Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique
Université dAbou bakr Belkaïd – Tlemcen
Faculté des Sciences
Département d'Informatique

MEMOIRE DE PROJET DE FIN D'ETUDES

En vue de l'obtention du diplôme de master en informatique

Spécialité : Système d'informations et Connaissances

Par :

Mlle MAIDRI FATIMA ZOHRA

Sur le Thème

EVALUATION DES METHODES DE DESAMBIGUÏSATION DANS LA CATEGORISATION DES TEXTES

Soutenu publiquement le 25 juin 2025 à Tlemcen devant le jury composé de :

Dr ABDERRAHIM Mohammed Alaeddine	**Président**
Dr BENTALLAH Mohammed Amine	**Encadrant**
Dr REMACI Zeyneb Yasmina	**Examinatrice**

Année Universitaire : 2024 / 2025

MAIDRI/2025

Remerciements :

Je tiens à exprimer ma profonde gratitude à mon encadrant Dr. BENTALLAH Mohammed Amine, Maitre de conférences et enseignant au département d'informatique de la faculté des Sciences université de Tlemcen, pour sa disponibilité, ses conseils judicieux et son continuel suivi du travail qui ont guidé ma réflexion tout au long de ce mémoire.

Je remercie également les membres du jury Dr ABDERRAHIM Mohammed Alaeddine en tant que Président du Jury et Dr RAMACI Zeyneb en tant examinatrice, pour avoir accepté d'examiner ce travail.

Ma reconnaissance va également à ma famille pour leur amour, leur patience et leurs encouragements, tout au long de ce parcours.

Un grand merci à ma tante SENOUDI Assia pour son soutien et présence encourageante tout le long de mes études.

A mes amies de Classe et à tous les enseignants du département d'informatique de la faculté des Sciences.

Dédicace

Je dédie ce travail à :

Mes chers parents qui méritent tous mon respect et mes remerciements pour leur amour, leur confiance et leur soutien à chaque étape de mon parcours.

Mes sœurs Khadija, Aicha Nour han et mon frère Mohammed, que Dieu vous protège.

Ma chère grand-mère maternelle, mes oncles et mes tantes, pour leur présence bienveillante.

A mes cousines, cousins et mes amies.

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ، وَالصَّلَاةُ وَالسَّلَامُ عَلَى أَشْرَفِ الْمُرْسَلِينَ، سَيِّدِنَا مُحَمَّدٍ وَعَلَى آلِهِ وَصَحْبِهِ أَجْمَعِينَ.

أما بعد:

Table des matières

Table des Figures :

Liste des tableaux :

Liste des sigles et acronymes :

TALN	*Traitement Automatique du Langage Naturel*
WSD	*Word Sense Disambiguation (Désambiguïsation lexicale)*
CT	*Catégorisation de Textes*
FN	*Faux Négative*
FP	*Faux Positive*
VN	*Vrai Négative*
VP	*Vrai Positive*
SVM	*Support Vector Machine*
K-NN	*K-Nearest Neighbors (K plus proche voisins)*
P	*Précision*
R	*Rappel*
WN	*WordNet*
MFS	*Most Frequent Sense (Sens le plus fréquent)*
PPR	*Personalized Page Rank*
UKB	*Unsupervised Knowledge-Based*
BERT	*Bidirectional Encoder Representations from Transformers*
SVD	*Singular Value Decomposition*
LSA	*Latent Semantic Analysis*
ExtJWNL	*extended Java WordNet Library*

Introduction générale

Le traitement automatique du langage naturel (TALN) est aujourd'hui un champ de recherche en plein essor, dont le but est de permettre aux machines d'analyser, d'interpréter et de manipuler le langage humain de manière efficace. Cette discipline interdisciplinaire, à la croisée de l'informatique, de linguistique et de l'intelligence artificielle, cherche à développer des systèmes capables de traiter automatiquement les textes dans toute leur diversité et complexité. Parmi de nombreuses applications du TALN, la catégorisation automatique de texte occupe une place centrale, offrant des solutions pour organiser et structurer l'information textuelle de manière systématique.

Cependant, la mise en œuvre pratique de ces systèmes révèle des défis technologiques considérables liés à la nature intrinsèquement ambiguë du langage humain. La richesse sémantique et la flexibilité linguistique, qui constituent la force expressive du langage naturel, se transforment en obstacles complexes pour les algorithmes de traitement automatique. L'un des obstacles majeurs auxquels font face ces systèmes réside dans la polysémie des mots, où un même terme peut revêtir plusieurs significations selon le contexte dans lequel il apparaît. Cette ambiguïté lexicale peut considérablement affecter la performance des algorithmes de classification, conduisant à des erreurs d'interprétation et des assignations incorrectes des catégories.

La désambiguïsation lexicale (Word Sense Disambiguation – WSD) émerge come une solution prometteuse à cette problématique. Cette technique vise à identifier automatiquement le sens approprié d'un mot ambigu dans un contexte donné, permettant ainsi une compréhension plus précise du contexte textuel. L'intégration de méthodes de WSD dans les systèmes de catégorisation de texte représente donc un axe de recherche important pour améliorer leur efficacité et leur fiabilité.

L'évaluation comparative de l'impact des différentes approches de WSD sur la performance des systèmes de catégorisation constitue un enjeu scientifique et technologique majeur. Il s'agit principalement de quantifier et de comparer les performances des différentes approches à travers l'analyse de métriques d'évaluation standardisées, permettant ainsi d'établir une hiérarchisation objective des méthodes de WSD dans le contexte spécifique de la catégorisation de texte. Cette évaluation comparative basée sur les métriques constitue un préalable essentiel pour guider les choix technologiques et orienter les recherches futures dans ce domaine.

Les enjeux de cette recherche dépassant le cadre purement académique et touchent de nombreux secteurs industriels et de services : système de recommandation personnalisés, analyse automatique de littérature scientifique, classification de brevets, surveillance de la réputation en ligne, aide au diagnostic médical par analyse de dossiers patient. Dans chacun de ces domaines l'amélioration de la précision de catégorisation grâce à une désambiguïsation lexicale efficace peut générer des gains substantiels en termes de pertinence des résultats et d'efficacité opérationnelle.

Notre mémoire est structuré en trois chapitres distincts :

Le premier chapitre est consacré à la catégorisation de texte. Nous commençons par présenter de manière approfondie le processus de catégorisation, en décrivant ses différentes étapes. Nous avons également décrit les méthodes de classification les plus connues et leurs principes fondamentaux, ainsi que les applications et les problèmes majeurs de cette tâche du traitement automatique du langage naturel (TALN).

Le deuxième chapitre est dédié à la désambiguïsation lexicale (WSD - Word Sense Disambiguation). Nous avons commencé par une définition de la WSD et présenté les différentes sources de connaissances nécessaires à cette tâche. Nous avons expliqué quelques algorithmes connus de la WSD et leurs caractéristiques.

Le troisième chapitre présente notre protocole expérimental détaillé. Nous y expliquons les étapes principales de notre système et décrivons la méthodologie adoptée pour évaluer limpact des approches de désambiguïsation lexicale sur la catégorisation de texte. Nous avons appliqué cinq approches de désambiguïsation lexicale basées sur le contexte, en nous appuyant sur l'ontologie WordNet (UKB, Lesk simplifié, MFS, all senses, hybride par vote) sur quatre méthodes de classification : SVM, Naive Bayes, K-NN et Random Forest. Dans limplémentation de la méthode UKB, nous avons apporté quelques modifications et adopté deux variantes : UKB avec seuil dynamique et UKB sans seuil. Nous avons également tenté dimplémenter une autre méthode de WSD par vote entre UKB et Lesk simplifié. Finalement, nous avons évalué les résultats des systèmes de classification avec chaque approche de WSD.

Cette évaluation empirique constitue le cœur de notre contribution, apportant un éclairage quantitatif sur lefficacité relative des différentes méthodes de désambiguïsation lorsquelles sont intégrées aux systèmes de catégorisation. Lanalyse comparative permet didentifier les approches les plus performantes et de hiérarchiser les méthodes selon leur capacité à améliorer la précision de classification.

Létude se conclut par une synthèse comparative des résultats expérimentaux, identifiant les méthodes de désambiguïsation les plus efficaces et proposant des orientations pour les recherches futures.

Chapitre 1 : Catégorisation de texte

1.1. Introduction :

Depuis le début de l'ère numérique la quantité d'informations a connu une grande évolution. En 2024, des estimations indiquaient que le nombre de pages avait indexé par Google environ 400 milliards de pages [1]. Cette augmentation massive des données que ce soit dans les documents, articles, courriels et autre type de contenus numériques a engendré une surcharge informationnelle. Une difficulté d'accéder aux informations pertinentes survient en surface ce qui pose un véritable défi : comment organiser efficacement ces informations pour pouvoir les exploiter rapidement et facilement [2] ?

