers contextes en utilisant sa langue avec précision et adaptabilité.

La créativité lexicale se définit comme étant l'ensemble des processus de formations de mots nouveaux dans la mesure où la langue n'est pas considérée comme une entité stable, mais qui évolue, mue et se transforme au fil du temps.

Le processus de créativité lexicale est essentiel pour revitaliser et enrichir une langue. Il consiste à créer de nouveaux mots, expressions et structures linguistiques en réponse aux changements sociaux, culturels et technologiques.

Cette innovation linguistique permet à une langue de rester pertinente et adaptable, tout en offrant aux locuteurs des moyens d'expression plus riches et précis pour communiquer.

Autrement dit, les langues naturelles évoluent en permanence pour répondre aux besoins changeants de la société. La créativité lexicale fait référence à la capacité de ces langues à créer de nouveaux mots et expressions pour s'adapter à l'évolution de la société. En gérant cette adaptation, la créativité lexicale contribue à l'enrichissement continu du lexique d'une communauté linguistique en fonction de son histoire et de ses besoins.

11. La néologie

La néologie est un phénomène naturel et constant dans l'évolution d'une langue donnée, elle permet à une langue de rester vivante adaptable et pertinente face aux défis du monde moderne.

Selon Marie - Françoise MORTUREUX, la néologie consiste en l'ensemble des procédés permettant de créer de nouveaux mots dans une langue donnée.

Ces procédés peuvent inclure la formation de mots par composition, dérivation, emprunt à d'autres langues, ou même par création pure et simple.

L'objectif de la néologie est souvent de combler des lacunes lexicales ou d'exprimer de nouvelles idées, concepts ou réalités émergentes. *« L'ensemble de procédés de formation de mot nouveaux »* (MORTUREUX M.F, 2008, P :137)

Dans cette perspective la néologie implique la création de nouveaux mots ou unités lexicales. Ces créations ne sont pas arbitraires, mais plutôt guidées par des règles de production qui sont intégrées dans le système lexical d'une langue donnée.

En d'autres termes, les nouveaux termes doivent être conformes aux structures linguistiques et aux normes d'utilisation de la langue pour être considérés comme des néologismes.

Cela peut inclure des processus tels que la dérivation, la composition ou d'autres formes de création lexicale qui sont cohérentes avec les règles et les conventions linguistiques de la langue en question.

La néologie peut se manifester de plusieurs façons, mais nous ce qui nous intéresse est la néologie de forme. Elle est aussi appelée néologie morphologique**.** Selon J. DUBOIS, « la néologie de forme consiste à fabriquer de nouvelles unités » alors que « la néologie de sens consiste à employer un signifiant existant déjà dans la langue considérée en lui conférant un contenu qu'il n'avait pas jusqu' alors- que ce contenu soit conceptuellement nouveau ou qu'il ait été jusque-là exprimé par un autre signifiant » (2012, P : 322).

En prenant en compte les définitions de J. DUBOIS, on constate que la néologie de forme se réfère au processus qui implique la génération de nouvelles unités linguistique, en se servant des règles morphologiques d'une langue. Tandis que la néologie de sens, est le fait d'attribuer à un signifiant existant dans la langue une nouvelle signification.

En effet la création de nouveaux mots par néologie de forme se manifeste par l'exploitation des procédés morphologiques comme l'affixation (suffixation et préfixation).

11.1. Les néologismes

Selon Aino- NIKLAS-SALMIEN, *« le néologisme peut être défini comme une unité nouvelle, de nature lexicale, dans un code linguistique défini. »* (1997, P : 140). Le néologisme est une unité lexicale nouvellement créée dans une langue donnée, souvent pour nommer de nouvelles idées, concepts ou phénomènes.

D'après ces définitions nous concluons que les néologismes sont des nouveaux termes qui se présentent d'abord dans les discours quotidiens des interlocuteurs avant de se lexicalisés dans les dictionnaires.

Quant à la néologie est l'ensemble des processus de création et de formations de nouveaux mots, ce processus peut se produire par différents procédés comme (la dérivation, la composition, la siglaison, la troncation, l'emprunt, etc.).

En d'autres mots, le néologisme est le produit et le résultat de la néologie. Cette dernière est considérée comme étant l'un des procédés qui stimulent l'enrichissement du lexique d'une langue. Cela démontre la capacité des langues à s'adapter et à répondre aux besoins changeants de la société et de la culture.

Selon Aino NIKLAS-SALMIEN, on distingue trois types de néologismes, le néologisme formel, le néologisme de sémantique et l'emprunt. Dans notre recherche nous nous intéresserons seulement aux néologismes formels.

Le néologisme formel désigne un mot nouveau introduit dans la langue, ce dernier est créé à partir des règles de formation propres à la langue. (1997, P : 140)

Dans ce chapitre nous avons survoler les grandes lignes des connaissances actuelles qui portent sur notre sujet de recherche, nous avons scruté les différentes disciplines impliquées dans le domaine du traitement automatique du langage. Nous avons commencé par la présentation de la morphologie computationnelle, les ressources lexicales, les ontologies. Celles-ci représentent les bases théoriques et méthodologiques de ce domaine interdisciplinaires, chacune d'elles apportent des contributions pour aborder la problématique des dérivés de noms propres.

Ensuite, nous avons discuté de l'importance d'utiliser des approches spécialisées dans le traitement des mots dérivés, en adoptant ces méthodes nous pouvons assurer une bonne compréhension dans divers contextes. Comment l'intégration des modèles d'apprentissage automatique dans le domaine du traitement automatique des mots dérivés, permettent une compréhension plus précise des relations lexicales, cela nous mènera à des résultats plus précis.

Pour ce qui est de la linguistique dans ce chapitre nous avons réunis les fondements théoriques nécessaire pour notre étude, nous avons également cité les travaux majeurs qui s'intéressent aux domaines de la lexicologie et de la morphologie. Ainsi que les définitions des concepts pertinents pour notre étude.

En effet ces disciplines nous offrent une compréhension approfondie sur la dynamique évolutive du langage. Enfin nous pouvons dire que ce chapitre représente une base solide pour l'orientation de nos futurs travaux et à l'avancement de notre projet.

Chapitre 02

Considérations méthodologiques et techniques

Avant d'entamer la définition des outils utilisés pour la création de ce logiciel nous expliquerons brièvement les procédés que nous utilisons pour l'aboutissement de notre projet. Car la conception d'un logiciel dédié à la définition des néologismes issus d'anthroponymes représente un angle rarement exploré dans le domaine de la linguistique computationnelle moderne.

Cette démarche ne se limite pas à la simple création d'un programme informatique, mais nécessite une méthodologie rigoureuse et structurée. En exposant notre approche méthodologique dans cette partie du mémoire, nous visons à clarifier et à guider à travers les étapes nécessaires pour comprendre et résoudre les défis spécifiques posés par ce domaine spécialisé. À travers une collecte méticuleuse des données linguistiques, une organisation systématique de celles-ci, et le choix réfléchi des outils de programmation.

1. Procédés et méthodes suivis

Dans le processus de création d'un logiciel de Traitement Automatique du Langage (TAL) destiné à donner des définitions à des néologismes issus d'anthroponymes. La première étape cruciale réside dans la collecte et l'analyse rigoureuse des données linguistiques.

Cela implique la réunion méticuleuse d'anthroponymes ainsi que des morphèmes dérivationnels et flexionnels pertinents. Il est aussi impératif de prêter une attention particulière aux cas particuliers qui pourraient engendrer des erreurs dans le programme, afin de garantir la fiabilité des résultats.

Une fois les données linguistiques réunies, la prochaine étape consiste à les organiser de manière systématique. Pour ce faire, nous créons deux fichiers distincts : l'un pour les anthroponymes et l'autre pour les morphèmes dérivationnels et flexionnels. Cette structuration facilitera le traitement des données lors de l'étape de programmation et permettra une gestion efficace des informations linguistiques.

Enfin, une fois les fichiers de données linguistiques établis, vient l'étape cruciale de la programmation du logiciel de TAL. Cette phase implique le choix des outils et des langages de programmation appropriés, ainsi que la conception et l'implémentation du programme lui-même.

1.1. Initialisation et découpage des données linguistiques

En intégrant les données linguistiques préalablement rassemblées dans le processus de développement, le logiciel pourra fournir des définitions précises et pertinentes pour les néologismes créés à partir d'anthroponymes.

Lorsque le programme démarre, il a besoin de se préparer pour manipuler des mots et des données stockées dans des fichiers. Pour ce faire, il commence par importer une fonction spécifique de Python qui sera utile plus tard.

Cette fonction permet de découper des chaînes de caractères en morceaux plus petits en fonction d'un motif donné.

Ensuite, le programme crée ses propres outils sur mesure, qui sont essentiellement des fonctions personnalisées. Ces fonctions sont là pour effectuer des tâches spécifiques dans le programme.

La première fonction, par exemple, permettra de créer un type particulier de structure de données appelée dictionnaire à partir des informations stockées dans un fichier. Ce dictionnaire associe des mots à leurs significations, ce qui facilitera plus tard la recherche de significations de mots spécifiques dans le programme.

En plus de cela, le programme crée également d'autres outils pour extraire des parties spécifiques d'un mot dérivé. Par exemple, il peut extraire le début ou la fin d'un mot en fonction d'une donnée.

Cette première partie du programme pose les bases nécessaires pour que le programme puisse comprendre et manipuler efficacement les mots et les informations contenues dans les fichiers.

Ces outils sur mesure seront utilisés tout au long du programme pour aider à comprendre les mots entrés par l'utilisateur et à fournir les réponses appropriées.

1.2. Définition des fonctions pour l'interaction avec les données

Après avoir préparé son environnement de travail et chargé les données nécessaires depuis les fichiers, le programme se prépare à interagir avec ces données en définissant plusieurs fonctions.

Ces fonctions sont essentiellement des outils spécialisés qui effectueront différentes tâches lorsque le programme sera en cours d'exécution. Elles sont conçues pour accomplir des actions spécifiques sur les données manipulées par le programme.

Chaque fonction est comme un instrument spécialisé conçu pour accomplir une tâche particulière. Par exemple, une fonction pourrait être conçue pour chercher des mots spécifiques dans une liste, une autre pour effectuer des calculs complexes, et ainsi de suite.

L'objectif de définir ces fonctions dès le début du programme est de rendre le code plus modulaire et plus facile à gérer. En les regroupant de cette manière, le programme peut être organisé de manière logique et chaque partie du code peut être réutilisée à différents endroits sans avoir à réécrire le même code plusieurs fois.

Cette partie du programme établit les outils nécessaires pour effectuer des actions spécifiques sur les données chargées dans le programme. Ces fonctions seront utilisées plus tard pour interagir avec les données et fournir les fonctionnalités nécessaires à l'utilisateur.

1.3. La manipulation et l'analyse des données linguistiques

Une fois que le programme a préparé son environnement de travail en important les fonctionnalités nécessaires et en chargeant les données à partir des fichiers, il se lance dans la création d'outils personnalisés pour manipuler ces données.

Ces outils sont comme des instruments spécialisés. Chaque outil a une fonction spécifique à remplir et est conçu pour accomplir une tâche précise lors de l'exécution du programme.

L'un de ces outils est conçu pour créer une structure de données particulière appelée dictionnaire à partir des informations stockées dans un fichier. Un dictionnaire est une sorte de liste spéciale où chaque élément est associé à une clé unique. Dans ce cas, les mots du fichier serviront de clés, et leurs significations ou d'autres informations associées serviront de valeurs.

Une autre paire d'outils est mise en place pour permettre au programme d'extraire des parties spécifiques d'un mot dérivé, son début ou sa fin ou les deux. Ces outils sont utiles lorsque le programme recherche des correspondances ou traite des mots entrés par l'utilisateur. Ils fonctionnent en segmentant différentes parties d'un mot, facilitant ainsi le traitement et l'analyse des données.

Cette partie du programme crée des outils personnalisés qui seront utilisés pour manipuler efficacement les mots et les données stockées dans les fichiers. Ces outils sont conçus pour rendre le programme plus flexible, modulaire et capable de gérer différents types de données et d'opérations de manière efficace.

1.4. Le raisonnement derrière l'utilisation de bases de données simplifiées

Il est à noter que lors de la conception de notre programme, nous avons envisagé d'intégrer une base de données plus vaste de noms, potentiellement téléchargée depuis une source telle que data.world. Une telle base de données aurait offert une plus grande diversité de noms et de variantes, ce qui aurait enrichi l'expérience utilisateur en permettant une recherche plus étendue et précise.

Cependant, pour des raisons de simplicité et de démonstration, nous avons décidé d'utiliser un fichier texte comme source de données pour notre programme. Bien que ce fichier texte ne contienne qu'une petite sélection de noms, il nous a permis de présenter efficacement le fonctionnement de notre programme sans ajouter de complexité inutile.

L'avantage d'utiliser un fichier texte est sa facilité d'utilisation et de manipulation. Il est simple à créer, à éditer et à partager, ce qui en fait un choix idéal pour nos besoins d'illustration.

De plus, cela permet aux utilisateurs de comprendre rapidement le fonctionnement de notre programme sans avoir à se familiariser avec des bases de données plus complexes.

Bien que nous ayons opté pour la simplicité dans ce cas, notre programme reste flexible et pourrait facilement être adapté pour intégrer une base de données plus robuste à l'avenir.

En utilisant des techniques similaires à celles présentées dans notre programme actuel, nous pourrions charger et interroger des données provenant de sources externes telles que data.world, offrant ainsi une expérience plus riche et plus variée aux utilisateurs.

Nous avons envisagé d'intégrer une API, Dans le développement de notre logiciel, pour tirer parti des capacités avancées d'une IA. Cette approche aurait permis d'automatiser l'analyse des noms propres dérivés et d'extraire automatiquement les caractéristiques des porteurs de ces noms.

Cependant, après examen approfondi, nous avons décidé de ne pas utiliser immédiatement cette approche pour notre première version. Nous avons préféré commencer par des exemples simples et manuellement rédigés pour illustrer clairement le fonctionnement de notre logiciel.

La raison principale de ce choix est la simplicité. En utilisant des exemples simples rédigés manuellement, nous pouvons fournir une compréhension claire et directe du fonctionnement de notre logiciel sans ajouter de complexité supplémentaire. Cela facilite également la compréhension pour nos utilisateurs, qui peuvent rapidement saisir comment utiliser notre logiciel sans avoir à comprendre les détails techniques d'une intégration d'API.

De plus, en commençant par des exemples simples, nous pouvons nous assurer que notre logiciel fonctionne correctement et efficacement avant d'explorer des options plus avancées telles que l'intégration d'une IA via une API. Cela nous permet de construire une base solide sur laquelle nous pourrons ensuite ajouter des fonctionnalités plus avancées et sophistiquées à mesure que notre logiciel évolue.