Historiquement les documents étaient classés manuellement selon des critères souvent propres à chaque structure. Cette méthode posait toujours des problèmes et constituait une source d'incohérences et d'erreurs, ce qui rend la recherche, l'exploitation des données et la prise de décision peu efficace. La nécessité d'organiser et de classer les informations efficacement a conduit au développement de la catégorisation automatique du texte [2]. Ce processus permet de classer automatiquement les documents en différents catégories prédéfinies en fonction de leur contenue [3].

Ce chapitre définit la catégorisation de texte tout en approfondissant son processus, ses méthodes et ses domaines d'application, mettant ainsi en évidence son importance dans la gestion et la recherche de l'information.

1.2. Définition de la catégorisation de texte :

La catégorisation de texte (CT) est le processus qui sert à assigner à un document une ou plusieurs catégories (classes) ont été définies soit préalablement par un expert, en analysant son contenu. L'objectif principale est d'effectuer une classification des documents de manière automatique.

Il existe plusieurs définitions de (CT) depuis son apparition, on peut citer les suivants :

Définition 01 : La CT consiste à chercher une liaison fonctionnelle entre *un ensemble de textes* et *un ensemble de catégories* (étiquettes, classes). Cette liaison fonctionnelle, que l'on appelle *modèle de prédiction*, est estimée par un apprentissage automatique [4].

Définition Formelle : La CT est la tâche consistant à attribuer une valeur booléenne à chaque paire $<dj, cj> \in D \times C$, où D est un domaine de documents et $C = \{c_1, \ldots, c|c|\}$ est un ensemble de catégories prédéfinies. Si T (true) est attribué à $<dj, cj>$, cela signifie que le document dj est classé dans la catégorie ci. Si F(false) est attribué, cela signifie qu'il ne l'est pas. L'objectif est de créer un classificateur capable de reproduire au mieux la manière dont les documents devraient être classés [3].

1.3. Processus de la catégorisation de texte :

Le processus de catégorisation repose sur une construction d'un modèle de prédiction. Il analyse le contenu et la signification du texte, et l'apprend comme une entrée, puis applique un étiquetage afin de lui attribuer l'étiquette la plus appropriée.

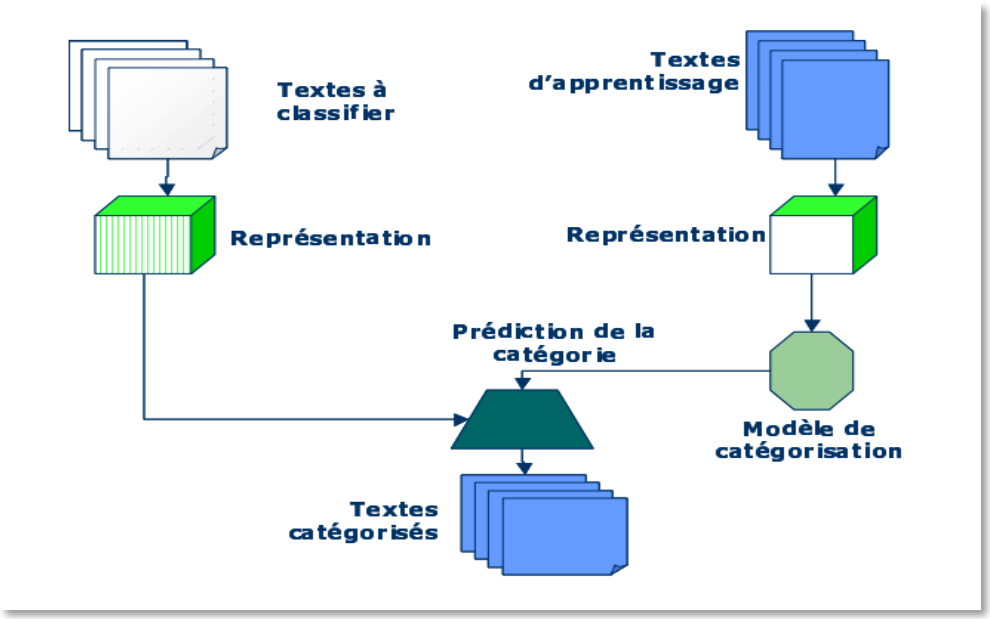


Figure 1 : Processus de la catégorisation de texte [4].

La Figure 1 représente le processus de catégorisation de texte. Comme l'illustre l'image on peut distinguer deux phases principales :

1. **Phase d'apprentissage :**
   - Collection des textes d'apprentissage.
   - Prétraitement des textes.
   - Représentation des textes.
   - Construction du modèle de catégorisation.
   - Evaluation du modèle.

2. **Phase de classification (ou d'exploitation) :**
   - Prétraitement des nouveaux textes.
   - Représentation des textes.
   - Prédiction de la catégorie (avec le modèle entraîné).
   - Attribution des catégories.

1.3.1. Représentation de texte :

« Puisqu'il n'existe pas actuellement une méthode d'apprentissage capable d'exploiter un texte directement sans que ce dernier soit bien structuré, une transformation préliminaire est alors indispensable » [3].

Les algorithmes d'apprentissage ne peuvent pas traiter directement ces données brute (données non structurées). Il est donc nécessaire de les convertir en vecteurs numériques avant de pouvoir les utiliser pour la modélisation. Il existe plusieurs types de représentations, voici quelques-uns :

1. **Représentations vectorielles :**

- **Sac de mot (Bag of words):**

  La représentation des textes la plus simple a été introduite dans le cadre du modèle vectoriel, elle représente un texte sous forme d'un vecteur dont chaque composante, fonction d'occurrence des mots, correspond à un terme. Cette représentation des textes exclut toute analyse grammaticale et toute notion de distance entre les mots, mais elle présente l'inconvénient de ne pas prendre en compte l'ordre des mots, limitant ainsi la capture du contexte sémantique [5].

- **Word Embedding :**

  Le Word Embedding (ou plongement lexical) est une méthode d'encodage qui vise à représenter les mots ou les phrases d'un texte par des vecteurs de nombres réels, décrit dans un modèle vectoriel. Le principe est de représenter chaque mot du vocabulaire **V** étudié par un vecteur de taille **m** et de projeter chacun de ces mots dans un espace vectoriel d'une taille fixe **N**. Cette représentation permet de capturer les relations sémantiques et syntaxiques entre les mots dans un espace vectoriel continu de faible dimension [6].

2. **Représentation conceptuelle :**

Cette méthode consiste à représenter un document sous forme d'un ensemble de concepts, généralement issu d'une ontologie ou d'un réseau sémantique, afin de capturer les relations sémantiques entre les termes. Elle permet d'utiliser des ressources comme **WordNet** pour créer des profils de catégories qui contiendront des concepts (synsets) caractérisant au mieux une catégorie par rapport aux autres [7]. Cette approche améliore la précision de la classification en exploitant les relations sémantiques entre les concepts.

3. **Représentation statique (N-gramme) :**

C'est une approche statique qui consiste à représenter un document par une séquence de n éléments consécutifs (mots ou caractères) extraits d'un texte. Elle permet de capturer les relations contextuelles entre les mots et est particulièrement efficace pour la classification de textes courts ainsi que pour la gestion des langues à morphologie riche, comme larabe et lallemand.

1.3.2. Mise en place du modèle de catégorisation :

Une fois les caractéristiques extraites, plusieurs algorithmes d'apprentissage peuvent être utilisés pour la classification du texte, les plus utilisés sont : machine à vecteurs supports SVM, K-plus proches voisins K-NN, les réseaux bayésiens, les réseaux de neurones, Régression logistique, arbre de décision …etc.

L'objectif principal de cette phase est d'obtenir un modèle précis, efficace et adapté aux contraintes des données et des ressources disponibles.

1. Algorithme de les K-Plus Proches Voisins :

Le k-plus proches voisins, traduit en anglais par k-nearest neighbors (K-NN), est un algorithme dapprentissage supervisé. Il représente les textes dans un espace vectoriel, où chaque axe correspond à un élément textuel. Il ne comporte pas de phase dappre ntissage. Lidée de base consiste à prédire la classe dun texte ***t*** en fonction des ***k*** textes les plus proches voisins, déjà enregistrés et étiquetés dans lensemble dapprentissage, en utilisant une métrique de distance [8].

Voici l'algorithme de K-NN, illustré par la Figure 2, qui montre son principe de fonctionnement :

Algorithme 5 Algorithme de classification par k-PPV

Paramètre : le nombre k de voisins

Contexte : un échantillon de l textes classés en $C = c_1, c_2, ..., c_n$ classes

1: **pour** chaque texte t **faire**
2: transformer le texte t en vecteur $\mathbf{t} = (x_1, x_2, ..., x_m)$
3: déterminer les k plus proches textes du texte t selon une métrique de distance
4: combiner les classes de ces k exemples en une classe c
5: **fin pour**

Sortie : le texte t associé à la classe c.