2. La définition des outils utilisés pour la création

Dans la partie qui suit du mémoire nous prenons le temps d'expliquer les bases de l'informatique afin de fournir les outils nécessaires à la compréhension des étapes et des processus que nous avons suivis pour créer notre logiciel. En comprenant les fondements de l'informatique, il sera plus facile de saisir les concepts et les techniques impliqués dans la conception et le développement de logiciels.

2.1. Définition des langages de programmation

Un langage de programmation est un ensemble de règles et de symboles utilisés pour écrire des instructions que l'ordinateur peut comprendre et exécuter. Ces instructions permettent de créer des programmes informatiques qui réalisent diverses tâches, telles que le traitement de données, le contrôle de périphériques, ou encore la résolution de problèmes complexes.

Les langages de programmation sont conçus pour être compréhensibles à la fois par les humains et par les machines. Ils offrent une syntaxe structurée et des fonctionnalités spécifiques qui permettent aux développeurs de décrire précisément les actions à effectuer.

Certains langages sont plus adaptés à certains types de tâches ou à certains paradigmes de programmation, comme la programmation impérative, la programmation orientée objet ou la programmation fonctionnelle.

En outre, les langages de programmation peuvent être de différents niveaux d'abstraction, ce qui signifie qu'ils offrent différents niveaux de détails sur la façon dont les instructions seront exécutées par l'ordinateur.

Certains langages de haut niveau, comme Python ou Java, fournissent une abstraction élevée, permettant aux développeurs de se concentrer sur la logique métier plutôt que sur les détails de l'implémentation.

D'autres langages, comme le langage machine ou le langage d'assemblage, sont de plus bas niveau et nécessitent une compréhension plus approfondie du fonctionnement interne de l'ordinateur.[4]

2.1.1. Les niveaux des langages de programmation

Les langages de programmation sont généralement classés en plusieurs niveaux en fonction de leur proximité avec le langage machine, qui est le langage compris directement par l'ordinateur. Voici une brève explication des principaux niveaux de langages de programmation :

- Langage machine Le langage machine est le langage binaire directement compris par l'ordinateur. Il est constitué de séquences de 0 et de 1 qui représentent les instructions et les données.
- Langage d'assemblage Aussi appelé langage assembleur, il est un peu plus compréhensible pour les humains que le langage machine, car il utilise des mnémoniques et des symboles pour représenter les instructions et les emplacements mémoire.
- Les langages, tels que le C et le C++, offrent un niveau d'abstraction supérieur au langage d'assemblage, mais restent proches du fonctionnement interne de l'ordinateur. Ils offrent un contrôle précis sur le matériel et sont souvent utilisés pour le développement système et la programmation de bas niveau.
- Langages de haut niveau Les langages de haut niveau, comme Python, Java et JavaScript, offrent un plus haut niveau d'abstraction, ce qui signifie qu'ils permettent aux développeurs de s'exprimer de manière plus proche du langage humain.

[4]David Liu, Principles of Programming Languages, p7-11

Ils fournissent généralement des fonctionnalités et des structures de contrôle plus avancées, ainsi qu'une meilleure portabilité entre les différentes plateformes.[5]

2.2. Le Python, définition et raisons d'utilisation

Python est un langage de programmation polyvalent de haut niveau, souvent utilisé pour le développement rapide d'applications et pour le prototypage de logiciels. Voici quelques points clés à connaître sur Python :

- Syntaxe claire et lisible Python est réputé pour sa syntaxe claire et lisible, ce qui en fait un langage très accessible, même pour les débutants. Les blocs de code sont délimités par l'indentation, ce qui rend le code Python particulièrement lisible.
- Python est un langage polyvalent qui peut être utilisé dans de nombreux domaines, notamment le développement web, l'analyse de données, l'intelligence artificielle, l'automatisation de tâches, et bien plus encore.
- Python possède une vaste bibliothèque standard qui offre un large éventail de fonctionnalités prêtes à l'emploi, permettant aux développeurs d'écrire des programmes complexes avec un minimum d'effort.
- Python est un langage dynamiquement typé, ce qui signifie que le type des variables est déterminé à l'exécution et peut changer au cours de l'exécution du programme. Cela offre une grande flexibilité mais nécessite une certaine prudence lors de la programmation pour éviter les erreurs de type.
- Python bénéficie d'une communauté de développeurs très active et engagée, ce qui se traduit par une abondance de ressources en ligne, de bibliothèques tierces et de frameworks qui facilitent le développement dans divers domaines.

Python est un langage de programmation populaire et puissant, apprécié pour sa simplicité, sa polyvalence et sa large adoption dans de nombreux secteurs de l'informatique.

[5] https://online.maryville.edu/online-bachelors-degrees/computer-science/careers/categories-of-programming-languages/

2.2.1. Les fonctions dans le Python

En Python, les fonctions sont des blocs de code réutilisables qui effectuent une tâche spécifique lorsqu'elles sont appelées. Elles permettent de diviser un programme en morceaux plus petits et plus gérables, ce qui facilite la lecture, la maintenance et la réutilisation du code.

Les fonctions sont définies à l'aide du mot-clé "def" suivi du nom de la fonction et de ses paramètres entre parenthèses. Voici un exemple simple de fonction en Python :

{def addition(a, b):return a + b}

Dans cet exemple, la fonction "addition" prend deux paramètres, "a" et "b", et renvoie leur somme.[6]

2.2.2. Les modules dans le Python

Les modules en Python sont des fichiers contenant du code Python qui peuvent être importés dans d'autres programmes Python. Les modules permettent d'organiser le code en regroupant des fonctions, des classes et d'autres éléments connexes. Pour utiliser un module dans un programme Python, on utilise le mot-clé "import" suivi du nom du module. Voici un exemple d'importation d'un module appelé "math" qui contient des fonctions mathématiques :

{import math

print(math.sqrt(16)) (Ceci affiche la racine carrée de 16 à l'écran)}

Dans cet exemple, nous importons le module "math" à l'aide du mot-clé "import" et utilisons la fonction "sqrt" (square root) du module pour calculer la racine carrée de 16.

[6]https://pythonnumericalmethods.studentorg.berkeley.edu/notebooks/chapter03.01-Function-Basics

Les fonctions en Python sont des blocs de code réutilisables qui effectuent une tâche spécifique, tandis que les modules sont des fichiers contenant du code Python qui peuvent être importés dans d'autres programmes.

Le mot-clé "import" est utilisé pour importer des modules et accéder à leurs fonctionnalités depuis un programme Python.

Les variables : En programmation, une variable est un conteneur utilisé pour stocker des données pouvant être modifiées ou manipulées dans un programme. Chaque variable a un nom unique qui la distingue des autres, ainsi qu'une valeur associée qui peut être de différents types tels que des entiers, des nombres à virgule flottante, des chaînes de caractères, des listes, etc.

La déclaration d'une variable consiste à créer une variable et à lui attribuer une valeur. En Python, la déclaration d'une variable se fait simplement en utilisant le nom de la variable suivi du symbole "=" (qui veut dire "recoit" et non pas "est égale a") et de la valeur que vous souhaitez lui attribuer. Voici un exemple simple de déclaration de variables en Python :

{x = 5 (Déclaration d'une variable x contenant la valeur 5)

nom = "Alice" (Déclaration d'une variable nom contenant la chaîne de caractères "Alice")}

Dans cet exemple, nous avons déclaré deux variables : "x" et "nom". La variable "x" contient la valeur 5, tandis que la variable "nom" contient la chaîne de caractères "Alice".

Il est important de noter que Python est un langage dynamiquement typé, ce qui signifie que vous n'avez pas besoin de spécifier le type de données d'une variable lors de sa déclaration. Le type de données est déterminé automatiquement en fonction de la valeur attribuée à la variable. De plus, les variables en Python peuvent être réaffectées avec une nouvelle valeur à tout moment. Par exemple :

{x = 5 (x contient la valeur 5)

x = 10 (x contient maintenant la valeur 10)}

Dans cet exemple, nous avons réaffecté la variable "x" avec une nouvelle valeur (10), remplaçant ainsi l'ancienne valeur (5).

Une variable en Python est un conteneur utilisé pour stocker des données, et sa déclaration consiste à lui attribuer une valeur en utilisant son nom suivi du symbole "=". Les variables peuvent être de différents types et peuvent être réaffectées avec de nouvelles valeurs à tout moment.[7]

2.3. Les paramètres de fonctions en programmation

En programmation, les paramètres sont des variables utilisées pour passer des données à une fonction lors de son appel. Ils permettent à une fonction d'accepter des valeurs externes qui peuvent être utilisées à l'intérieur de la fonction pour effectuer des opérations ou des calculs spécifiques.

Lors de la définition d'une fonction, vous pouvez spécifier les paramètres qu'elle acceptera en plaçant leurs noms entre parenthèses après le nom de la fonction.

Ces paramètres agissent comme des variables locales à l'intérieur de la fonction, et leur valeur est déterminée lors de l'appel de la fonction. Voici un exemple simple de définition d'une fonction avec des paramètres en Python :

{def addition (a, b): return a + b}

Dans cet exemple, la fonction "addition" accepte deux paramètres, "a" et "b". Lorsque la fonction est appelée avec des valeurs spécifiques pour ces paramètres, par exemple addition (3, 5), les valeurs passées en argument sont assignées aux

[7] https://sites.pitt.edu/~naraehan/python3/mbb20

paramètres correspondants à l'intérieur de la fonction. Dans ce cas, "a" sera égal à 3 et "b" sera égal à 5, et la fonction renverra leur somme, soit 8.

Les paramètres permettent de rendre les fonctions plus flexibles et réutilisables, car ils permettent à une fonction d'accepter différentes valeurs en fonction des besoins du programme. De plus, les paramètres peuvent avoir des valeurs par défaut, ce qui signifie qu'ils peuvent être omis lors de l'appel de la fonction si une valeur par défaut a été spécifiée dans la définition de la fonction.

Les paramètres sont des variables utilisées pour passer des données à une fonction lors de son appel. Ils permettent à une fonction d'accepter des valeurs externes qui peuvent être utilisées à l'intérieur de la fonction pour effectuer des opérations spécifiques.[8]

2.4. L'appels aux fonctions dans le Python

L'appel à une fonction en Python consiste à utiliser le nom de la fonction suivie de parenthèses contenant les arguments nécessaires (le cas échéant). Voici un exemple simple d'appel à une fonction en utilisant la fonction "addition" que nous avons définie précédemment :

```
{resultat = addition(3, 5)

print(resultat)  ( Affiche 8 )}
```

Dans cet exemple, nous appelons la fonction "addition" en lui passant deux arguments : 3 et 5. La fonction additionne ces deux nombres et renvoie le résultat, qui est stocké dans la variable "resultat". Ensuite, nous imprimons ce résultat à l'aide de la fonction "print".

[8] https://textbooks.cs.ksu.edu/intro-python/06-functions/04-param-arg/
https://cs.du.edu/~intropython/intro-to-programming/function_parameters

Il est important de noter que les paramètres passés à une fonction peuvent être de différents types : des nombres, des chaînes de caractères, des listes, des dictionnaires, voire même d'autres fonctions. Les fonctions peuvent également renvoyer des valeurs en utilisant le mot-clé "return". Dans notre exemple précédent, la fonction "addition" renvoie la somme des deux nombres passés en argument.Il est également possible de définir des fonctions sans paramètres.

Par exemple :

{def bonjour():

print("Bonjour !")

bonjour () (Appel de la fonction bonjour())}

Dans cet exemple, la fonction "bonjour" ne prend aucun paramètre, mais elle affiche simplement le message "Bonjour !" lorsqu'elle est appelée.

L'appel à une fonction en Python consiste à utiliser son nom suivi de parenthèses contenant les arguments nécessaires (s'il y en a). La fonction est ensuite exécutée avec ces arguments, et éventuellement renvoie une valeur grâce au mot-clé "return".[9]

3. Le code

3.1. Définition des deux datasets utilisés dans le programme

Un dataset, en science des données, désigne simplement une collection organisée de données. Ces données peuvent provenir de différentes sources comme des expériences, des enquêtes, des simulations ou encore des capteurs.

L'objectif principal d'un dataset est de rendre les données facilement accessibles et utilisables pour des analyses ou pour former des modèles d'apprentissage automatique.[10]

[9] https://textbooks.cs.ksu.edu/intro-python/06-functions/02-basics/
https://cs.stanford.edu/people/nick/py/python-function
[10] https://pll.harvard.edu/course/introduction-data-science-python

Dans ce programme, nos données sont dans deux fichiers .txt (fichier text). Les deux jeux de données utilisés dans ce programme servent à enrichir l'analyse morphologique des mots et à leur attribuer des significations contextuelles. Voici une explication détaillée de chaque jeu de données :

3.1.1. Le dataset morphlex.txt

Ce fichier contient une liste de morphèmes dérivationnels ainsi que leurs explications, avec chaque paire clé-valeur séparée par deux points ":".

Par exemple :

"eur:fonction de"

"iste:qui est adepte d'une ideologie ou theorie de"

Ce jeu de données fournit des informations sur la signification des morphèmes dérivationnels. Les morphèmes dérivationnels sont des unités linguistiques qui modifient le sens ou la fonction d'un mot lorsqu'ils sont ajoutés à celui-ci.

Par exemple, dans le mot "antimacroniste", le morphème dérivationnel "anti-" indique une opposition ou un rejet, tandis que le morphème dérivationnel "-iste" indique l'adhésion à une idéologie ou une théorie. En associant ces morphèmes à leurs explications, le programme peut déduire le sens global des mots composés.

3.1.2. Le dataset noms.txt

Ce fichier contient une liste de noms de personnages politiques ainsi que leurs caractéristiques principales, avec chaque paire clé-valeur séparée par deux points ":". Par exemple :

"ghandi:pacifiste"

"churchill:stratège"

Ce jeu de données associe des noms propres à leurs caractéristiques principales. Cela permet d'identifier les noms propres dans un texte et de fournir des informations supplémentaires sur ces noms, ce qui peut être utile pour comprendre le contexte dans lequel ils sont utilisés.

Par exemple, dans le mot "antimacroniste", le programme peut détecter le nom propre "macron" et lui attribuer la caractéristique "président français", ce qui contribue à comprendre le sens global du mot composé.

Ces deux jeux de données fournissent des informations complémentaires qui permettent au programme d'analyser morphologiquement les mots et de leur attribuer des significations contextuelles plus riches. Ils sont essentiels pour comprendre et interpréter le sens des mots composés dans un texte.

3.2. Le principe du Python

Le code effectue une analyse morphologique des mots présents dans un texte en détectant les morphèmes dérivationnels qui composent les noms propres et en les reliant à leur signification grâce aux dictionnaires de morphèmes dérivationnels et de noms propres avec leurs caractéristiques principales.