Figure 2 : Algorithme de classification par K-NN [4].

La méthode K-NN utilise deux paramètres : le nombre k et la fonction de similarité. Une mesure est très utilisée est la similarité cosinus, qui consiste à quantifier la similarité entre deux documents en calculant le cosinus de l'angle entre leurs vecteurs [8] :

Mesure Cosinus :

$$cos(a,b) = \frac{\sum a \times b}{\sqrt{\sum a^2 + b^2}}$$

2. Machine à vecteurs supports SVM :

La machine à vecteurs supports (SVM) est un algorithme d'apprentissage supervisé appartenant à la catégorie des classificateurs linéaires, utilisé pour les problèmes de classification. Il a été introduit par Vapnik en 1995. Les SVM cherchent à déterminer une surface de décision pour séparer les points de l'ensemble d'apprentissage en deux classes. La décision prise repose sur les vecteurs de supports sélectionnés pour définir la frontière entre les classes [4].

Voici quelques notions de base des SVM d'après [9] :

- **Hyperplan :**

  Un hyperplan est une surface de séparation dans un espace à ***n*** dimensions qui permet de diviser les données den différents classe. Il est utilisé pour séparer de manière optimale les classes tout en maximisant la marge entre elles.

- **Vecteurs support :**

  Les vecteurs de support sont les points de données les plus proches de l'hyperplan. Ils déterminent la position et l'orientation de cet hyperplan. Ce sont ces points qui influencent la marge de séparation et jouent un rôle clé dans la construction du modèle SVM.

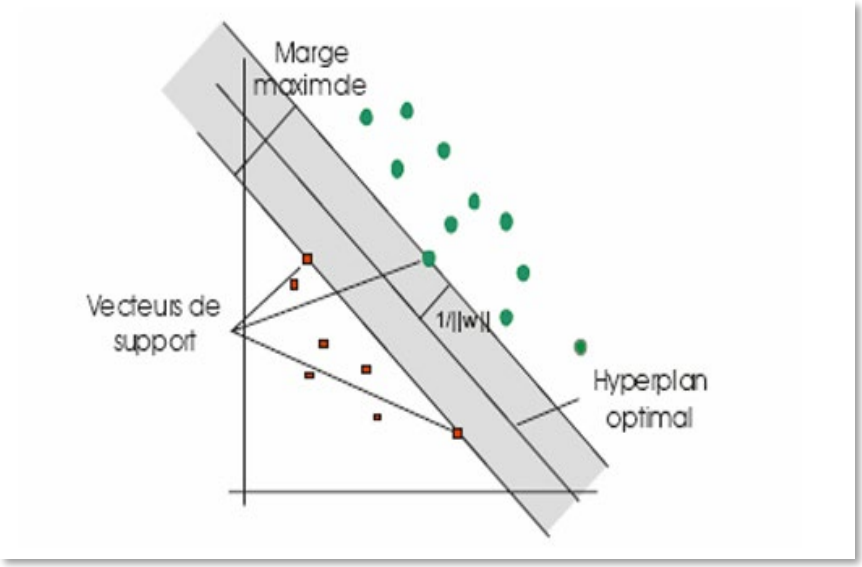


Figure 3 : Représentation de l'hyperplan séparateur optimal qui maximise la marge dans l'espace de rescription.

3. Naïve Bayes :

Le Naïve Bayes, également connu sous le nom de « Simple Bayes » ou « idiot's Bayes », est un algorithme d'apprentissage supervisé appartenant à la catégorie des classificateurs probabilistes, utilisé pour les problèmes de classification. Il a été formalisé par Duda et Hart en 1973 et popularisé dans le contexte d'apprentissage automatique par Langley en 1992 [10][11]. Les classificateurs Naïve Bayes cherchent à déterminer la classe la plus probable d'une instance en appliquant le théorème de Bayes avec une hypothèse d'indépendance conditionnelle entre les caractéristiques. La décision prise repose sur le calcul des probabilités a posteriori pour chaque classe possible [12].

Voici quelques notions de base de Naïve Bayes [13] :

- **Théorème de Bayes** :

  Le théorème de bayes constitue le fondement mathématique de l'algorithme. Il permet de calculer la probabilité a posteriori $P(C|X)$ d'une classe C étant donné un vecteur de caractéristiques X, en utilisant la formule :

$$P(C|X) = \frac{P(X|C) \times P(C)}{P(X)}$$

- **Hypothèse d'indépendance naïve** :

  L'hypothèse naïve suppose que toutes les caractéristiques sont conditionnellement indépendantes étant donnée la classe. Cette simplification permet de décomposer $P(X|C)$ en produit des probabilités individuelles :

$$P(C|X) = \prod P(x_i|C).$$

Remarque : Bien que rarement vérifiée en pratique, cette hypothèse rend l'algorithme computationnellement efficace et souvent performant.

4. Random Forest :

Le Random Forest est un algorithme d'apprentissage supervisé appartenant à la catégorie des méthodes d'ensemble par arbres, utilisé pour les tâches de classifications et régression. Il a été introduit par Leo Breiman en 2001. En classification, la méthode Random Forest combine plusieurs arbres de décision construits indépendamment sur des échantillons aléatoires des données d'entraînement pour reproduire une prédiction finale par vote majoritaire. La décision finale repose sur l'agrégation des prédictions individuelles de chaque arbre, permettant de réduire le sur-apprentissage et améliorer la généralisation [14].

Les principales notions sur lesquelles repose cette méthode sont les suivantes [14] :

- **Bagging (Bootstrap Aggregation)** :

  Le bagging est une technique qui consiste à créer plusieurs échantillons d'entraînements en effectuant un tirage aléatoire avec remise à partir de l'ensemble d'entraînement original. Chaque arbre de décision est entraîné sur un échantillon différent, ce qui introduit de la diversité dans la forêt et améliore la robustesse du modèle.

- **Sélection aléatoire de caractéristiques** :

  A chaque nœud de chaque arbre, seul un sous-ensemble aléatoire des caractéristiques disponibles est considéré pour effectuer la division. Cette randomisation supplémentaire permet de décorréler les arbres entre eux et d'éviter que certaines caractéristiques dominantes influencent tous les arbres de la même manière.

- **Agrégation des prédictions** :

  La prédiction finale du Random Forest est obtenue en combinant les prédictions de tous les arbres individuels. Pour la classification, c'est le vote majoritaire qui détermine la classe prédite, tandis que pour la régression, c'est la moyenne des prédictions qui est utilisée.

5. Réseaux de neurones :

Les réseaux de neurones artificiels s'inspirent du fonctionnement du cerveau humain, qui diffère totalement de celui d'un ordinateur, avec des nœuds de neurones interconnectés à la manière d'une toile [15]. Il s'agit d'une structure composée de plusieurs couches successives de nœuds, permettant d'appliquer une transformation non linéaire aux vecteurs d'entrée. Dans le cas de la classification, ces vecteurs sont constitués de mots pondérés selon leur importance et sont convertis en un vecteur de catégories. La disposition des neurones ainsi que le nombre de couche utilisées influencent directement la qualité de la classification. Cependant, par rapport aux autres méthodes de classification supervisée, les réseaux de neurones présentent l'inconvénient d'un coût d'apprentissage élevé, ce qui peut rendre leur entraînement plus long et plus exigeant en ressources.

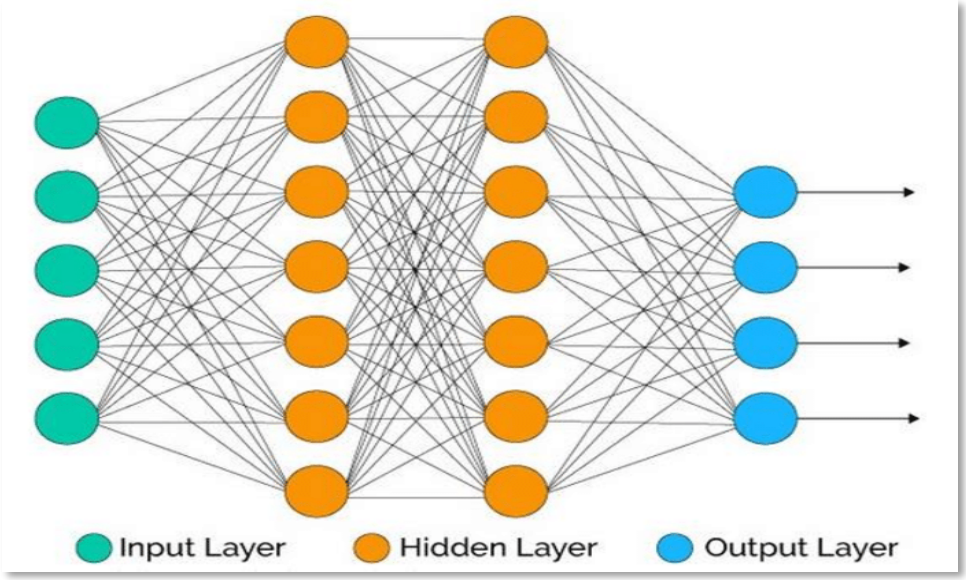


Figure 4 : Artificial Neural Network Architecture.