Voici une explication détaillée du fonctionnement du code :

- Création des dictionnaires à partir des fichiers texte : La fonction create_dictionary_from_file(filename) crée un dictionnaire à partir d'un fichier texte en séparant chaque ligne du fichier en deux parties (clé et valeur) sur la base d'un séparateur spécifique (dans ce cas, le caractère ":"). Ces dictionnaires contiennent les morphèmes dérivationnels et leurs explications, ainsi que les noms de personnages politiques et leurs caractéristiques principales.
- Création d'une liste de mots à partir du texte : La fonction create_list_from_text (filename) sépare tous les mots d'un texte et les place dans une liste.
- Détection des noms propres dans le texte : Le code parcourt chaque mot du texte et vérifie s'il correspond à un nom propre présent dans la liste de noms propres.

- Extraction des préfixes et suffixes : Si le mot contient un nom propre mais n'est pas égal à ce nom propre (ce qui signifie qu'il est composé d'un préfixe, d'un suffixe ou des deux), le code extrait le préfixe et le suffixe à l'aide des fonctions extract_prefix et extract_suffix.
- Affichage des morphèmes dérivationnels et de leurs explications : Si un préfixe ou un suffixe est détecté et correspond à un morphème dérivationnel présent dans le dictionnaire, le code affiche le morphème dérivationnel et son explication, ainsi que le nom propre associé et sa caractéristique principale.

Ce code identifie les morphèmes dérivationnels présents dans les noms propres d'un texte et les relie à leur signification à l'aide de dictionnaires contenant des informations sur les morphèmes dérivationnels et les noms propres.

4. Les futures éventualités du programme et fonctionnement de ses potentielles utilisations

Pour améliorer ce programme prototype afin de comprendre les tendances politiques populaires dans les commentaires sur les réseaux sociaux, nous pouvons intégrer des technologies d'intelligence artificielle (IA) pour analyser les sentiments des commentaires et des API pour récupérer les commentaires à partir des réseaux sociaux. Voici une explication détaillée de ces concepts et des méthodes que nous pouvons utiliser :

4.1. Une intelligence artificielle (IA)

L'intelligence artificielle (IA) est un domaine de l'informatique qui vise à créer des systèmes capables d'exécuter des tâches qui nécessitent normalement l'intelligence humaine. Ces systèmes sont souvent basés sur des algorithmes d'apprentissage automatique et de traitement du langage naturel. L'IA peut être utilisée pour analyser les données, prendre des décisions, reconnaître des schémas et bien plus encore.

4.2. Une API (Interface de programmation d'application)

Une API est un ensemble de règles et de protocoles qui permettent à des logiciels de communiquer entre eux. Les API sont utilisées pour permettre à des applications et à des services différents de partager des données et de fonctionner ensemble de manière transparente.

Dans ce contexte, nous pouvons utiliser des API pour récupérer des commentaires à partir des réseaux sociaux et pour effectuer l'analyse de sentiments.

5. La récupération des commentaires des réseaux sociaux

Pour récupérer des commentaires des réseaux sociaux, nous pouvons utiliser des API fournies par les plateformes de médias sociaux telles que Twitter, Facebook, Instagram, etc. Ces API permettent aux développeurs d'accéder aux données publiques des réseaux sociaux, y compris les commentaires, les messages, les tweets, etc.

En utilisant ces API, nous pouvons extraire les commentaires pertinents liés aux tendances politiques, sociales...

6. L'analyse des sentiments

L'analyse des sentiments est un processus permettant de déterminer l'attitude générale d'un texte, qu'elle soit positive, négative ou neutre. En utilisant des techniques de traitement du langage naturel (NLP) et d'apprentissage automatique, nous pouvons entraîner des modèles d'analyse des sentiments sur de grands ensembles de données annotées. Ces modèles peuvent ensuite être utilisés pour évaluer le sentiment des commentaires extraits des réseaux sociaux à l'égard de sujets spécifiques.

7. Les outils et bibliothèques utilisable en Python pour intégrer ces fonctions

Pour mettre en œuvre ces fonctionnalités dans Python, nous pouvons utiliser divers outils et bibliothèques :

- **Tweepy** est une bibliothèque Python pour interagir avec l'API Twitter et récupérer des tweets.
- **Facebook Graph API** est une API permettant d'accéder aux données publiques de Facebook, y compris les commentaires et les messages.
- **TextBlob** est une bibliothèque Python pour l'analyse des sentiments, qui fournit une interface simple pour effectuer des tâches de NLP telles que l'analyse de sentiments.
- **NLTK (Natural Language Toolkit)** est une bibliothèque Python très utilisée pour le traitement du langage naturel, comprenant des fonctionnalités telles que le tokenization, le stemming, la lemmatization, etc.

En combinant ces outils et technologies, nous pouvons améliorer le programme prototype pour comprendre les tendances politiques dans les commentaires sur les réseaux sociaux en récupérant les commentaires à l'aide d'API, puis en utilisant l'analyse des sentiments pour évaluer l'attitude des utilisateurs à l'égard des sujets politiques spécifiques.

8. Une parenthèse sur la potentielle future intégration d'un modèle d'apprentissage automatique

Apprentissage Automatique et TALN : Un tandem prometteur pour la dérivation de mots à partir de noms propres. Dans la quête de traiter la dérivation de mots à partir de noms propres, les approches fondées sur l'apprentissage automatique émergent comme des outils particulièrement puissants dans le domaine du Traitement Automatique du Langage Naturel (TALN).

Notamment, les modèles de langage pré-entraînés tels que BERT ont démontré des performances exceptionnelles dans diverses tâches linguistiques, ouvrant ainsi des perspectives prometteuses pour leur adaptation à notre problématique spécifique.

Ces modèles, par leur capacité à saisir de manière contextuelle les relations complexes au sein des phrases, s'avèrent adaptés à la compréhension et à la génération de mots dérivés liés aux noms propres.

Leur utilisation dans le cadre de notre projet pourrait représenter une avancée significative, offrant une approche plus sophistiquée pour appréhender les nuances morphologiques et sémantiques inhérentes à la dérivation de mots.

L'adaptation de ces modèles à notre objectif spécifique impliquerait un entraînement sur des corpus spécialisés, où les noms propres et leurs dérivés sont abondamment représentés.

Ce processus permettrait aux modèles de développer une sensibilité particulière aux schémas morphologiques liés à la dérivation dans notre domaine d'intérêt, améliorant ainsi leur capacité à générer des mots dérivés de manière précise.

L'intégration de ces modèles adaptés dans un cadre de traitement automatique dédié à la dérivation de mots à partir de noms propres serait une étape logique. Cette intégration pourrait contribuer significativement aux phases de désambiguïsation sémantique et de compréhension morphologique, fournissant ainsi des résultats plus précis et adaptés à notre contexte spécifique.

Un avantage notable de ces approches basées sur l'apprentissage automatique réside dans leur capacité à généraliser à partir de vastes ensembles de données. Cette propriété est particulièrement pertinente pour la dérivation de mots, où la variété de cas et de schémas morphologiques peut être apprise de manière exhaustive, conférant ainsi aux modèles une adaptabilité aux différentes subtilités du langage.

En conclusion, l'association de l'apprentissage automatique et du TALN émerge comme un tandem puissant dans notre quête de dérivation de mots à partir de noms propres. L'utilisation de modèles de langage pré-entraînés offre des opportunités d'exploration significatives, promettant des résultats précis et contextualisés dans notre domaine spécifique d'intérêt.

Cependant, il est crucial d'examiner attentivement plusieurs considérations quant à l’incorporation dans notre application. L'une des premières préoccupations concerne la complexité et la taille de ces modèles étant des architectures

sophistiquées, peuvent présenter des défis en termes de ressources matérielles et de temps de traitement.

La gestion de la mémoire et l'utilisation efficace de l'unité de traitement graphique (GPU) sont des aspects cruciaux, notamment si notre logiciel est destiné à fonctionner sur des plateformes avec des capacités matérielles limitées.

En outre, la question de l'adaptation des modèles à des domaines spécifiques se pose. Bien que ces modèles soient performants dans des tâches générales de traitement du langage naturel, leur capacité à saisir des nuances dans des domaines spécialisés peut nécessiter une adaptation supplémentaire.

En d'autres termes, les modèles pré-entraînés peuvent ne pas toujours être directement applicables à des contextes spécifiques sans une personnalisation appropriée.

L'importance des données d'entraînement est un autre aspect crucial. Les modèles sont formés sur des corpus de données volumineux, généralement englobant une gamme diversifiée de sujets.

Pour garantir une performance optimale dans des domaines spécifiques, il peut être nécessaire de compléter ces données d'entraînement avec des ensembles spécifiques à notre application.

Par ailleurs, le temps d'inférence, soit le temps nécessaire pour que le modèle génère des prédictions sur de nouvelles données, doit être pris en compte. Des modèles plus grands, bien que puissants, peuvent entraîner des temps d'inférence plus longs, ce qui peut affecter l'expérience utilisateur, en particulier dans des applications nécessitant des réponses en temps réel.

Il est essentiel de reconnaître que la performance des modèles pré-entraînés dépend étroitement de la qualité des données d'entraînement. Des biais peuvent être introduits si les données ne sont pas représentatives ou équilibrées, ce qui souligne l'importance de garantir la qualité et la diversité des données utilisées.

En somme, l'intégration de modèles de langage pré-entraînés dans notre logiciel peut être bénéfique, mais il est primordial de procéder avec une compréhension approfondie de ces modèles, de leurs besoins en ressources, et de leurs limites. Une évaluation réfléchie de ces considérations garantira une intégration réussie et une utilisation optimale des capacités offertes par ces modèles de pointe.

9. Le choix des noms à inclure

Dans le cadre de notre projet de développement d'un logiciel dédié au traitement automatique des mots communs dérivés à partir de noms propres, une réflexion approfondie sur le choix des noms à inclure dans notre modèle s'avère essentielle. Nous devons décider s'il est plus judicieux de nous concentrer sur les noms humains, tels que les noms de personnes, ou d'inclure également des noms de lieu et d'objets dans notre analyse sémantique.

En envisageant de prioriser les noms humains, nous reconnaissons certains avantages stratégiques. Tout d'abord, la fréquence d'utilisation des noms de personnes dans la création de mots dérivés est souvent plus élevée que celle des noms de lieu ou d'objets. Cela offre une ample source de données pour l'entraînement de notre modèle, renforçant ainsi sa capacité à reconnaître et générer des termes pertinents.

De plus, la pertinence dans certains contextes peut être un facteur déterminant. Si notre logiciel est destiné à des domaines où les noms de personnes jouent un rôle prépondérant, tels que la politique, la culture populaire ou d'autres secteurs centrés sur des figures humaines, il peut être logique de mettre l'accent sur ces noms pour une analyse plus ciblée.

Cependant, il est néanmoins intéressant d'explorer d'autres options et de décrire brièvement les considérations liées à l'inclusion des noms de lieu ou d'objets dans notre analyse sémantique.

Une option à considérer est d'élargir notre modèle pour inclure des noms de lieu. Cette approche pourrait être pertinente dans des domaines spécifiques où les références géographiques jouent un rôle significatif dans la création de termes

dérivés. Cela pourrait être le cas dans des secteurs tels que la géopolitique, le tourisme ou d'autres contextes où les lieux spécifiques sont étroitement liés à des concepts particuliers.

Une autre option à explorer serait l'inclusion des noms d'objets. Cela pourrait être pertinent dans des domaines tels que la technologie, la science ou l'industrie, où les noms d'objets peuvent être fréquemment utilisés pour créer des termes spécialisés.

Ces options supplémentaires soulèvent des questions cruciales. La spécificité du domaine dans lequel notre logiciel sera utilisé est un facteur déterminant. Si les noms de lieu ou d'objets sont incontournables dans ce contexte, les inclure peut-être essentiel pour garantir la pertinence de notre modèle.

La diversité reste également une considération importante. Si l'objectif est d'assurer une représentation exhaustive du langage, envisager d'inclure des noms de lieu ou d'objets peut enrichir la variété des termes dérivés que notre logiciel peut reconnaître et générer.

En résumé, bien que nous prenions la décision de prioriser les noms humains, la réflexion sur d'autres options telles que l'inclusion des noms de lieu ou d'objets demeure pertinente pour une future extension. La spécificité du domaine et la recherche d'une diversité équilibrée restent des facteurs clés dans la configuration finale de notre modèle de traitement automatique des mots dérivés.

10. L’acquisition d’une base de données de noms de famille

Dans notre projet de développement, le logiciel doit être capable de détecter et reconnaitre les noms, c’est pour cela que l'acquisition d'une base de données exhaustive de noms de famille est une étape cruciale. Nous avons opté pour une approche efficiente en téléchargeant un ensemble de données depuis data.world, une plateforme reconnue pour sa qualité et sa diversité de ressources.

Cette démarche présente plusieurs avantages significatifs qui contribueront à l'efficacité de notre logiciel de détection de noms de famille.

11. Data.world

Data.world est une plateforme innovante de collaboration et de partage de données, se positionne comme un acteur majeur dans l'écosystème de la gestion de données. Fondée sur la conviction que les données sont plus puissantes lorsqu'elles sont partagées et exploitées collectivement, cette plateforme offre un terrain fertile pour la collaboration, l'exploration et l'exploitation de données riches et variées.

Au cœur de la mission de data.world se trouve la volonté de créer un espace où les individus, les entreprises et les organisations peuvent collaborer de manière transparente autour des données.

La plateforme propose une interface mettant l'accent sur l'accessibilité des données et la facilité de collaboration, faisant ainsi tomber les barrières qui pourraient entraver la découverte et l'utilisation des informations.

La diversité des ensembles de données disponibles sur data.world est une caractéristique distinctive. Des données démographiques aux données météorologiques en passant par des jeux de données spécifiques à des industries, la plateforme offre une variété de ressources qui peuvent répondre aux besoins les plus divers.Cette diversité stimule l'innovation en permettant aux utilisateurs de croiser et d'analyser des données provenant de sources hétérogènes.

Data.world va au-delà de la simple mise à disposition de données en facilitant la collaboration entre les utilisateurs. La possibilité d'annoter, de discuter et de partager des insights directement au sein de la plateforme favorise une compréhension commune et une exploration collaborative des données. Cette dimension sociale de la plateforme crée un écosystème dynamique où les connaissances émergent de la mise en commun des perspectives individuelles.

Par ailleurs, la transparence et la traçabilité sont des valeurs fondamentales de data.world. Chaque ensemble de données est accompagné de métadonnées détaillées, permettant aux utilisateurs de comprendre l'origine, la qualité et l'utilisation

appropriée des données. Cette transparence renforce la confiance des utilisateurs et favorise une utilisation responsable des informations partagées.