6. Arbre de décision :

Les arbres de décision sont une des techniques le plus populaires pour les problèmes de classification de données, grâce à leur simplicité de mise en œuvre et à l'interprétabilité des règles de décision issues de l'apprentissage. Leur objectif est de prédire automatiquement la valeur d'une **variable endogène** (à prédire) à partir de **variable exogènes** (prédictives). Cette méthode repose sur un partitionnement récursif des données, permettant ainsi d'affiner progressivement les décisions en fonction des caractéristiques des observations [4]. Cependant, ils sont sensibles au sur-apprentissage, ce qui peut les amener à s'adapter excessivement aux données d'entraînement et à perdre en capacité de généralisation sur les nouvelles données.

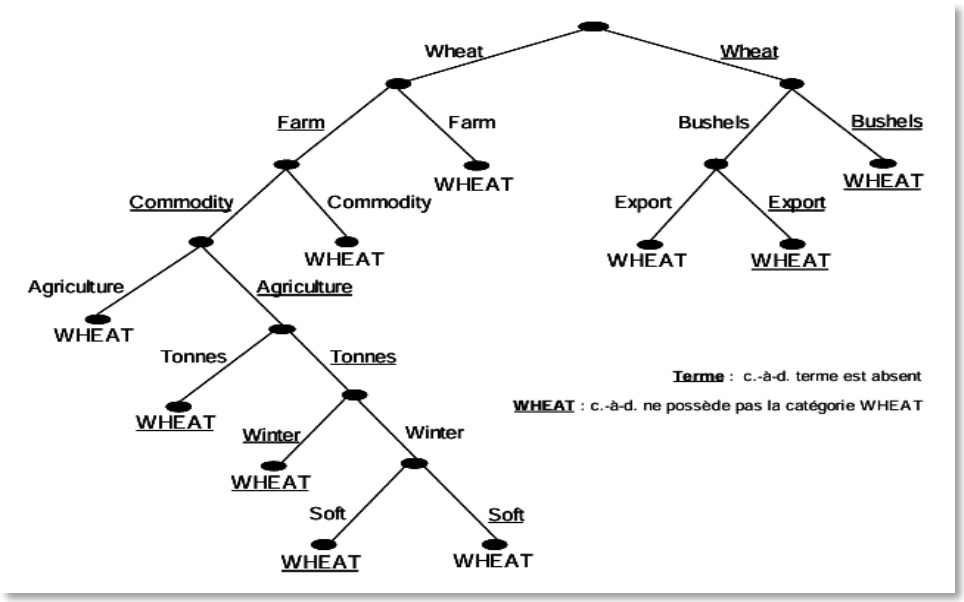


Figure 5 : Exemple d'arbre de décision appliqué sur le corpus Reuters [4].

1.3.3. Evaluation :

Une fois le modèle de classification créé, il est essentiel de le tester et de la valider avant son application réelle. Pour cela, on évalue ses performances en comparant les classes prédites par le modèle avec les classes réelle sur un ensemble de données connu.

Cette évaluation est généralement **empirique** plutôt qu'analytique, car elle repose sur l'observation des performances du modèle sur un corpus d'évaluation. En effet, la catégorisation **« vrai »** est par définition subjective : c'est l'association d'un texte libre à une catégorie ou classe, en fonction des informations que contient le corpus d'évaluation [4].

On peut distinguer 2 types de catégorisation :

- **Catégorisation binaire :** est un problème de classification où l'objectif est de déterminer si un document appartient ou non à une seule classe donnée. En d'autres termes, il existe deux classes possibles : la classe cible $\boldsymbol{C}$ et la classe opposée $\neg \boldsymbol{C}$.

- **Catégorisation multi-classes :** lorsque le nombre de classe $|\boldsymbol{C}| > 2$, une approche consiste à « couper » le processus de catégorisation en sous-problèmes de catégorisation binaire, ce qui peut parfois améliore la performance [4].

Pour mesurer la performance du classifieur, on utilise quatre valeurs importantes pour chaque classe :

- Vrai Positif (VP) : Documents attribués à leurs vraies catégories.
- Faux Positif (FP) : Documents attribués à des mauvais catégories.

- Faux Négatif (FN) : Le nombre de documents inconvenablement non attribués.
- Vrai Négatif (VN) : Le nombre de documents non attribués à une catégorie convenablement.

Classifieur		Expert	
		c_i	$\neg c_i$
	c_i	VP_i	FP_i
	$\neg c_i$	FN_i	TN_i

Figure 6 : Table de contingence [4].

- **Précision (*Precision*) :**

La précision est définie comme le rapport entre le nombre de vrais positifs et la somme des vrais positifs et des faux positifs. Elle mesure la proportion des prédictions positives qui sont réellement correctes [16]. Elle exprimée par la formule suivante :

$$\text{Précision (P)} = \frac{VP}{\text{VP} + \text{FP}}$$

- **Rappel (*Recall*) :**

Le rappel est défini comme le rapport entre le nombre d'exemples positifs correctement classés et le nombre total d'exemples positifs dans le jeu de données.

$$\text{Rappel (R)} = \frac{VP}{\text{VP} + \text{FN}}$$

- **F-mesure** :

C'est la combinaison de la précision et du rappel et leur pondération, elle est définie comme suit :

$$\text{F} - \text{mesure} = \frac{2PR}{P + R}$$

1.4. Les problèmes de la catégorisation de texte :

Malgré les progrès des algorithmes d'apprentissage automatique, la catégorisation de texte fait face à plusieurs défis qui impactent sa précision et son efficacité. Parmi les principales difficultés, on peut citer quelques-uns :

1.4.1. Redondance (Synonymie) :

La redondance et la synonymie permettent d'exprimer le même concept par des expressions différents, plusieurs façons d'exprimer la même chose. Cette difficulté est liée à la nature des documents traité exprimés en langage naturel contrairement aux données numériques.

Dans une représentation vectorielle d'un document, les termes sont traités individuellement, ce qui disperse les occurrences d'un même concept. Il est alors important de regrouper ces termes au sein d'un même ensemble sémantique.

Pour y remédier, la conception d'une ontologie permet de mieux définir les significations des termes. Toutefois, cette approche implique de coûts supplémentaires liés à son élaboration et à sa maintenance [17].

1.4.2. Polysémie (Ambiguïté) :

Contrairement aux données numériques, les données textuelles possèdent une richesse sémantique importante, car elles sont conçues et interprétées par la pensée humaine. A la différence des langages informatiques, le langage naturel tolère des entorses aux règles grammaticales, ce qui peut entraîner plusieurs interprétations d'un même énoncé. Un même mot peut ainsi avoir plusieurs significations selon le contexte dans lequel il est utilisé.

Cette polysémie rend parfois les mots peu fiables en tant que descripteurs, compliquant ainsi l'interprétation automatique du langage et posant des défis majeurs dans la tâche de traitement et de classification des textes [17].

1.4.3. Sur-apprentissage :

Le nombre de termes très important et très varié qui n'apparaissent pas de manière récurrente dans l'ensemble des textes entraînent de nombreux espaces vides dans le tableau de grande dimension (textes * termes). Cela peut favoriser le sur-apprentissage, un phénomène où le modèle apprend trop bien les données d'entraînement mais peine à généraliser sur de nouveaux textes. En conséquence, bien qu'il classe correctement les documents de la base d'apprentissage, il devient moins performant face à des textes inédits.

Pour limiter ce problème, il est essentiel de réduire la dimensionnalité en sélectionnant des termes pertinents. Idéalement, le nombre de textes doit être 50 à 100 fois supérieur à celui des termes. Cependant, en raison du faible nombre de textes disponibles, il est préférable de filtrer les termes tout en conservant ceux qui sont essentiels à la classification [3].

1.5. Domaine d'application :

Les méthodes classification présentent de nombreux avantages, mais leur fonctionnalité principale est de définir le contenu d'un document en seulement quelques mots. Ce classement facilite ensuite les recherches précises grâce à des filtres appliqués sur un ensemble de documents textuels. Voici autres applications possibles [17] :

- Le classement automatique de différents communiqués de presse, ou messages sur des forums en différentes matières (« les actualités de la région », « la bourse », « culture », etc…).
- Indexation automatique des documents sur des catégories d'index de bibliothèques.
- La gestion de bases documentaires (mémoire d'entreprise). Ce système permet d'organiser l'information en catégories thématiques, offrant ainsi à l'utilisateur une navigation plus fluide et intuitive.
- Les filtres internet en général, et en particulier les filtres anti-spams.
- L'analyse des sentiments et l'extraction d'opinions.

1.6. Conclusion :

En résumé, la classification des textes joue un rôle clé dans le traitement du langage naturel en permettant d'attribuer des étiquettes aux documents afin d'optimiser leur organisation et leur analyse. Cette approche repose sur diverses techniques, allants des modèles de machine Learning aux méthodes probabilistes et déductives, et s'adapte à de nombreuses applications comme l'analyse de sentiments ou la détection de spams. Grâce aux différentes méthodes d'évaluations des classificateurs, il est possible d'améliorer la précision et la fiabilité des modèles. Son utilisation croissante dans divers secteurs témoigne de son importance pour extraire des informations pertinentes et faciliter la prise de décision.

Chapitre 2 : Désambiguïsation lexicale

2.1. Introduction :

Le langage humain est intrinsèquement ambigu, ce qui signifie qu'un même mot peut revêtir plusieurs interprétations selon le contexte dans lequel il apparaît. Cette polysémie pose un défi majeur, comme le montrent les exemples suivants :

(a) She went to the bank to deposit money.
(b) The river overflowed its bank after the storm.

Dans ces deux phrases, le mot **bank** prend des significations totalement différentes : dans la première, il fait référence à un établissement financier, tandis que dans la seconde, il désigne la berge d'une rivière [18].

Bien que cette distinction semble triviale pour un humain, elle représente une problématique complexe pour une machine. Contrairement aux humains qui, interprètent naturellement les ambiguïtés du langage sans effort conscient, les systèmes informatiques doivent traiter des informations textuelles non structurées et les convertir en représentation formelles afin d'en extraire le sens sous-jacent. Cependant, cette tâche est entravée par plusieurs obstacles, notamment la polysémie des mots, les ambigüités liées à la structure syntaxiques des phrases, les variations dans les registres et styles linguistiques [18].

Ces difficultés rendent l'identification du sens d'un mot en contexte loin d'être triviale pour une machine. C'est précisément pour pallier ces limitations qu'intervient la désambiguïsation lexicale, ou Word Sense Disambiguation (WSD).

Dans ce chapitre allons aborder en détail ce concept, en définissant d'abord ses principes fondamentaux. Nous présenterons ensuite les ressources linguistiques nécessaire, puis les principales approches adoptées pour résoudre ce problème, avec leurs avantages et limites. Nous consacrerons également une section aux méthodes d'évaluation permettant de mesurer la performance des systèmes de désambiguïsation.

2.2. Définition :

La désambiguïsation lexicale, connue en anglais sous le terme Word Sense Disambiguation (WSD), est un problème classique dans le traitement automatique des langues (TAL), désigne le processus visant à attribuer le sens le plus probable à un mot donné dans un document, à partir d'un inventaire de sens prédéfini [19].

Il existe d'autres définitions de la (WSD) depuis son apparition, on peut citer les suivants :

Définition 01 : D'après Yarowsky, la WSD est formellement décrite comme une tache de classification, où, étant donné une entré x, correspondant à l'occurrence d'un mot dans un contexte donné, le système automatique doit prédire la class correcte de $y \in Y$, où Y représente l'ensemble de sens possible du mot.

Définition 02 : selon Navigli (2009), la WSD est la tâche qui consiste à déterminer automatiquement le sens approprié d'un mot en fonction du contexte dans lequel il apparaît.

Elle peut être formellement décrite comme une tâche de classification, où les sens des mots sont les classes, et chaque mot doit être associé à un ou plusieurs sens issus d'un dictionnaire lexicale (comme WordNet). Comme l'ensemble des sens possibles varie selon le mot, la WSD correspond en réalité à ***n*** tâche de classification distincte, où ***n*** représente la taille du lexique [18].

Navigli distingue deux grandes variantes :

WSD ciblée (lexical sample) : on désambiguïse un ensemble restreint de mots cibles, souvent un mot par phrase, avec des méthodes supervisées.

WSD à tous les mots (all-words WSD) : on désambiguïse tous les mots lexicaux d'un texte (noms, verbes, adjectifs, adverbes), ce qui exige des systèmes à couverture étendue.

2.3. Sources de connaissances externes :

La connaissance constitue un élément fondamental dans la désambiguïsation lexicale. Les sources de connaissances fournissent les données indispensables pour associer correctement les sens aux mots. Ces sources peuvent prendre des formes diverses : corpus textuels (bruts ou annotés), dictionnaires électroniques, thésaurus, glossaires, ontologies, etc.

On distingue deux catégories principales pour les ressources utilisées dans le domaine du WSD [18] :

2.3.1. Ressources structurées :

- **Bases de connaissances lexicales :**

  WordNet représente l'exemple le plus emblématique, organisé autour du concept de synsets (ensemble de synonymes). Comme l'a souligné Navigli (2009), WordNet est devenue une ressource quasi-indispensable dans la plupart des approches modernes de WSD, offrant une structure hiérarchique composée de plus de 155 000 mots organisés en environ 117 000 synsets.

- **Ontologie :**

  Ces structures formelles représentent les concepts d'un domaine et leurs relations. Elles permettent d'intégrer des connaissances expertes dans le processus de désambiguïsation, particulièrement efficaces pour les domaines spécialisés.

- **Thésaurus :**

  Ces ressources indiquent les relations lexicales entre les mots, comme la synonymie, l'antonymie, ainsi que d'autres types de liens sémantiques. Le Roget's International Thesaurus qui introduit par Roget (1911) est le plus utilisé en WSD, avec 250 000 entrées.

- **Dictionnaires électroniques (MRDs) :**

  Depuis les années 1980, ces dictionnaires en format numérique sont devenus essentiels en traitement automatique du langage naturel comme (Collins English Dictionary, Oxford Advanced Learner's Dictionary, etc).

2.3.2. Ressources non structurées :

- **Corpus annotés :**

  Un corpus annoté est une collection de textes où les mots ambigus sont étiquetés manuellement ou automatiquement avec leur sens correct, généralement référencés dans une base lexicale (comme WordNet ou BabelNet).

  Les corpus les plus utilisés :

  - SemCor (anglais, annoté avec WordNet).
  - Senseval/SemEval (multilingue, évaluations de WSD).
  - OMSTI (multilingue, aligné avec BabelNet).
  - DSO (anglais, annotations de sens fréquents).

- **Corpus bruts :**

  Grandes collections de textes non annotés utilisées notamment dans les approches non supervisées. Ces corpus permettent d'extraire de patrons contextuels et des co-occurrences qui servent d'indices pour la désambiguïsation.

- **Corpus parallèles :**

  Ensembles de textes avec leurs traductions dans différentes langues, exploitant le fait que la traduction d'un mot peut lever son ambigüité dans la langue source.

L'intégration de plusieurs sources de connaissances complémentaires constitue souvent la stratégie la plus efficace pour améliorer les performances des systèmes de WSD, chaque type offrant des perspective différentes sur le sens des mots.

2.4. Approches du WSD :

2.4.1. Basée sur des connaissances :

Les méthodes de désambiguïsation lexicale basée sur les connaissances exploitent des ressources lexicales manuellement construites, telles que des dictionnaires ou des bases de données sémantiques, pour identifier le sens approprié d'un mot dans son contexte. Contrairement aux approches supervisées, ces méthodes ne nécessitent pas de corpus annotés avec des étiquettes de sens. Cette famille de méthodes repose sur des ressources lexicales (comme WordNet et BabelNet), et on peut les classer selon trois grandes stratégies principales : basée sur les glosses et basée sur les graphes, basée sur les statistiques [19].

1. L'algorithme Lesk :

Parmi les approches de désambiguïsation lexicale fondées sur les glosses (ou définitions des sens dans un dictionnaire), l'algorithme Lesk est l'un des plus anciens et des plus influents.

Introduit par Michael E. Lesk en 1986, constitue l'une des approches les plus classiques en désambiguïsation lexicale. Il repose sur un principe simple mais efficace : le calcul de la similarité entre les mots du contexte et les définitions des sens dans un dictionnaire [19].

Au fil du temp, plusieurs versions de l'algorithme de Lesk ont été proposés afin de surmonter les limites du modèle original :

- **Lesk original :**

  Dans sa version originale, l'algorithme compare les définitions de tous les sens possibles des mots ambigus présents dans une phrase. Pour chaque mot à désambiguïser, il sélectionne le sens dont la définition partage le plus grand nombre de mots avec les définitions des sens possibles des autres mots de la phrase [20]. Cette approche part du principe que des mots apparaissant ensemble dans un texte sont susceptibles d'avoir des définitions partageant des termes communs.