En conclusion, data.world incarne l'émergence d'une communauté axée sur le partage et la collaboration autour des données. En offrant un espace où la diversité des connaissances peut converger, la plateforme se positionne comme un catalyseur pour des découvertes innovantes et des solutions collaboratives aux défis complexes de notre époque. Que ce soit pour les chercheurs, les entreprises ou les passionnés de données, data.world offre une plateforme dynamique pour explorer et exploiter le potentiel infini des informations partagées.

Notre décision d'utiliser cette source est, En premier lieu, le gain de temps qui est indéniable. Évitant ainsi la nécessité de collecter et de nettoyer les données manuellement, nous sommes en mesure de consacrer nos ressources à des aspects plus stratégiques du projet.

Cette approche préalablement organisée permet également de bénéficier d'une diversité de noms provenant de différentes sources, renforçant ainsi la robustesse de notre modèle de détection.

L'intégration de l'ensemble de données téléchargé dans notre logiciel s'avère simplifiée grâce à la structure bien définie et à la qualité documentaire de ces données. Cela garantit une intégration fluide et rapide dans notre système, facilitant ainsi le processus de développement.

Cependant, nous restons attentifs à quelques considérations essentielles. La représentativité de l'ensemble de données est un point focal, et nous nous assurons que la diversité des noms de famille correspond à la variété que notre application peut potentiellement rencontrer. La taille de l'ensemble de données est également scrutée pour s'assurer de sa pertinence par rapport à nos besoins spécifiques.

En ce qui concerne la licence d'utilisation, nous nous engageons à respecter toutes les conditions stipulées pour garantir une utilisation éthique des données

téléchargées. En parallèle, la mise à jour régulière de notre base de données est planifiée afin de tenir compte des évolutions potentielles des noms de famille au fil du temps.

Ainsi, l'utilisation d'un ensemble de données provenant de data.world s'inscrit comme une solution pragmatique, contribuant de manière significative à la réussite de notre projet de détection de noms de famille, tout en assurant la qualité et la pertinence des données que nous intégrons dans notre système.

Voilà qui conclut ce chapitre, où avons mis en lumière le domaine de l'informatique. Nous avons consacré la première partie de ce chapitre, a la présentation des bases fondamentales de l'informatique qu'on a utilisé pour la réalisation de notre prototype, nous avons détaillé les outils et les technologies implémentés, ainsi que la démarche qu'on a adopté dans le développement de notre application.

Dans cette seconde partie, nous avons exploré la possibilité d'intégrer un modèle d'apprentissage automatique dans notre logiciel, avec pour objectif de d'élargir la base de données des caractéristiques des noms propres et des morphèmes dérivationnels.

Nous avons également justifié notre décision de se concentrer exclusivement sur les anthroponymes et non sur tous les noms propres. Cette décision a été motivée principalement par leur fréquence d'utilisation plus élevée par rapport à d'autres types de noms propres.

De plus, nous avons détaillé notre approche pour acquérir une base de données de noms de famille, en utilisant des données téléchargées depuis data.world. Pour finir avec une présentation de cette plateforme, réputée pour sa vaste collection de jeux de données publics et collaboratifs, nous a permis d'accéder à une source fiable et riche en informations.

En conclusion, cette section a présenté en détail notre démarche technique ainsi que nos réflexions sur les développements futurs, mettant en lumière la robustesse et l'évolutivité de notre application.

Chapitre 03

Réalisation du programme

Dans ce troisième chapitre, nous explorerons en détail le fonctionnement de notre programme de dérivation morphologique, conçu pour analyser et fournir le sens des mots dérivés à partir de noms propres combinés avec des morphèmes dérivationnels, comme "darwinisme".

L'objectif principal de ce chapitre est de démontrer comment notre programme opère, en mettant en lumière ses mécanismes internes, ses capacités de gestion des variations et des exceptions, et en illustrant son utilisation à travers des exemples concrets.

Nous commencerons par une explication détaillée des schémas structurels qui sous-tendent notre programme, offrant ainsi une compréhension claire des processus impliqués. Ces schémas incluent les différentes étapes de traitement linguistique, de la reconnaissance des noms propres à la génération d'une définition.

Ensuite, nous présenterons plusieurs captures d'écran démontrant l'interface utilisateur et les résultats produits par le programme. Ces illustrations permettront de visualiser les différentes fonctionnalités du programme, y compris sa capacité à gérer les variations linguistiques et les exceptions aux règles de dérivation standard. Nous montrerons comment le programme s'adapte aux particularités des noms propres et des morphèmes dérivationnels pour fournir des définitions précises et pertinentes.

Enfin, nous discuterons des défis rencontrés lors du développement du programme et des solutions mises en œuvre ou qui pourront être mis en œuvre pour les surmonter. Cette section mettra en évidence les aspects innovants du programme et son potentiel pour des applications futures dans le domaine de la linguistique computationnelle.

En somme, ce chapitre vise à offrir une compréhension complète et pratique de notre programme de dérivation morphologique, en combinant des explications théoriques avec des démonstrations pratiques. Cela permettra de mieux apprécier la robustesse et la flexibilité de l'outil que nous avons développé, ainsi que son utilité pour les chercheurs et les étudiants en TAL.

1. L'explication et schématisation du code

1.1. Importation d'une fonction

```
from re import split  # Importe la fonction split du module re
```

La fonction split() en Python permet de découper une phrase ou un paragraphe en morceaux plus petits en utilisant un marqueur spécial appelé séparateur. Le module re quant à lui est comme un ensemble d'outils spéciaux pour travailler avec des mots et des phrases en utilisant des motifs spécifiques que vous définissez. Cela nous permet de rechercher, de filtrer et de manipuler du texte de manière très précise et puissante.

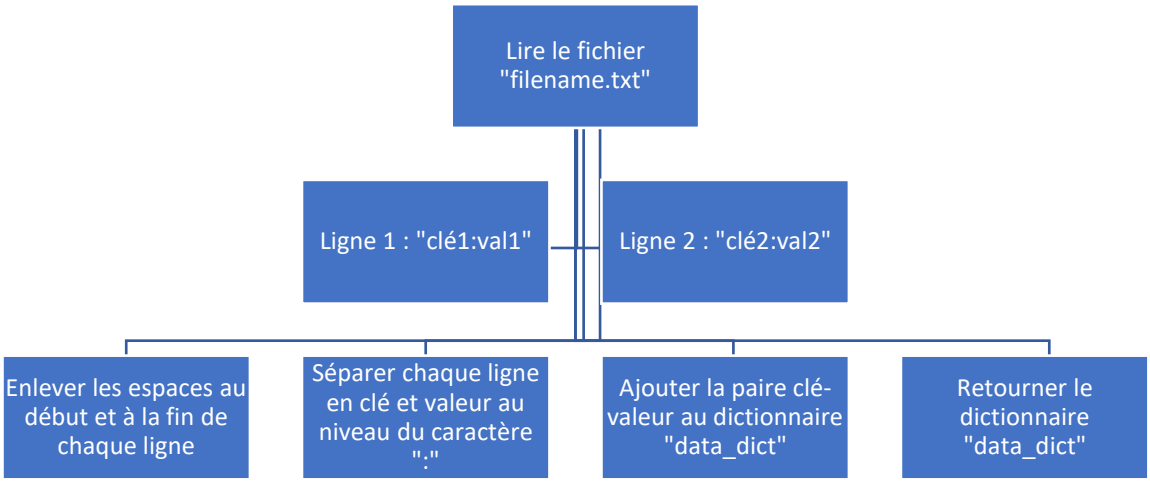

Figure 1: Schématisation d'importation d'une fonction

1.2. La création d'un dictionnaire à partir d'un fichier

Cette fonction effectue une action précise. Elle ouvre un fichier texte, puis lit son contenu ligne par ligne. Chaque ligne du fichier contient deux morceaux d'information séparés par le caractère " : ". Le premier morceau est appelé "clé" et le second "valeur".

La fonction prend chaque paire clé-valeur trouvée sur une ligne et les enregistre dans une sorte de liste organisée. Cette liste est appelée un "dictionnaire". Ainsi, après que la fonction a terminé son travail, elle a créé un dictionnaire qui contient toutes les associations clé-valeur présentes dans le fichier. Cela permet d'accéder rapidement à la valeur associée à chaque clé lorsque nécessaire.

```
defcreate_dictionary_from_file(filename):  # Définit une fonction pour créer un dictionnaire à partir d'un fichier
   data_dict= {}  # Initialise un dictionnaire vide

   withopen(filename, 'r') asfile:  # Ouvre le fichier en mode lecture
      for line infile:  # Parcourt chaque ligne du fichier
         line=line.strip()  # Enlève les espaces au début et à la fin de la ligne
         if line:  # Vérifie si la ligne n'est pas vide
            key, value =line.split(':')  # Sépare la ligne en clé et valeur au niveau du caractère ':'
            data_dict[key] = value  # Ajoute la paire clé-valeur dans le dictionnaire
   returndata_dict  # Retourne le dictionnaire rempli
```

1.3. L'extraction du préfixe

Cette fonction reçoit deux mots en entrée : dérivé et lexème. Ensuite, elle examine le premier mot, dérivé, pour trouver la partie qui est juste avant le deuxième mot, lexème. Une fois cette partie trouvée, la fonction la renvoie en tant que résultat. En résumé, cette fonction prend deux mots en entrée et identifie la portion du premier mot qui se trouve immédiatement avant le deuxième mot.

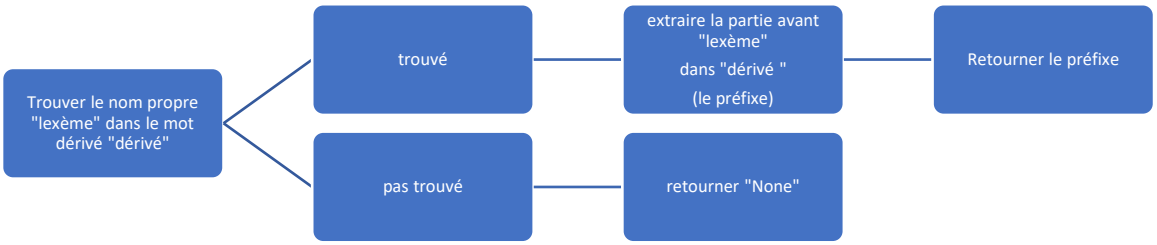

Figure 2: Schématisation d'extraction du préfixe

```
defextract_prefix(dérivé, lexème):  # Définit une fonction pour extraire le préfixe
    index=dérivé.find(lexème)  # Trouve l'index de la racine dans le mot plus grand
    if index !=-1:  # Vérifie si la racine a été trouvé
        prefix= dérivé[:index]  # Extrait le préfixe
        returnprefix  # Retourne le préfixe
    returnNone  # Retourne None si  la racine n'est pas trouvé
```

1.4. L'extraction du suffixe

La fonction "Extraire un suffixe" est presque exactement similaire à la fonction "Extraire un préfixe", mais avec une petite différence. Tout comme la fonction "Extraire un préfixe", elle prend deux mots en entrée : "dérivé" et "lexème", et examine le premier mot, "dérivé", pour identifier une partie spécifique. Cependant, au lieu de trouver la partie qui précède le deuxième mot, cette fonction cherche la partie qui vient juste après le deuxième mot. Ainsi, elle identifie la portion du premier mot qui se trouve immédiatement après le deuxième mot donné en entrée.

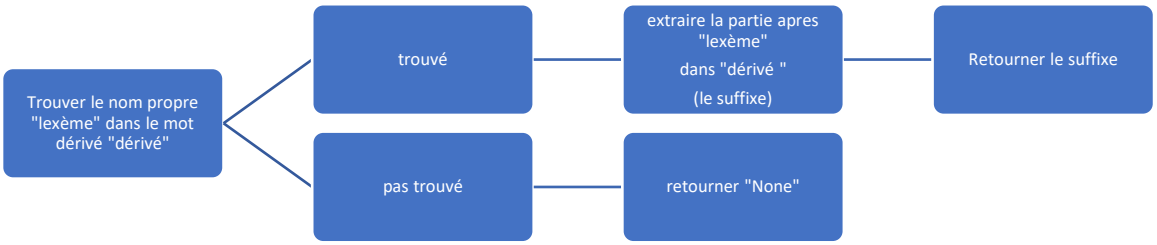

Figure 3: Schématisation d'extraction du suffixe

```
defextract_suffix(dérivé, lexème):  # Définit une fonction pour extraire le suffixe
    index=dérivé.find(lexème)  # Trouve l'index de la racine dans le mot plus grand
    if index !=-1:  # Vérifie si la racine a été trouvé
        suffix= dérivé[index +len(lexème):]  # Extrait le suffixe
        returnsuffix  # Retourne le suffixe
    returnNone  # Retourne None si la racine n'est pas trouvé
```

1.5. L'assignation des noms de fichiers

Lorsque nous parlons d'"assignation des noms de fichiers", nous désignons simplement le fait d'attribuer des noms spécifiques à des fichiers que nous voulons utiliser dans notre programme. Par exemple, si nous avons plusieurs fichiers sur notre ordinateur, chacun contenant différentes informations. Pour que notre programme sache quels fichiers utiliser, nous devons lui dire quels sont ces fichiers et quels noms ils portent.

Cela signifie que nous avons deux fichiers : "morphlex.txt" et "noms.txt". Nous avons donné à ces fichiers des noms spécifiques pour pouvoir les référencer facilement dans notre code.

Maintenant, chaque fois que nous voulons travailler avec le contenu de "morphlex.txt", nous pouvons simplement utiliser morphemeFlexd, et chaque fois que nous avons besoin du contenu de "noms.txt", nous pouvons utiliser names.