- **Lesk simplifié :**

  La version simplifiée de l'algorithme, également appelée « simplified Lesk » proposé par Kilgarriff & Rosenzweig en 1998, cette version ne compare plus les définitions entre elles, mais confronte directement le contexte d'apparition du mot ambigu (les mots environnants) avec chacune des définitions de ses sens possibles. Cette adaptation réduit considérablement la complexité computationnelle tout en maintenant des performances acceptables [19]. Dans cette version, le calcul du chevauchement entre le contexte et les définitions des sens est un élément central. Ce calcul permet de déterminer le sens le plus probable, voici la formule classique du calcul du score de chevauchement pour un sens i [21] :

$$Score(sens_i) = |contexte \cap définition((sens_i)|$$

L'algorithme Lesk simplifié suit un processus précis. Dans ce qui suit, on cite ses principales étapes :

Pour chaque mot ambigu ***W*** dans le texte :

- Extraire son contexte ***C*** (fenêtre de ***n*** mots autours de ***W***).
- Prétraiter ***C*** (suppression des mots vides, lemmatisation, etc.).
- Pour chaque sens possible ***S_i*** de ***W*** dans le dictionnaire :
  - Extraire et prétraiter la définition ***D_i***.
  - Calculer le score de chevauchement.
- Sélectionner le sens ***S_best*** avec le score maximal.

Cette version simplifiée maintient l'essence du principe de Lesk tout en offrant des performances acceptables avec une complexité algorithmique bien moindre, ce qui explique sa popularité dans les applications pratiques de désambiguïsation lexicale.

- **Lesk étendu :**

  La version étendue, également appelé « extended Lesk » enrichit les définitions des sens avec des informations supplémentaires provenant du réseau sémantique. Comme le détaille Segonne (2021), cette version intègre non seulement les définitions des sens candidats, mais aussi celles des sens reliés sémantiquement (hyperonymes, synonymes, méronymes, etc.). Cette extension permet de pallier le problème de la parcimonie des définitions, souvent trop courtes pour permette des recoupements significatifs.

Malgré son ancienneté, l'algorithme de Lesk reste une référence importante dans le domaine de la désambiguïsation lexicale, tant pour sa simplicité conceptuelle que pour sa capacité à fonctionner sans nécessiter de vastes corpus d'entraînement annotés. Il constitue une souvent une Baseline solide pour évaluer les performances de nouvelles approches de désambiguïsations.

2. Sens le plus fréquent :

L'algorithme du sens le plus fréquent (Most Frequent Sense - MFS ou MFS baseline), est une méthode de base en WSD automatique. Cette approche statique consiste à assigner systématiquement à un mot polysémique son sens le plus fréquent dans un corpus de référence, indépendamment du contexte dans lequel il apparaît.

L'algorithme MFS repose sur l'observation empirique que dans la plupart des corpus, les mots polysémiques (les mots qui ont plusieurs sens) ont une distribution inégale de leur sens, avec généralement un sens dominant qui apparaît beaucoup plus fréquemment que les autres [22]. Il procède donc en deux étapes :

- **Phase d'apprentissage :** Calculer la fréquence de chaque sens pour chaque mot polysémique dans un corpus annoté.
- **Phase de prédiction :** Pour chaque occurrence d'un mot polysémique, assigner automatiquement le sens le plus fréquent observé lors de l'apprentissage.

Cet algorithme peut être implémenté soit en s'appuyant sur les fréquences de sens extraites d'un corpus manuellement annoté, soit en utilisant l'ordre de priorité des sens défini dans une base lexicale comme WordNet où les sens sont classés par fréquence d'usage.

3. Personalized PageRank :

Les ressources lexicales comme WordNet fournissent une structure de graphe essentielle pour les méthodes de WSD basées sur les connaissances. Wn organise les sens des mots (synsets) en un réseau sémantique où les nœuds sont des concepts et les arêtes sont des relations lexicales [23]. Le tableau 1 présente les principales relations dans Wn :

Tableau 1: Quelques relations dans WordNet [23]

Relation	Description	Exemple
Hypernym	Is a generalization of	"Vehicle" is a hypernym of "bicycle"
Hyponym	Is a kind of	"Bicycle" is a hyponym of "vehicle"
Meronym	Is part/member of	"Branch" is a meronym of "tree"
Holonym	Contains part	"Tree" is a holonym of "branch"

Parmi les méthodes exploitant ces relations sémantiques, le Presonalized PageRank (PPR) se distingue comme une adaptation de l'algorithme PageRank original de Google développé par

Larry Page et Sergey Brin 1998, et qui a été appliquer avec succès à la désambiguïsation lexicale. Contrairement au PageRank classique qui calcule l'importance globale des nœuds dans un graphe G, le PPR introduit une téléportation (ou saut aléatoire) biaisée vers un sous-ensemble de nœuds spécifiques plutôt que vers n'importe quel nœud de graphe, reflétant ainsi des préférences contextuelles [24]. Voici la formule de base l'algorithme de PPR :

$$\boldsymbol{Pr = cMPr + (1 - c)v}$$

Avec :

Pr : est le vecteur Personalized PageRank que nous cherchons.

c : est le facteur d'amortissement (typiquement 0.85).

M : est la matrice de transition du graphe (normalisée par le degré sortant).

v : est la probabilité de téléportation vers le nœud A selon une distribution personnalisée (vecteur de personnalisation).

Voici les étapes principales [25] de fonctionnement du PPR :

1. **Construction du graphe *G* :** Création d'un graphe où les nœuds sont les sens possibles des mots à désambiguïser et les mots du contexte. Les arêtes représentent les relations sémantiques entre ces éléments.

2. **Initialisation du vecteur de personnalisation *v* :** Création d'un vecteur qui donne un poids initial plus important aux mots du contexte (souvent appelé « bias » ou biais).

3. **Application de l'algorithme PPR :** L'exécution de l'algorithme itératif qui propage les scores d'importances à travers le graphe en tenant compte du vecteur de personnalisation. Le nombre d'itérations se situe généralement entre 20 et 30 itérations pour atteindre la convergence.

4. **Extraction des scores finaux :** Après la convergence, récupération des scores d'importance pour chaque sens candidat des mots ambigus.

5. **Sélection du sens :** Pour chaque mot ambigu, sélection du sens ayant obtenu le score le plus élevé.

La Figure 7 montre un exemple de trois sens du mot « coach », avec des liens vers des concepts associés [26] :

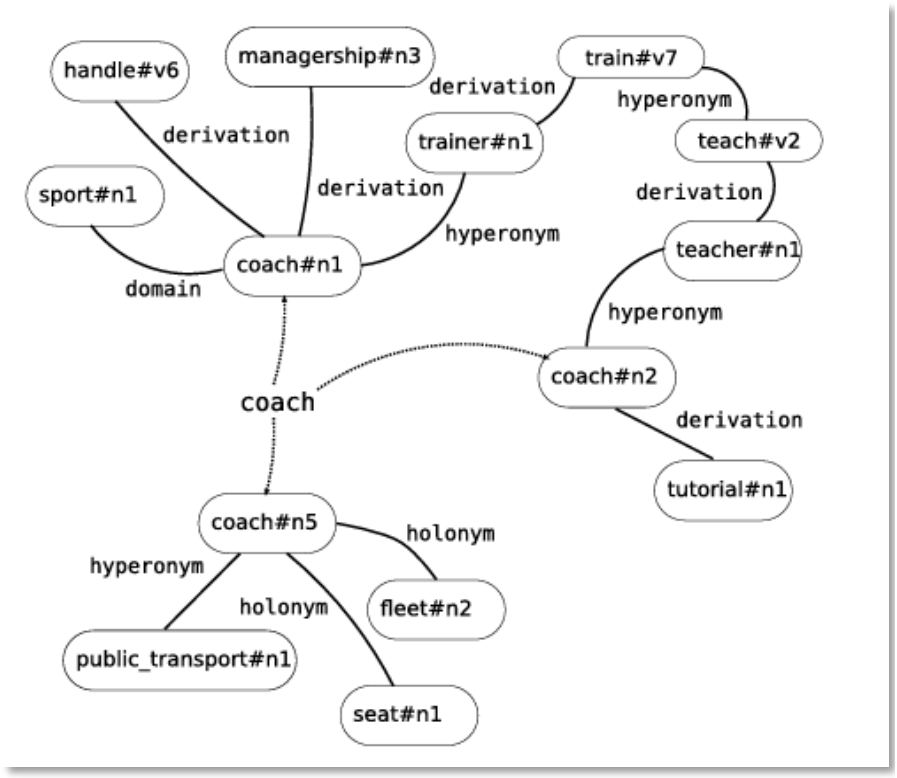


Figure 7 : Exemple de trois sens du mot « coach » [26].