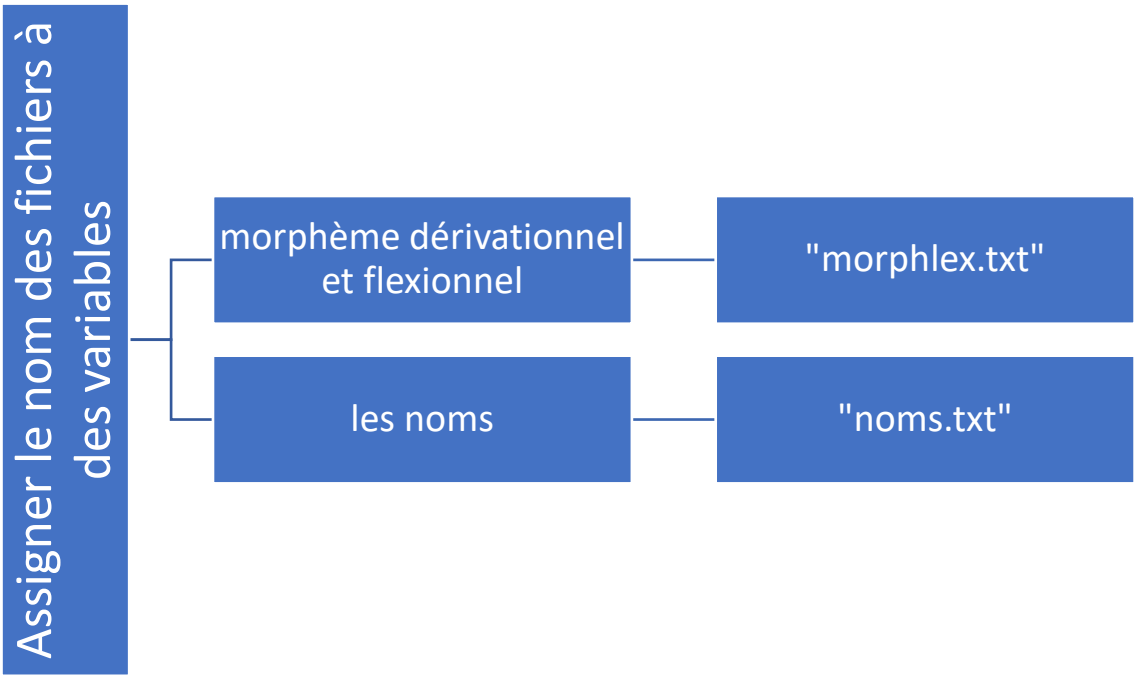

Figure 4: Schématisation de L'assignation des noms de fichiers

```
# Assignation des noms de fichiers à des variables
morphemeDerivationnel="morphlex.txt"
names ="noms.txt"
```

1.6. La création des dictionnaires et listes à partir des fichiers

Nous nous référons ici à la manipulation des données contenues dans des fichiers pour les charger dans des structures de données appropriées dans notre programme, telles que des dictionnaires et des listes. Prenons un exemple simple : supposons que nous ayons un fichier texte nommé "noms.txt" qui contient une liste de noms et de leurs correspondants.

Pour utiliser ces données dans notre programme Python, nous devons d'abord les lire depuis le fichier, puis les organiser de manière à ce que notre programme puisse les comprendre et les manipuler facilement.

```
# Création des dictionnaires et des listes à partir des fichiers

morphlexes_dict=create_dictionary_from_file(morphemeDerivationnel)

name_dict=create_dictionary_from_file(names)

fullnames_list=name_dict.keys()  # Obtient les clés du dictionnaire des noms
```

Cela signifie que nous avons deux fichiers : "morphlex.txt" et "noms.txt". Nous utilisons ensuite une fonction appelée create_dictionary_from_file() pour lire ces fichiers et créer des dictionnaires à partir de leurs contenus. Un dictionnaire est une structure de données qui associe des clés à des valeurs, ce qui est utile pour stocker des informations comme des paires clé-valeur (par exemple, un mot et sa signification).

Dans notre cas, nous créons un dictionnaire morphlexes_dict à partir du fichier "morphlex.txt", qui semble contenir des informations liées à des morphèmes.

Nous créons également un dictionnaire name_dict à partir du fichier "noms.txt", qui semble contenir des noms et leurs correspondants.

Enfin, nous utilisons name_dict.keys() pour obtenir les clés de ce dictionnaire (c'est-à-dire les noms) et les stocker dans une liste appelée fullnames_list. Cela nous permettra d'accéder facilement à tous les noms contenus dans le fichier "noms.txt".

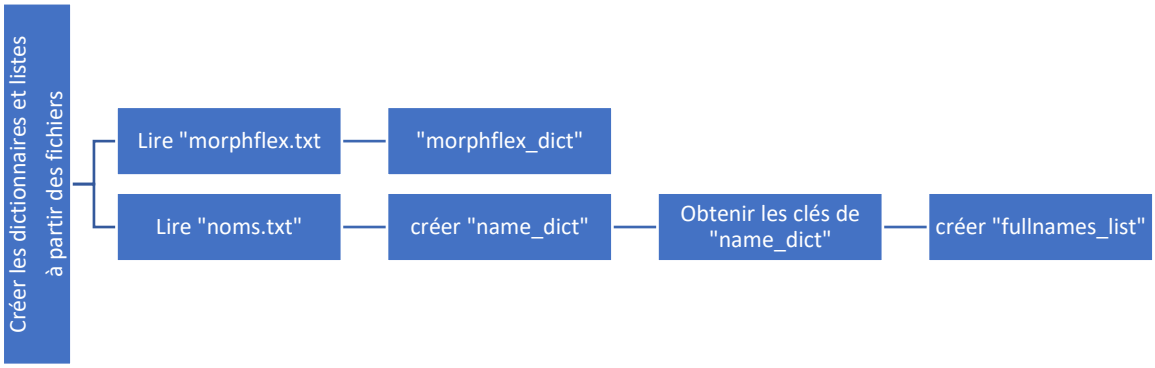

Figure 5: Schématisation de La création des dictionnaires et listes à partir des fichiers

1.7. Le traitement d'un mot d'entrée

Nous nous référons au processus de prendre un mot que l'utilisateur entre et d'effectuer diverses opérations dessus dans notre programme. Cela peut inclure la recherche de correspondances, la manipulation de chaînes de caractères, ou d'autres actions en fonction des besoins du programme.

Dans notre code, le processus de traitement d'un mot d'entrée se déroule comme suit :

- Récupération de la liste des noms à partir du dictionnaire name_dict.
- Initialisation d'une chaîne vide appelée string pour stocker la réponse.
- Parcours de chaque nom dans la liste des noms.
- Conversion du nom et du mot d'entrée en minuscules pour assurer une correspondance insensible à la casse.

- Vérification si le nom est contenu dans le mot d'entrée.
- Si le nom est trouvé dans le mot d'entrée :
  - Extraction du préfixe et du suffixe du mot d'entrée.
  - Vérification si le préfixe et le suffixe sont présents dans le dictionnaire morphlexes_dict. + Construction de la réponse en utilisant les informations trouvées dans les dictionnaires.
- Si le nom n'est pas trouvé dans le mot d'entrée, un message indiquant que le mot n'est pas encore dans la base de données est renvoyé.

En somme, le processus de traitement d'un mot d'entrée implique de comparer le mot d'entrée avec une liste de noms, d'extraire des informations en fonction des correspondances trouvées, et de construire une réponse basée sur ces informations. Cela permet à notre programme de répondre de manière appropriée aux mots d'entrée fournis par l'utilisateur.

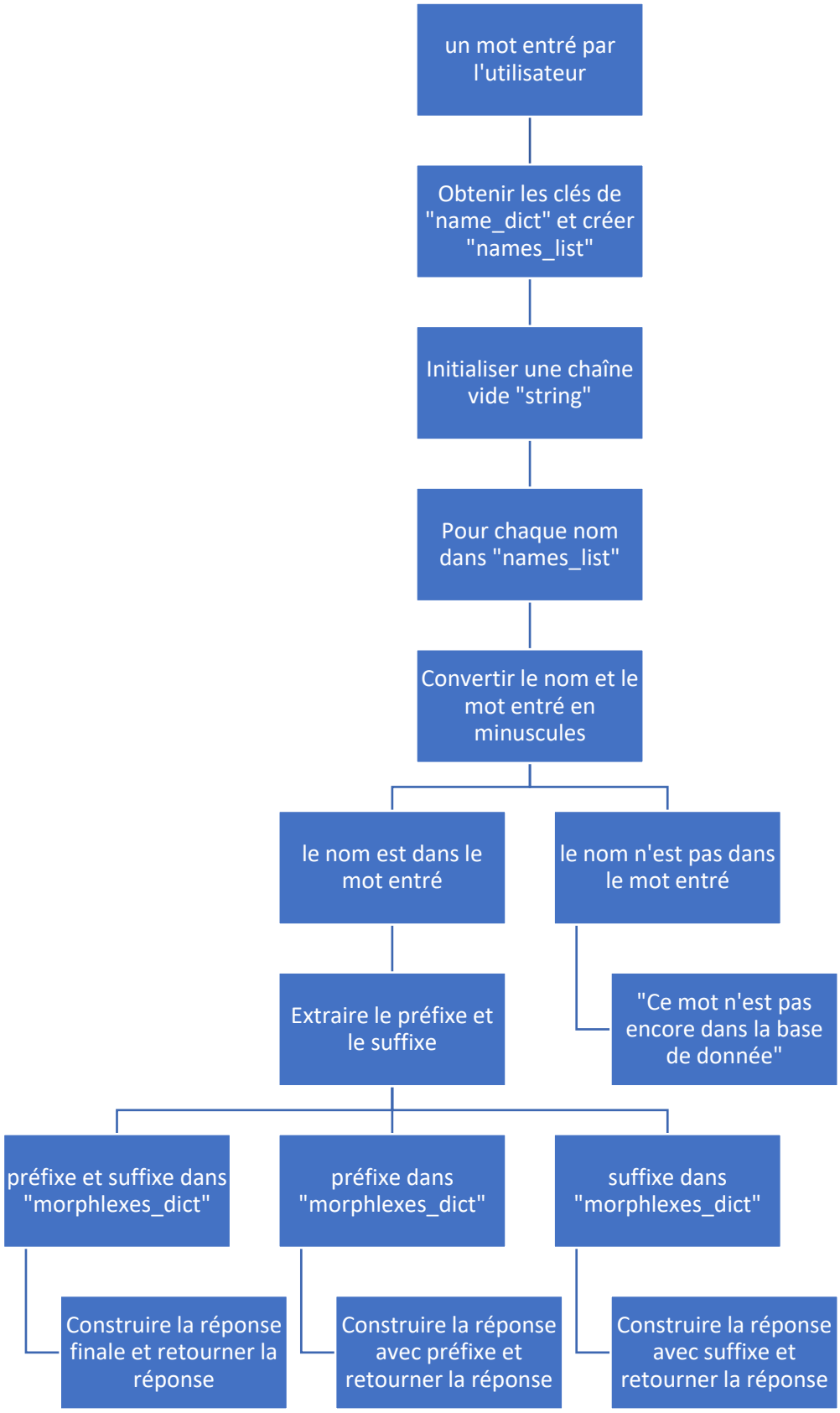

Figure 6: Schématisation du traitement d'un mot d'entrée

```
defprocess_input(input_word):  # Définit une fonction pour traiter un mot entré par l'utilisateur
    names_list=name_dict.keys()  # Obtient les clés du dictionnaire des noms
    string=""  # Initialise une chaîne vide pour construire la réponse
    fornameinnames_list:  # Parcourt tous les noms dans la liste
        lowername=name.lower()  # Convertit le nom en minuscules
        input_word_lower=input_word.lower()  # Convertit le mot entré en minuscules
        iflowernameininput_word_lower:  # Vérifie si le nom est dans le mot entré
            prefix =extract_prefix(input_word_lower, lowername)  # Extrait le préfixe
            suffix=extract_suffix(input_word_lower, lowername)  # Extrait le suffixe
            ifprefixandsuffixinmorphlexes_dict:  # Vérifie si le préfixe et le suffixe sont dans le dictionnaire
                string=f"{morphlexes_dict[prefix]}\n{morphlexes_dict[suffix]}{name}{name_dict[name]}"  # Construit la réponse
                return string  # Retourne la réponse
            elifprefixinmorphlexes_dict:  # Vérifie si le préfixe est dans le dictionnaire
                string=f"{morphlexes_dict[prefix]}"  # Construit la réponse
                return string  # Retourne la réponse
            elifsuffixinmorphlexes_dict:  # Vérifie si le suffixe est dans le dictionnaire
                        string=f"{morphlexes_dict[suffix]}{name}{name_dict[name]}"  # Construit la réponse
                return string  # Retourne la réponse
        else:
            string="ce mot n'est pas encore dans la base de donnée"  # Message si le mot n'est pas trouvé
    return string  # Retourne la réponse ou le message d'erreur
```

En somme, le processus de traitement d'un mot d'entrée implique de comparer le mot d'entrée avec une liste de noms, d'extraire des informations en fonction des correspondances trouvées, et de construire une réponse basée sur ces informations. Cela permet à notre programme de répondre de manière appropriée aux mots d'entrée fournis par l'utilisateur.

2. Exemples démontrant l'utilisation et flexibilité du programme

Voici quelques exemples d'utilisation du programme pour illustrer son fonctionnement plus clairement et plus concrètement.

Exemple 1 : Utilisation Simple - "macroniste"

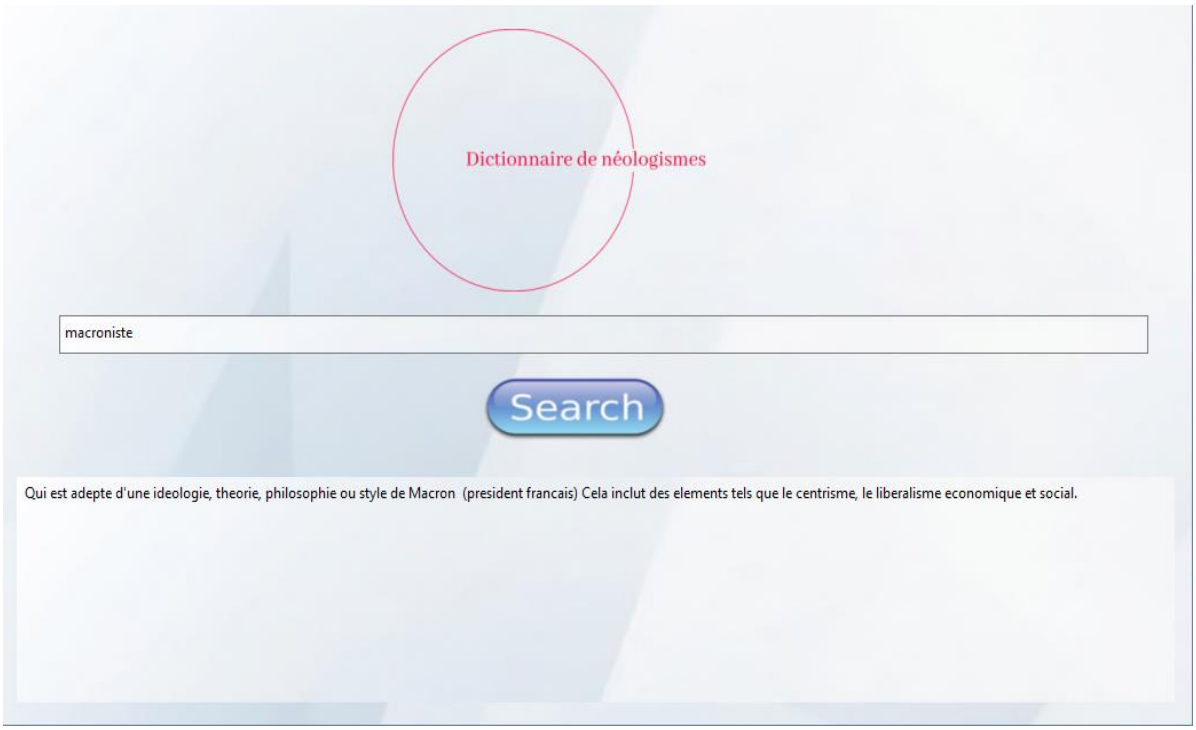

Figure 7: Exemple 1 : Utilisation Simple - "macroniste"

Dans cet exemple, nous voyons comment le programme interprète un mot dérivé d'un nom propre associé à un suffixe dérivationnel simple. En entrant "macroniste", le logiciel identifie "Macron" comme le nom propre et "-iste" comme le morphème dérivationnel signifiant "adepte ou partisan".