2.4.2. Approches supervisées :

Les méthodes supervisées de désambiguïsation lexicale exploitent des corpus annotés manuellement où chaque occurrence de mot ambigu est étiquetée avec son sens correct. Ces approches traitent la WSD comme un problème de classification où les sens constituent les classes à prédire [18]. Parmi les méthodes supervisées, deux paradigmes dominent : les classifieurs statistiques traditionnels (SVM, Random Forest) et les architectures neuronales modernes basées sur les transformers.

1. Support Vector Machine (SVM) :

Le classifieur SVM (Support Vector Machine) est un algorithme d'apprentissage supervisé qui cherche à séparer les données en classes distinctes en maximisant la marge entre elles, grâce à un hyperplan optimal, éventuellement dans un espace transformé par un noyau pour traiter des problèmes non linéaire (voir le chapitre 1).

Le SVM, bien qu'initialement conçu comme un classifieur binaire, s'adapte à la WSD (un problème mutliclasse) en décomposant la tâche en plusieurs sous-problème binaires (approche one-vs-rest), ou chaque sens d'un mot est opposé aux autres, et le sens avec la confiance la plus élevée est sélectionné. Son efficacité repose sur la transformation des données textuelles (mots, contextes) en vecteur via des kernels (fonction de similarité), qui mesurent la similarité entre

exemples dans un espace de features optimisé. Cette capacité à projeter les données dans des espaces de haute dimension permet au SVM de capturer des motifs complexes dans les contextes linguistiques, expliquant des performances supérieures en WSD par rapport à d'autre méthodes supervisées [18].

2. Naïve Bayes :

Dans le contexte de la désambiguïsation lexicale WSD, un classifieur Naïve Bayes est un modèle probabiliste supervisé qui détermine le sens approprié d'un mot polysémique en appliquant le théorème de Bayes avec l'hypothèse s'indépendance conditionnelle entre les caractéristiques contextuelles [27]. Voici la formule de l'algorithme de Naïve Bayes [18] :

$$\hat{S} = \underset{S_i \in Senses_D(\omega}{\operatorname{argmax}} P(S_i | f_1, \dots, f_m) = \underset{S_i \in Senses_D(\omega)}{\operatorname{argmax}} \frac{P(f_1, \dots, f_m | S_i) P(S_i)}{P(f_1, \dots \dots, f_m)}$$

$$= \underset{S_i \in Senses_D(\omega)}{\operatorname{argmax}} P(S_i) \prod_{j=1}^{m} P(f_j | S_i)$$

Ce modèle calcule pour chaque sens possible ***Si*** la probabilité a posteriori (la probabilité d'un événement après avoir observé des données) ***P (Si | f1, …, fm)*** en multipliant la probabilité a priori ***P(Si)*** par le produit des probabilités conditionnelles ***P (fj | si)*** de chaque caractéristiques ***fj*** (comme les mots adjacents, les relations syntaxiques, etc.), puis sélectionne le sens $\hat{S}$ qui maximise cette probabilité.
Les paramètres sont estimés à partir des fréquences d'occurrences dans un corpus d'entraînement annoté, avec des techniques de lissage pour traiter les comptages nuls, et malgré sa simplicité et son hypothèse d'indépendance souvent irréaliste, cette approche s'avère efficace et compétitive avec d'autres méthodes supervisées pour la WSD, offrant un bon compromis entre performance et complexité computationnelle [18].

3. Bidirectional Encoder Representations from Transformers (BERT) :

BERT est un modèle de langage basé sur l'architecture Transformer qui révolutionne le traitement automatique du langage naturel en pré-entraînant des représentations bidirectionnelles profondes à partir de texte non-étiqueté par un conditionnement conjoint sur les contextes gauche et droit dans toutes les couches. BERT et ses dérivés dominent la plupart des benchmarks d'évaluation existants (test de référence standardisé pour l'évaluation), y compris ceux pour la désambiguïsation lexicale à grâce à leur capacité à capturer les nuances sémantiques sensibles au contexte [28].

Les analyses révèlent que BERT peut capturer avec précision les distinctions de de sens de haut niveau, même avec un nombre limité d'exemples disponibles pour chaque sens de mot, et que les modèles de langage arrivent presque à résoudre la désambiguïsation des noms lorsque les conditions d'entraînement sont optimales. Deux stratégies principales sont identifiées pour

utiliser BERT en WSD : le fine-tunning et l'extraction de caractéristiques, cette dernière approche étant plus robuste face au biais de sens et capable de mieux exploiter les données d'entraînement limitées disponibles, avec la simple stratégie d'extraction par moyenne des embeddings contextualisés (représentation vectorielle d'un mot selon le contexte) se révélant robuste même avec seulement trois phrases d'entraînement par sens de mot [28].

2.4.3. Approches non-supervisées :

Les approches non supervisées en WSD reposent sur le principe fondamental que des sens similaires d'un mot ambigu tendent à apparaître dans des contextes similaires, permettant qinsi d'induire automatiquement les sens à partir du texte brut par regroupement des occurrences selon des mesures de similarité contextuelle. Cette capacité d'induction ou de discrimination automatique des sens présente un avantage considérable pour contourner le goulot d'étranglement de l'acquisition manuelle de connaissances annotées, ouvrant la voie à des méthodes scalables et indépendantes de l'intervention humaine [29].

- **Clustering contextuelle :**

  Clustering contextuelle (Context Clustering) est une approche non supervisée de WSD qui représente chaque occurrence d'un mot cible dans un corpus par un vecteur contextuel construit à partir des mots environnants, puis applique des algorithmes de clustering pour regrouper ces vecteurs en clusters censé correspondre aux différents sens du mot ambigu. Cette méthode s'appuie sur l'hypothèse distributionnelle selon laquelle le profil de cooccurrence des mots exprime implicitement leur sémantique, utilisant des techniques comme lanalyse séma ntique latente (LSA) et la décomposition en valeurs singulières (SVD) pour réduire la dimensionnalité de lespace vectoriel et calculer la similarité contextuelle via des mesures géométriques comme le cosinus [30].

Bien que cette approche présente des défis liés au besoin de grandes quantités de données non étiquetées et à la correspondance potentielle imparfaite entre clusters et sens réels, elle offre lavantage de découvrir automatiquement les sens des mots sans s upervision manuelle.

Le domaine de la désambiguïsation lexicale ne se limite pas aux ces trois paradigmes classiques (méthodes fondées sur les connaissances, supervisées et non supervisées). Il bénéfice d'approches innovantes qui repoussent les frontières du traitement automatique du langage :

- **Méthodes semi-supervisées :** Optimisations des performances en exploitant à la fois des données annotées (rares) et non annotées (abondantes) comme le Co-training, Self-training.
- **Méthodes hybrides :** combinaison synergique de ressources lexicales structurées et de modèles statistiques neuronaux.

Cette pluralité méthodologique souligne à la fois la vitalité de la recherche et la complexité intrinsèque de la modélisation de l'ambiguïté lexicale.

2.5. Conclusion :

Dans ce chapitre, nous avons introduit la désambiguïsation lexicale (Word Sense Disambiguation – WSD), une étape essentielle pour améliorer la compréhension automatique du langage naturel. Après avoir défini les enjeux liés à l'ambiguïté sémantique des mots dans un contexte donné, nous avons présenté les principales approches développées pour y remédier.

L'accent a été mis sur les méthodes basées sur la connaissance, qui s'appuient sur des ressources lexicales structurées telles que WordNet. Ces approches, qui n'exigent pas de corpus annotés, se distinguent par leur capacité à exploiter la richesse des relations sémantiques entre les sens des mots pour effectuer la désambiguïsation. Leur intérêt réside notamment dans leur applicabilité à des domaines ou des langues pour lesquels les données annotées sont rares ou inexistantes.

Cette base théorique constitue un socle essentiel pour les expérimentations qui seront présentées dans le chapitre suivant. Celui-ci sera consacré à l'évaluation empirique de différentes méthodes de WSD appliquées à la classification de textes. L'objectif sera d'analyser dans quelle mesure la désambiguïsation lexicale peut améliorer les performances de la classification automatique, en comparant des résultats obtenus avec et sans WSD.

Chapitre 3 : Expérimentation

3.1. Introduction :

Le chapitre trois présente le cadre expérimental mis en place pour évaluer limpact des différentes méthodes de désambiguïsation lexical (Word Sense Disambiguation : WSD) sur les performances des systèmes de catégorisation automatique de textes. Faces à lambiguïté inhérente au langage naturel, où un même terme peut revêtir plusieurs significations selon son contexte, la désambiguïsation lexicale constitue un enjeu majeur pour améliorer la représentation sémantique des documents et par conséquent, la précision de leur classification.