Le programme fournit alors la définition suivante : " Qui est adepte d'une idéologie, théorie, philosophie ou style de Macron (président français).

Cela inclut des éléments tels que le centrisme, le libéralisme économique et social.". Cette analyse simple montre l'efficacité du logiciel dans la reconnaissance et l'interprétation de mots dérivés de manière directe et précise.

Exemple 2 : Utilisation Simple - "darwinisme"

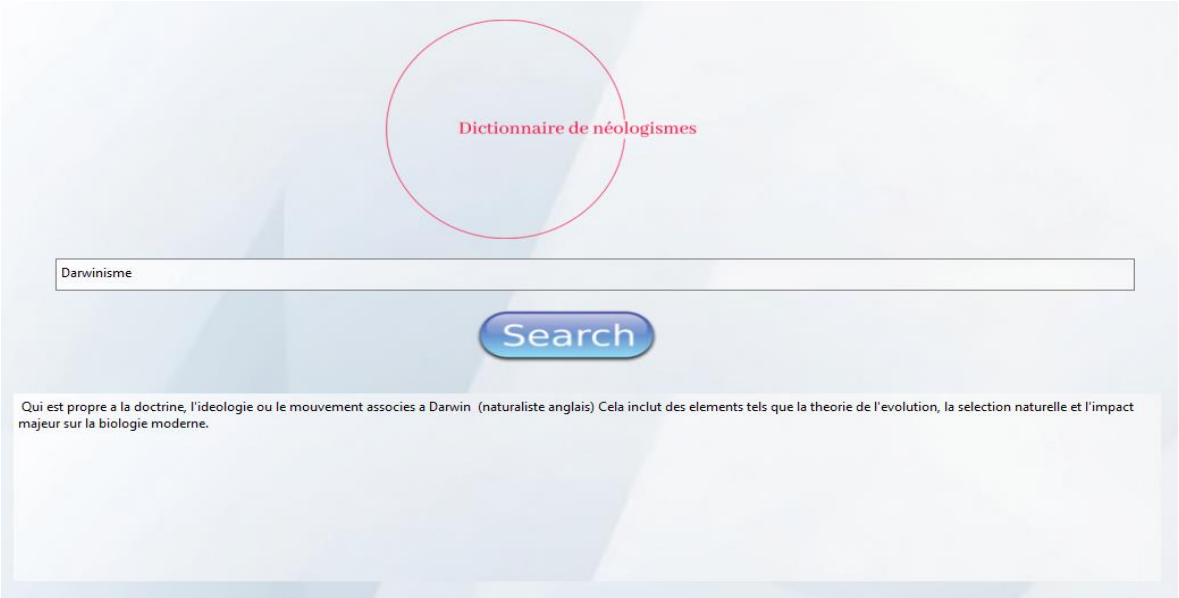

Figure 8: Exemple 2 : Utilisation Simple - "darwinisme"

Un autre exemple de l'utilisation simple de votre logiciel est illustré avec le mot "darwinisme". Ici, le programme détecte "Darwin" comme le nom propre et "-isme" comme le morphème dérivationnel signifiant "doctrine ou théorie".

Ainsi, le logiciel renvoie : " Qui est propre à la doctrine, l'idéologie ou le mouvement associes à Darwin (naturaliste anglais) Cela inclut des éléments tels que la théorie de l'évolution, la sélection naturelle et l'impact majeur sur la biologie moderne.". Cet exemple démontre la capacité du logiciel à traiter des termes dérivés de noms propres célèbres et à fournir des définitions claires et contextuelles.

Exemple 3 : Utilisation avec Dérivations Flexionnelles de Nombre - "marxiste/marxistes"

Figure 9: Exemple 3 : Utilisation avec Dérivations Flexionnelles de Nombre - "marxiste »

Figure 10: Exemple 3 : Utilisation avec Dérivations Flexionnelles de Nombre - " marxistes"

Notre programme peut également gérer les flexions de nombre, comme illustré avec "marxiste" et "marxistes". Pour "marxiste", le logiciel identifie "Marx" et le suffixe "-iste", fournissant la définition " Qui est adepte d'une idéologie, théorie, philosophie ou style de Marx (philosophe et économiste allemand) Cela inclut des éléments tels que le matérialisme historique, la critique du capitalisme et une vision sociopolitique radicale.

Lorsqu'on entre "marxistes", le programme reconnaît la forme plurielle et renvoie " Qui sont adeptes d'une idéologie, théorie, philosophie ou style de Marx (philosophe et économiste allemand). Cela inclut des éléments tels que le matérialisme historique, la critique du capitalisme et une vision sociopolitique radicale.". Cette fonctionnalité montre que le logiciel peut différencier entre les formes singulières et plurielles tout en maintenant la cohérence de la définition.

Exemple 4 : Utilisation avec Dérivations Flexionnelles de Genre et de Nombre - "napoléonien/napoléoniennes"

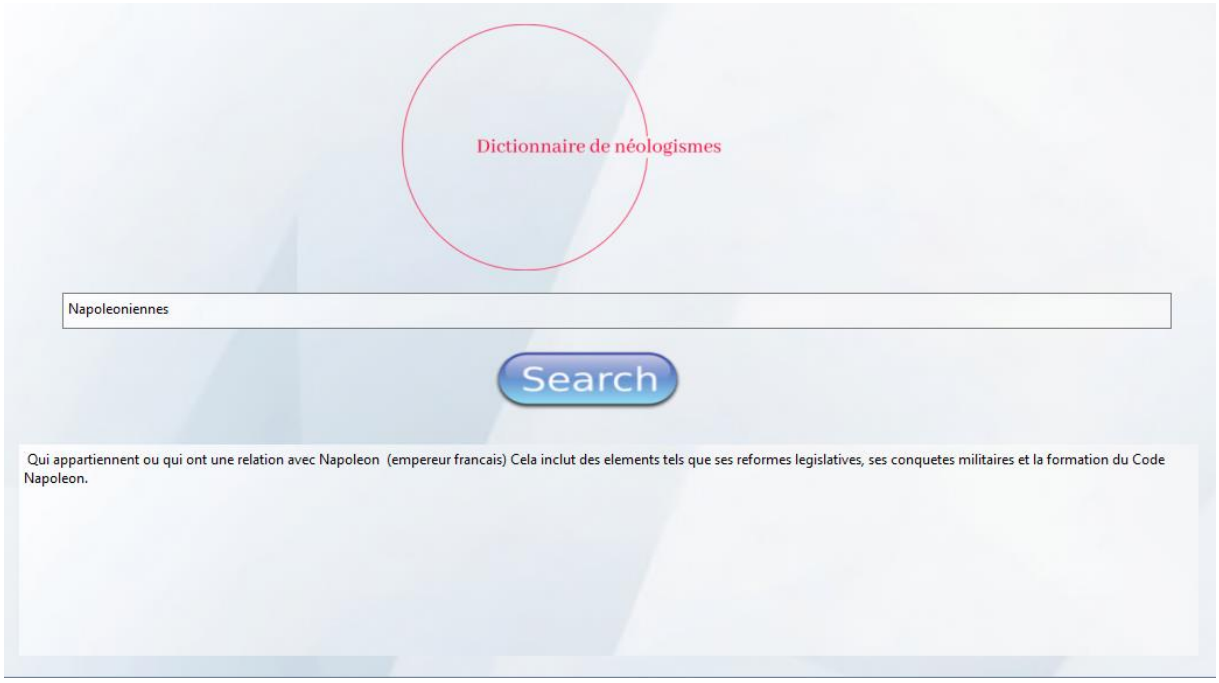

Figure 11: Exemple 4 : Utilisation avec Dérivations Flexionnelles de Genre et de Nombre - "napoléonien/napoléoniennes"

Le programme gère également les flexions de genre et de nombre, comme démontré avec "napoléonien" et "napoléoniennes". Pour "napoléonien", le logiciel identifie le nom propre "Napoléonien" et fournit la définition " Qui appartient ou qui a une relation avec Napoléon (empereur français) Cela inclut des éléments tels que ses réformes législatives, ses conquêtes militaires et la formation du Code Napoléon.

Lorsqu'on entre "napoléoniennes", le programme détecte la forme féminine et plurielle et renvoie " Qui appartiennent ou qui ont une relation avec Napoléon (empereur français) Cela inclut des éléments tels que ses réformes législatives, ses conquêtes militaires et la formation du Code Napoléon. ". Cet exemple illustre la capacité du logiciel à gérer des flexions morphologiques complexes, en adaptant la définition en conséquence.

Exemple 5 : Exception - "Trotski/trotskiste"

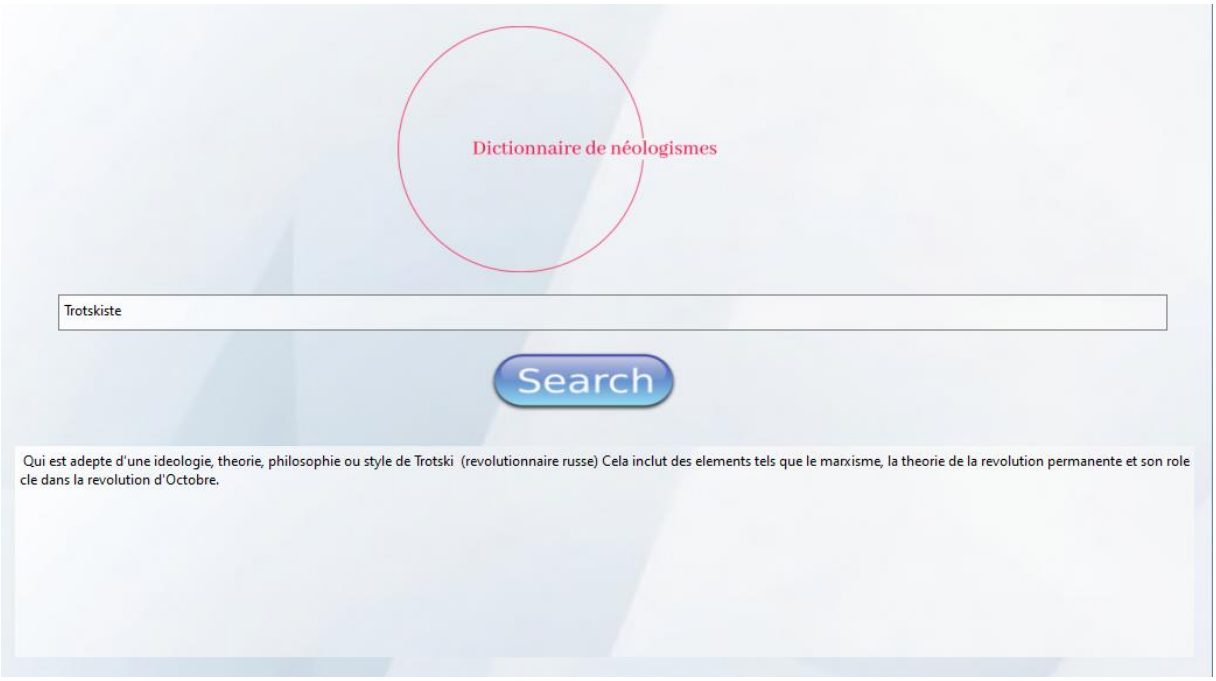

Figure 12: Exemple 5 : Exception - "Trotski/trotskiste"

Le logiciel est également capable de gérer des exceptions morphologiques. Par exemple, pour "Trotski" et "trotskiste", bien que l'ajout du suffixe "-iste" pourrait suggérer "trotskiiste", le programme connaît l'exception et fournit correctement "Trotskiste : Qui est adepte d'une idéologie, théorie, philosophie ou style de Trotski (révolutionnaire russe) Cela inclut des éléments tels que le marxisme, la théorie de la révolution permanente et son rôle clé dans la révolution d'Octobre.". Cela démontre l'intelligence du logiciel dans le traitement des règles exceptionnelles de dérivation, évitant les erreurs communes.

Exemple 6 : Exception - "hugo/hugolien"

Figure 13: Exemple 6 : Exception - "hugo/hugolien"

Le logiciel montre sa capacité à gérer des exceptions particulières avec des noms comme "Hugo" et "hugolien". Plutôt que de simplement ajouter un suffixe standard, le programme reconnaît la forme correcte et fournit la définition appropriée : "Hugolien : Qui appartient ou qui a une relation avec Hugo (écrivain et poète français) Cela inclut des éléments tels que le romantisme et l'engagement politique. "

Cet exemple souligne la sophistication du logiciel dans la gestion des dérivations lexicales spécifiques et des cas particuliers, assurant des définitions précises et contextuelles.

Exemple 7 : Exception - "Staline/stalination"

Notre logiciel est également conçu pour gérer des exceptions où des modifications spécifiques doivent être apportées aux noms propres avant d'ajouter un suffixe. Un exemple pertinent est celui de "Staline" et "stalination". Lorsqu'on forme le mot "stalination", il est nécessaire de retirer le "e" final du nom propre "Staline". Notre programme reconnaît cette règle morphologique particulière et traite correctement l'entrée.

Ainsi, lorsqu'on entre "stalination", le logiciel identifie "Staline" comme le nom propre, retire le "e" final, et ajoute le suffixe "-ation". Le programme fournit alors la définition "Stalination : Processus ou doctrine associée à Joseph Staline". Cet exemple démontre la capacité avancée de notre logiciel à gérer les exceptions morphologiques spécifiques, en appliquant des règles de transformation avant l'ajout des suffixes, garantissant ainsi des définitions précises et contextuelles.

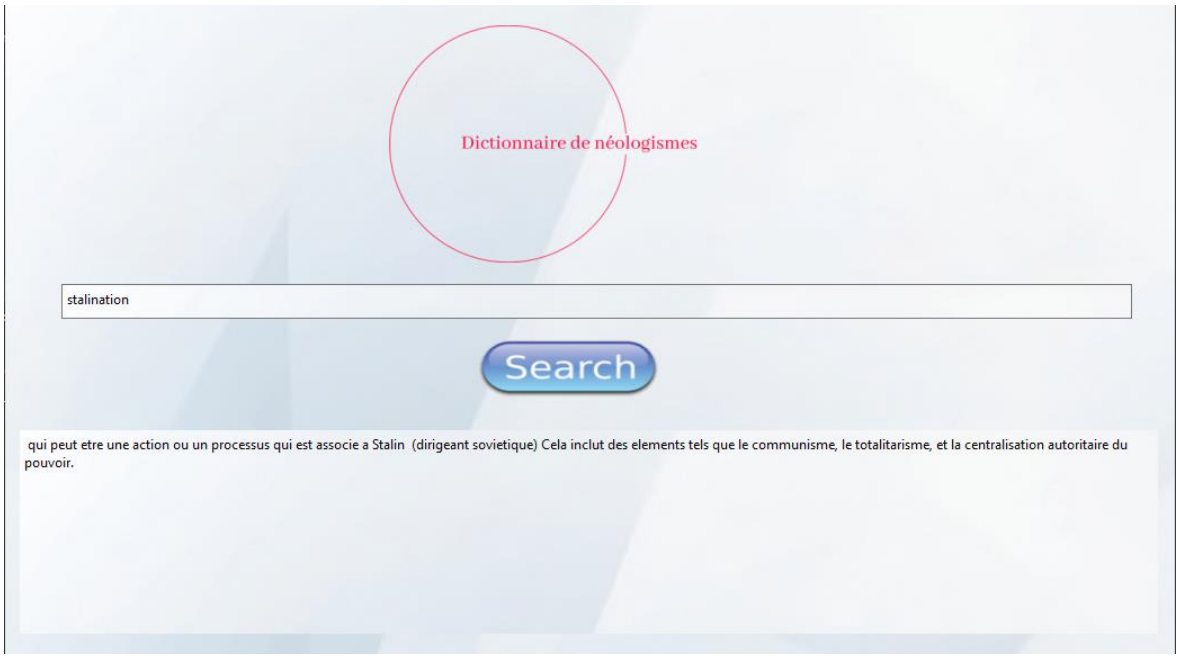

Figure 14: Exemple 7 : Exception - "Staline/stalination"

Exemple 8 : Intégration d'un Préfixe - "detrumpation"

Figure 15: Exemple 8 : Intégration d'un Préfixe - "detrumpation"

Notre logiciel est également conçu pour gérer l'intégration de préfixes avant l'ajout d'un suffixe dérivationnel. Un exemple pertinent est celui de "detrumpation". Lorsqu'on forme ce mot, il est nécessaire d'intégrer le préfixe "dé-" pour indiquer l'action de défaire quelque chose associé à "Trump". Notre programme reconnaît cette règle morphologique particulière et traite correctement l'entrée. Ainsi, lorsqu'on entre "detrumpation", le logiciel identifie "Trump" comme le nom propre, ajoute le préfixe "dé-", puis le suffixe "-ation".

Le programme fournit alors la définition "Detrumpation : rejet de ce qui peut être une action, politique ou un processus qui est associe à Trump (président américain) Cela inclut des éléments tels que le populisme, le nationalisme, et le conservatisme économique et social.". Cet exemple démontre la capacité avancée de notre logiciel à gérer l'intégration des préfixes et des suffixes, garantissant ainsi des définitions précises et contextuelles.

Ces exemples montrent la robustesse et la précision de votre logiciel dans l'analyse et l'interprétation des mots dérivés.

En traitant les flexions de nombre et de genre, ainsi que les exceptions morphologiques, le programme offre une solution complète et fiable pour l'analyse lexicale, démontrant son utilité et sa flexibilité dans divers contextes linguistiques.

3. Les lacunes du logiciel et les potentielles extensions

3.1. Les difficultés du Projet

Le développement d'un logiciel visant à effectuer un traitement automatique des mots communs dérivés à partir de noms propres, bien que prometteur, n'est pas exempt de certaines difficultés inhérentes qui méritent une réflexion approfondie. Ces difficultés, inhérentes à la nature complexe du langage et des données associées, peuvent potentiellement influencer la performance et l'efficacité du logiciel. L'une des difficultés clés concerne la sensibilité aux variations de saisie.

En raison de la diversité des façons dont les utilisateurs peuvent saisir ou formater les noms propres, le logiciel peut présenter une sensibilité accrue à ces variations. Des nuances subtiles dans la présentation des noms propres pourraient entraîner des résultats incohérents, nécessitant ainsi une gestion attentive de la qualité des données en entrée.

Une autre difficulté cruciale à considérer est l'ambiguïté morphologique inhérente à certains noms propres. Des situations telles que la présence d'homographes ou la possibilité d'interprétations multiples peuvent compliquer le processus de dérivation lexicale. Le logiciel doit relever ce défi en intégrant des mécanismes sophistiqués pour démêler ces ambiguïtés et fournir des résultats précis.

Le manque de données exhaustives représente également un défi significatif. La création d'un dictionnaire robuste de noms propres avec des caractéristiques associées dépend de la disponibilité de données complètes. Si les informations sur une personne sont incomplètes, la capacité du logiciel à effectuer une dérivation lexicale précise pour cette personne particulière pourrait être compromise.

Par ailleurs, l'évolution constante des noms propres au fil du temps peut constituer une difficulté dynamique. Les mises à jour fréquentes du dictionnaire sont essentielles pour suivre ces évolutions, mais cette exigence souligne la nécessité d'un mécanisme agile pour maintenir la pertinence du logiciel dans un paysage linguistique en constante évolution.

Enfin, les problèmes potentiels liés à la polyvalence des noms propres ajoutent une couche de complexité. Certains noms propres peuvent être utilisés dans des contextes variés, introduisant ainsi des nuances sémantiques difficiles à capturer de manière précise dans la dérivation lexicale des mots communs.En reconnaissant ces difficultés, le projet bénéficiera d'une approche proactive pour atténuer ces défis. Des mécanismes de gestion de l'ambiguïté, des stratégies de mise à jour efficaces du dictionnaire et des techniques pour traiter la polyvalence des noms propres seront autant de pistes à explorer pour renforcer la fiabilité et la robustesse du logiciel face à ces défis inhérents.

3.2. Les perspectives du projet

En considérant les perspectives à long terme de notre projet de développement, plusieurs axes d'évolution et d'amélioration se dessinent, ouvrant ainsi des horizons prometteurs pour renforcer la qualité et l'applicabilité du système.

Une première perspective cruciale réside dans l'amélioration continue de la précision. Nous envisageons d'explorer des approches avancées d'analyse sémantique et d'intégrer des modèles d'apprentissage automatique pour affiner la compréhension contextuelle. Cette démarche devrait contribuer significativement à une dérivation lexicale plus précise et nuancée.

L'intégration de la sémantique dans le processus de dérivation représente une autre perspective clé. En comprenant plus profondément les relations sémantiques entre les noms propres et les mots dérivés, le logiciel pourrait fournir des définitions plus riches, prenant en compte les subtilités du sens dans différents contextes.

L'adaptabilité à l'évolution du langage est une perspective essentielle. Le développement de mécanismes d'adaptabilité permettra au logiciel de suivre les changements dans les noms propres et les morphèmes dérivationnels, assurant ainsi sa pertinence continue dans un environnement linguistique en constante évolution.

Une extension du logiciel vers d'autres langues constitue une perspective d'expansion importante. En tenant compte des particularités morphologiques et culturelles propres à chaque langue, cette extension permettra de rendre le logiciel plus polyvalent et applicable à une diversité de contextes linguistiques.

Enfin, l'identification de domaines spécifiques d'application demeure une perspective pratique. En adaptant le logiciel pour répondre aux besoins spécifiques de la recherche académique, du journalisme ou de la veille stratégique.

Notre utilité sera maximisée, offrant des solutions ciblées et pertinentes dans des contextes d'utilisation spécifiques, notamment dans l'analyse fine des tendances politiques émanant des discours en ligne.

Un logiciel qui a la capacité d'analyse des tendances politiques sur les réseaux sociaux porte en elle des avantages considérables, ouvrant ainsi de nouvelles perspectives dans la compréhension du discours politique en ligne.

Dans le cadre de ce projet, l'un des premiers avantages à souligner réside dans la capacité du logiciel à réaliser une identification précise des termes politiques dérivés à partir de noms propres. Cette fonctionnalité promet d'apporter une profondeur d'analyse significative, permettant une compréhension détaillée du langage spécifique utilisé dans le contexte politique sur les plateformes en ligne.

Une caractéristique clé de notre logiciel réside dans sa capacité à détecter en temps réel les tendances émergentes dans le discours politique en ligne. En scrutant continuellement les nouveaux mots dérivés, il offre un aperçu instantané des sujets et des idées qui suscitent l'attention des utilisateurs, fournissant ainsi un outil puissant pour rester à la pointe de l'évolution des débats politiques.

Le suivi de l'évolution du langage politique au fil du temps constitue un autre atout majeur. Grâce à une analyse constante des mots dérivés, notre logiciel offre une perspective historique précieuse, permettant d'observer et de comprendre les changements linguistiques dans le discours politique, offrant ainsi une vision contextuelle indispensable.

L'analyse des caractéristiques associées aux personnalités politiques est une fonctionnalité novatrice. En utilisant un dictionnaire de noms propres enrichi de traits spécifiques, le logiciel offre une approche approfondie des tendances politiques en mettant en lumière les traits, les thèmes et les concepts récurrents associés à certaines personnalités. Cette dimension enrichie la compréhension de la perception en ligne de ces figures politiques.

Le logiciel excelle également dans l'identification des mots-clés essentiels associés aux discours politiques. Cette capacité fournit une vue détaillée des thèmes dominants dans les conversations en ligne, permettant une analyse fine des préoccupations politiques majeures et des sujets clés qui façonnent les débats.

Un aspect innovant de notre logiciel réside dans sa capacité potentielle à contribuer à la prédiction des tendances politiques futures. En analysant les tendances émergentes, il pourrait offrir une perspective avant-gardiste sur les mouvements futurs dans l'opinion publique, contribuant ainsi à anticiper les sujets politiques à venir.

L'application pratique de ces résultats dans des contextes de recherche académique, de journalisme politique ou de veille stratégique souligne l'utilité polyvalente de notre logiciel. Les professionnels de ces domaines bénéficieraient ainsi d'informations actualisées sur les dynamiques politiques en ligne, consolidant ainsi la pertinence et la valeur de notre projet.

En combinant ces avantages, notre projet aspire à créer une plateforme complète d'analyse des tendances politiques, apportant ainsi une contribution significative à la compréhension des opinions publiques et des dynamiques politiques contemporaines.

En envisageant ces perspectives, le projet évoluera vers une solution plus sophistiquée, adaptable et pertinente, répondant ainsi aux enjeux du traitement automatique des mots dérivés issus de noms propres dans un paysage linguistique en constante évolution.

En conclusion, nous avons exploré en détail le fonctionnement du logiciel, en mettant en lumière le processus de traitement d'un mot d'entrée. Ce processus débute donc par une comparaison systématique du mot d'entrée avec une base de données de noms prédéfinis.

Cette première étape est essentielle pour identifier les correspondances potentielles qui orienteront le traitement ultérieur. Une fois ces correspondances trouvées, le logiciel procède à l'extraction des informations pertinentes associées, ce qui constitue une phase cruciale car la qualité des données extraites détermine la pertinence de la réponse finale.

Enfin, le logiciel construit une réponse basée sur ces informations, visant à répondre de manière appropriée au mot d'entrée fourni par l'utilisateur.

Après cela, nous avons illustré ce processus à travers divers exemples concrets, démontrant ainsi l'efficacité et la polyvalence du logiciel. Ces exemples ont mis en évidence la capacité du programme à gérer différents mots d'entrée en prenant compte des exceptions et des variations en genre et en nombre et à produire des réponses adaptées. Le logiciel parvient alors à répondre de manière précise et pertinente, améliorant ainsi considérablement l'expérience utilisateur.

Cette analyse a permis de souligner non seulement le fonctionnement interne du logiciel, mais également son potentiel à traiter efficacement les données linguistiques pour générer des réponses cohérentes. Les mécanismes de comparaison, d'extraction et de construction de réponses se révèlent être des éléments clés qui rendent ce programme fiable pour de nombreuses applications pratiques.

Conclusion générale

Dans le cadre de notre mémoire, nous avons exploré le domaine de la création d'un logiciel de traitement automatique visant à attribuer un sens aux mots dérivés de noms propres, les anthroponymes spécifiquement. Ce sujet est au cœur des avancées technologiques actuelles, où la compréhension et l'analyse du langage naturel sont essentielles.

L'objectif principal de cette étude était d'élaborer un outil innovant capable de décoder et d'interpréter les nuances sémantiques des mots dérivés de noms propres, offrant ainsi une perspective nouvelle dans le domaine de la linguistique informatique. À travers une approche méthodique et rigoureuse, chaque étape de ce projet a été minutieusement analysée et développée, avec pour ambition ultime de contribuer à l'évolution des technologies linguistiques et à l'amélioration de la compréhension automatique du langage humain.

En premier lieu, nous avons amorcé notre étude en examinant de manière approfondie les connaissances contemporaines liées à notre sujet de recherche. Notre analyse s'est concentrée sur les différentes disciplines qui jouent un rôle essentiel dans le domaine du traitement automatique du langage. Nous avons initié ce chapitre en présentant la morphologie computationnelle, les ressources lexicales et les ontologies. Ces éléments constituent les fondements théoriques et méthodologiques de ce domaine interdisciplinaire, apportant chacun leur contribution à la résolution des problématiques liées aux dérivés de noms propres.

Après cela, nous avons approfondi notre étude en linguistique en mettant en lumière les fondements théoriques indispensables à notre recherche. Nous avons également présenté les travaux majeurs qui se penchent sur les domaines de la lexicologie et de la morphologie, offrant ainsi un panorama complet des références clés dans notre domaine d'étude. De plus, nous avons explicité les définitions des concepts pertinents pour notre recherche, ce qui nous a permis d'établir une base solide pour notre analyse et l’utilisation d’un système standardisé et scientifique.

Également, les disciplines telles que la morphologie computationnelle, les ressources lexicales et les ontologies nous ont permis d'acquérir une compréhension approfondie de la dynamique évolutive du langage.

Cette consolidation des connaissances théoriques et méthodologiques nous positionne favorablement pour aborder les défis liés aux dérivés de noms propres et pour explorer de nouvelles perspectives dans ce domaine interdisciplinaire et constitue donc une base solide pour orienter nos futurs travaux et faire progresser notre projet de recherche dans le domaine du traitement automatique du langage.

Dans un second lieu nous mettons en lumière le domaine de l'informatique. D'abord à la présentation des bases fondamentales de l'informatique que nous avons exploitées pour la réalisation de notre prototype. Nous avons minutieusement détaillé les outils, les technologies mises en œuvre, ainsi que la méthodologie adoptée tout au long du développement de notre application.

Cette immersion dans le domaine de l'informatique s'avère cruciale pour la conception et la concrétisation de notre projet. En intégrant de manière approfondie les principes et les pratiques essentiels de l'informatique, nous avons pu garantir l'efficacité et la pertinence de notre approche de recherche.

Ensuite, nous avons examiné la faisabilité d'intégrer un modèle d'apprentissage automatique dans notre logiciel. L'objectif principal était d'élargir la base de données des caractéristiques des noms propres et des morphèmes dérivationnels. Cette approche avant-gardiste visait à enrichir notre système en exploitant les capacités de l'apprentissage automatique pour une analyse plus approfondie et précise des données linguistiques.

Cette démarche nous a permis d'explorer de nouvelles avenues pour améliorer la qualité et la richesse des informations linguistiques traitées par notre logiciel. En intégrant des technologies de pointe telles que l'apprentissage automatique, nous avons ouvert la voie à des possibilités de développement et d'optimisation significatives pour notre projet.

Nous avons aussi pris soin de justifier notre choix de nous concentrer exclusivement sur les anthroponymes et non sur l'ensemble des noms propres. Cette décision découle principalement de leur fréquence d'utilisation plus élevée par rapport à d'autres catégories de noms propres. En se focalisant sur les anthroponymes, qui représentent une part significative des noms propres couramment utilisés, nous visons à maximiser la pertinence et l'applicabilité de notre étude.

En mettant en lumière cette justification, nous soulignons l'importance stratégique de cibler spécifiquement les anthroponymes pour une analyse approfondie et significative. Cette approche nous permet de concentrer nos efforts de recherche sur un domaine linguistique essentiel et largement répandu, offrant ainsi une base solide pour notre étude et son application pratique.

Dans le cadre de notre recherche, nous avons également exposé en détail notre démarche pour acquérir une base de données de noms de famille. Pour ce faire, nous avons d'abord opté pour l'utilisation de données téléchargées depuis la plateforme data.world. Cette plateforme est renommée pour sa vaste collection de jeux de données publics et collaboratifs, offrant ainsi un accès privilégié à une source fiable et riche en informations.

En exploitant les ressources disponibles sur data.world, nous pouvons accéder à une variété de données pertinentes et actualisées concernant les noms de famille. Cette approche nous permet de disposer d'une base de données solide et diversifiée, essentielle pour mener à bien notre étude sur les anthroponymes et enrichir notre analyse linguistique. Mais nous avons finalement opté pour un simple fichier txt a titre d'exemple pour ce prototype. Notre raisonnement pour ce choix a aussi été justifié.

En somme, nous avons présenté en détail notre démarche technique ainsi que nos réflexions sur les développements futurs, mettant en lumière la robustesse et l'évolutivité de notre application. Nous avons exposé les bases fondamentales de notre approche technique en soulignant sa solidité.

De plus, nous avons abordé les perspectives d'amélioration et d'expansion de notre application, démontrant ainsi son potentiel évolutif. Cette analyse approfondie a permis de mettre en avant la qualité et la durabilité de notre travail, tout en laissant entrevoir les opportunités de croissance à venir.

Pour finir, nous avons examiné en détail le fonctionnement du logiciel, en mettant en évidence le processus de traitement d'un mot d'entrée. Ce processus démarre par une comparaison systématique du mot d'entrée avec une base de données de noms prédéfinis. Nous avons analysé en profondeur comment le logiciel effectue cette comparaison pour identifier et traiter efficacement les mots d'entrée, ce qui constitue un élément clé de son fonctionnement.

Dans le processus mentionné, la première étape de comparaison avec la base de données de noms prédéfinis permet d'identifier les correspondances potentielles, créant ainsi une base solide pour la suite du traitement. Cette étape initiale est cruciale car elle oriente le logiciel vers les données pertinentes à analyser plus en détail.

Une fois que les correspondances sont repérées, le logiciel passe à l'extraction des informations associées. Cette phase revêt une importance majeure car la qualité des données extraites influencera directement la pertinence de la réponse finale. Il est essentiel que le logiciel soit capable d'extraire avec précision les informations nécessaires pour garantir une réponse adéquate à l'utilisateur.

Enfin, sur la base des informations extraites, le logiciel construit la réponse qu'il fournira à l'utilisateur. Cette étape finale vise à utiliser efficacement les données extraites pour formuler une réponse pertinente et adaptée au mot d'entrée fourni par l'utilisateur. La qualité de cette réponse dépend directement de la précision de l'extraction des informations et de la capacité du logiciel à les interpréter de manière adéquate.

Après avoir décrit le processus en détail, nous avons illustré ce fonctionnement à travers divers exemples concrets. Ces exemples ont démontré l'efficacité et la polyvalence du logiciel, mettant en lumière sa capacité à traiter efficacement différents mots d'entrée. Le logiciel a montré sa capacité à gérer les exceptions, les variations en genre et en nombre, et à produire des réponses adaptées en conséquence.

Grâce à cette capacité d'adaptation et de traitement précis, le logiciel est en mesure de répondre de manière pertinente et précise, ce qui améliore significativement l'expérience utilisateur. Cette démonstration concrète des fonctionnalités du logiciel a permis de mettre en avant sa fiabilité et sa capacité à fournir des réponses de qualité, renforçant ainsi sa valeur dans divers contextes d'utilisation.

Lors de l'exploration détaillée du logiciel, nous avons mis en lumière le processus de traitement d'un mot d'entrée, soulignant l'importance de la comparaison systématique avec une base de données de noms prédéfinis. Cette étape initiale est essentielle pour identifier les correspondances potentielles qui guideront le traitement ultérieur.

Une fois ces correspondances trouvées, le logiciel extrait les informations pertinentes associées, une phase cruciale car la qualité des données extraites détermine la pertinence de la réponse finale. Enfin, le logiciel construit une réponse basée sur ces informations, visant à répondre de manière appropriée au mot d'entrée fourni par l'utilisateur.

Par la suite, nous avons illustré ce processus à travers divers exemples concrets, mettant en avant l'efficacité et la polyvalence du logiciel. Ces exemples ont démontré sa capacité à gérer différents mots d'entrée, en tenant compte des exceptions et variations en genre et en nombre, pour produire des réponses adaptées. Grâce à cette capacité, le logiciel parvient à répondre de manière précise et pertinente, améliorant ainsi considérablement l'expérience utilisateur.

En analysant le logiciel, nous avons souligné son fonctionnement interne et son potentiel à traiter efficacement les données linguistiques pour générer des réponses cohérentes.

Les mécanismes de comparaison, d'extraction et de construction de réponses se révèlent être des éléments clés qui rendent ce programme fiable pour de nombreuses applications pratiques. Cette analyse a mis en lumière la fiabilité et l'adaptabilité du logiciel, renforçant sa valeur dans divers contextes d'utilisation.

Références bibliographiques

Ouvrages

- APOTHLÉOZ Denis, 2002, *La construction du lexique français*, Ophrys, Paris.
- BEESLEY Kenneth R, KARTTUNEN Lauri, 2003, *Finite State Morphology*, Center for the Study of Language and Information, Californie
- BEESLEY Kenneth R, 2004, *Morphological Analysis and Generation:A First-Step in Natural Language Processing*,2004, Meylan, France.
- CORBIN Danielle, 1991, *Morphologie dérivationnelle et structuration du lexique (Tome I et II*), Presses universitaires du septenrion.
- DUBOIS Jean et Al, 2012, *Dictionnaire de linguistique et des sciences du langage*, éd Larousse. Paris.
- FRADIN Bernard, 2003, *Nouvelles approches en morphologie*, Presses universitaires de France. Paris.
- GAUDIN François et Louis GUESPIN, 2000, *Initiation à la lexicologie française. De la néologie aux dictionnaires*, Duclot, Bruxelles.
- MEL'CUK Igor A., André CLAS, Alain POLGUERE, 1995, *Introduction à la lexicologie explicative et combinatoire*, Editions Duclot, Louvain-la-Neuve.
- GARDES-TAMINE, Joelle, 2008, *La grammaire. Phonologie, Morphologie, Lexicologie*, Armand Colin, Paris.
- GUILBERT Louis, 1975, *La créativité lexicale*, Librairie Larousse, Paris.
- LEHMANN Alise & Françoise. MARTIN-BERTHET, 2013, *Lexicologie, Sémantique, Morphologie, Lexicographie*, Armand Colin, Paris
- MATORÉ Georges, 1953, *La méthode en lexicologie*, Marcel Didier 4 et 6, rue de la Sorbonne, Paris.
- MORTUREUX Marie-Françoise, 2008, *La lexicologie entre langue et discours*, Armand Colin, Paris.
- MOUNIN Georges, 1974, *Dictionnaire de la linguistique*,Puf, Paris.
- NIKLAS-SALMIEN Aino, 1997, *La lexicologie*, Armand Colin, Paris.
- PRUVOST Jean, Sablayrolles Jean-François, 2019, *Les néologismes*, Que sais-je?, Paris.

- Sablayrolles Jean-François, 2000, *La néologie en français contemporain. Examen du concept et analyse de productions néologiques récentes*, Honoré Champion, Paris.
- SAUSSURE Ferdinand. (DE), 2016, *Cours de linguistique générale*, Editions Talantikit, Bejaia.

Articles

- Moussaouer, Abderahim 2017/ 2, « Les néologismes polylexicaux et le défigement dans l'humour de Fellag : de la créativité à l'interprétation », *Dans Éla. Études de linguistique appliquée*, (N°186), pages 151 à 165. Éditions Klincksiec. https://www.cairn.info/revue-ela-2017-2-page-151.htm&wt.src=pdf consulté le 06/04/2024.
- DAL G., HATHOUT Nabil, NAMER Fiammetta. 2004, « Morphologie constructionnelle et traitement automatique des langues : le projet MORTAL », Université de Lille, France.consulté le 06/04/2024.
- Dal GEORGETTE, 2019, « État actuel sur les études en morphologie en France et à l'international » https://hal.science/hal-03639345/documentconsulté le 06/04/2024.
- GUILBERT Louis,1973 « Théorie du néologisme », *Cahiers de l'Association internationale des études françaises*, n°25. pp. 9-29. https://www.persee.fr/doc/caief_0571-5865_1973_num_25_1_1020consultéle 06/04/2024.
- MARCELLESI. Chr, 1974, « Néologie et fonctions du langage. », *Langages, Numéro 36.* Persée [en ligne], consulté le : 06-04-2024.https://www.persee.fr/doc/lgge_0458-726x_1974_num_8_36_2278consulté le 06/04/2024.
- MOLINO Jean, 1985, « Où en est la morphologie ? », *Langages*, 20e année, n°78. Le retour de la morphologie https://www.persee.fr/doc/lgge_0458-726x_1985_num_20_78_2462consulté le 06/04/2024

- MORTUREUX M. F, 2002, « Ephémérité est-il français ? », Linx [en ligne]. https://journals.openedition.org/consulté le 06/04/2024

Site internet

- https://www.home.uniosnabrueck.de/bschwisc/archives/formation.htm#_Toc530742580 consulté le 07/01/2024
- https://aleph.edinum.org/5736#tocto1n3 consulté le 10/01/2024
- https://www.cairn.info/nouvelles-approches-en-morphologie--9782130515487.htm consulté le 15/01/2024
- https://online.maryville.edu/online-bachelors-degrees/computer-science/careers/categories-of-programming-languages/ consulté le 01/02/2024
- https://pythonnumericalmethods.studentorg.berkeley.edu/notebooks/chapter03.01-Function-Basics consulté le 02/02/2024
- https://sites.pitt.edu/~naraehan/python3/mbb20 consulté le 06/02/2024
- https://textbooks.cs.ksu.edu/intro-python/06-functions/04-param-arg/ consulté le 06/02/2024
- https://cs.du.edu/~intropython/intro-to-programming/function_parameters consulté le 11/02/2024
- https://textbooks.cs.ksu.edu/intro-python/06-functions/02-basics/ consulté le 17/02/2024
- https://cs.stanford.edu/people/nick/py/python-function consulté le 18/02/2024
- https://pll.harvard.edu/course/introduction-data-science-python consulté le 20/0/2024

Thèse et mémoire

- REFRAFI S , Thèse de doctorat, Université de BISKRA, 2019/2020, P : 41http://thesis.univ-biskra.dz/4980/1/these%202%20pdf%201.pdf/
- Sidi Salah Tassadit, Senoussi Ahlem, Mémoire de master, Université de BEJAIA,2016/2017, P :18 https://www.univ-bejaia.dz/xmlui/handle/123456789/101/

Résumé

Le domaine du Traitement Automatique des Langues (TAL) vise, entre autres, à créer des applications pour faciliter le traitement des données linguistiques à grande échelle. La linguistique sert, dans ce cadre, à fournir des données structurées et normalisées facilitant leur automatisation.

Ce mémoire de recherche propose le développement d'un modèle de prototype pour le traitement automatique des néologismes dérivés d'anthroponymes. Ce prototype automatise la segmentation de ces néologismes, pour une compréhension détaillée des termes à partir de leurs composants morphologiques.

Chaque néologisme est accompagné d'une définition qui expose ses caractéristiques spécifiques et ses concepts clés. Cette approche cherche à fournir un outil pour l'analyse et la compréhension des nouveaux termes linguistiques.

Abstract

The field of Automatic Language Processing (ALP) aims, among other goals, to create software to facilitate the processing of linguistic data on a large scale. In this context, linguistics is used to provide structured and standardized data to facilitate its automation.

This research paper proposes the development of a prototype model for the automatic processing of neologisms derived from anthroponyms. This prototype automates the segmentation of these neologisms, providing a detailed understanding of the terms based on their morphological components.

Each neologism is accompanied by a definition outlining its specific characteristics and key concepts. The purpose of this approach is to provide a tool for analyzing and understanding new linguistic terms.

الملخص

يهدف مجال المعالجة الآلية للغات(ALP) ، إلى إنشاء تطبيقات لتسهيل معالجة البيانات اللغوية على نطاق واسع. وفي هذا السياق، تُستخدم اللسانيات لتوفير بيانات منظمة وموحدة لتسهيل المعالجة اللغوية.

يقترح هذا البحث تطوير نموذج مثالي أولي للمعالجة الآلية للمصطلحات الجديدة المشتقة من الأسماء البشرية. يعمل هذا النموذج الأولي على تجزئة هذه المصطلحات الجديدة، من أجل فهم مفصّل للمصطلحات بناءً على مكوناتها الصرفية.

ويرافق كل مصطلح جديد تعريف يحدد خصائصه المميزة ومفاهيمه الرئيسية. والهدف من هذه المنهجية هو توفير أداة لتحليل وفهم المصطلحات اللغوية الجديدة المشتقة من الأسماء البشرية .

Mots-clés

TAL ; Traitement Automatique des Langues ; Linguistique ; Sciences du langage ; Lexicologie ; Noms propres ; Dérivation ; Mots dérivés ; Suffixe ; Préfixe ; Dictionnaire automatique ; Analyse lexicale ; logiciel ; programmation.