Notre étude se concentre sur cinq approches de WSD aux caractéristiques complémentaires :

All senses, UKB (Unsupervised knowledge based), Most Frequent Sense (MFS), Simplified Lesk et une méthode hybride par vote.

Lobjectif principal de notre travail est de déterminer quantitativement dans quelles mesure s ces approche de désambiguïsation contribuent à améliorer les performances des systèmes de catégorisation textuelle par rapport aux représentations conceptuelles traditionnelles qui n'intègrent aucune désambiguïsation.

Le protocole expérimental que nous avons conçu sarticule autour dune architecture complète de traitement, depuis lanalyse linguistique des documents jusquà leur catégorisation, en passant par les étapes de représentation conceptuelle et de désambiguïsation sémantique. Cette architecture, dont la cohérence a été validée pour garantir une évaluation rigoureuse, intègre des composants modulaires permettant disoler et dévaluer spécifiquement la contribution des approches de WSD.

Pour structurer notre analyse, ce chapitre suit une progression méthodique. Nous commençant par présenter l'architecture du notre système, en détaillant le pipeline de traitement depuis les préprocesseurs jusqu'à la catégorisation, avec un accent sur le rôle central du module de désambiguïsation lexicale (WSD). Nous décrivons ensuite les corpus et les ressources utilisés, incluant les données textuelles, les environnements et les bibliothèques utilisées. La section expérimentation expose le protocole suivi, les métriques de performance (précision, rappel, F1-score) et les différentes configurations testées.

Une discussion critique permet d'analyser les résultats, en soulignant les forces et limites des approches ainsi que les biais potentiels.

Enfin une conclusion synthétise les contributions majeures de cette étude et esquisse des pistes et perspectives pour de futures recherches.

3.2. L’architecture du système :

Notre système dévaluation est conçu pour analyser et catégoriser automatiquement des textes en combinant des techniques avancées de Traitement Automatique du Langage (TAL) et d’apprentissage automatique. L’objectif est d’extraire des informations linguistiques et sémantiques pour classer les textes dans des catégories prédéfinies.

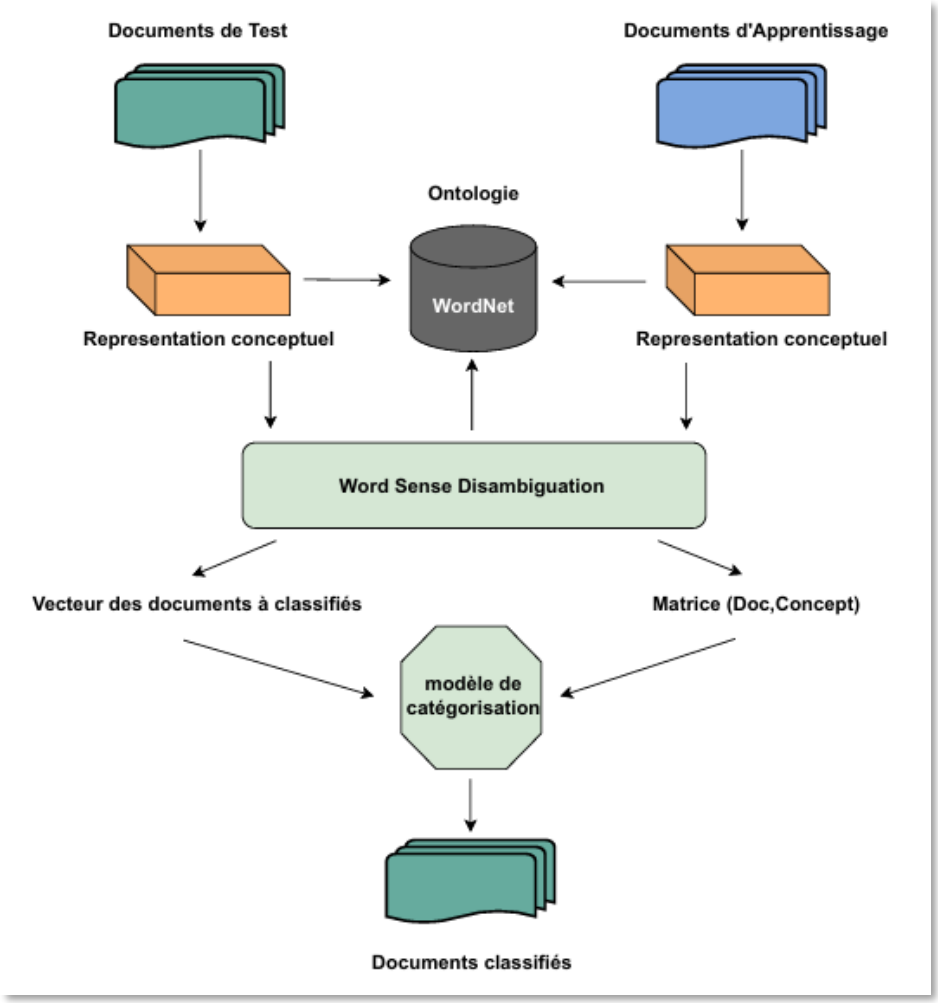


Figure 8 : L’architecture du système de classification avec WSD.

La Figure 8 représente les étapes principales du notre système de catégorisation, qui se divise en deux phases principales : l’apprentissage à partir de documents étiquetés, et l’évaluation à l’aide d’un ensemble de teste distinct.

Ce système repose d’abord sur une transformation préalable des documents, décrite dans la phase suivante.

3.2.1. Représentation conceptuelle :

Cette phase transforme les documents textuels bruts en une représentation sémantique structurée à travers plusieurs sous-étapes :

1. Prétraitement linguistique :

◊ **Nettoyage des balises et caractères HTML :**

Cette étape consiste à supprimer les balises HTML/XML (<p>, , <script>, etc.) et convertir les entités HTML (&amp ; , &lt ; ,   ; , etc.) en leurs caractères standards (&, <, espace, etc.).

◊ **Segmentation en phrases** :

Dans le domaine du traitement automatique du langage naturel, il existe plusieurs types de segmentation : la segmentation en mot (tokenisation), la segmentation en phrases, la segmentation thématique. Pour ce projet, nous avons choisi la segmentation en phrases car elle qui répond le mieux à nos besoins.

Cette étape nous permet de découper un texte continu en unités syntaxiques complètes appelées **segments** en identifiant les frontières entre les phrases à l'aide d'indices comme la ponctuation, les majuscules. Voici un exemple :

1. Texte brut: "Dr. Smith arrived at 5 p.m. He was excited! The meeting started shortly after."

2. Segmentation en phrases :
   - "Dr. Smith arrived at 5 p.m."
   - "He was excited!"
   - "The meeting started shortly after."

◊ **Tokenisation :**

Une fois que nous avons détecté les phrases, nous procédons au découpage de ces segments en unités minimales appelées **tokens**. Ces tokens peuvent être des mots, des signes de ponctuation, des nombres ou des symboles.

Cette étape de tokenisation nous permet de transformer le texte segmenté en phrases en une séquence d'éléments linguistiques élémentaires que nous pourrons ensuite traiter individuellement avec nos algorithmes du langage naturel.

◊ **Lemmatisation :**

Après la tokenisation, nous appliquons la lemmatisation qui consiste à réduire les tokens à leur forme canonique que nous appelons **lemme**. Nous procédons à cette réduction pour diminuer la variabilité lexicale en ramenant chaque mot à sa forme de base (ex : « said » -> « say»).

◊ **Etiquetage grammatical :**

Une fois la lemmatisation effectuée, nous assignons à chaque lemme **un tag grammatical** ainsi que ses propriétés morphologiques (genre, nombre, temps, etc.). Ces tags permettent de déterminer la catégorie grammaticale du lemme (nom, verbe, adjectif, adverbe), en s'appuyant sur des modèles statistiques combinés avec des règles linguistiques pour résoudre les ambiguïtés. Par exemple, le mot « Light » peut être interprété comme un nom ou un verbe selon le contexte.

Le tableau 2 illustre un exemple de l'étiquetage grammatical de la phrase « The cats eats the mouse. »

Tableau 2: Résultat de l'étiquetage morphosyntaxique d'un exemple.

Mot	**Étiquette**	**Propriétés morphologiques**	**Explication**
The	DT	<det> <def> <art>	Déterminant (article défini)
Cat	NN	<noun> <sg>	Nom commun singulier
eats	VBZ	<verb> <pres> <p3> <sg>	Verbe 3éme personne singulier en présent
The	DT	<det> <def> <art>	Déterminant
mouse	NN	<noun> <sg>	Nom commun singulier
.	Fp	<punct>	Ponctuation final

◊ **Analyse syntaxique :**

En nous appuyant sur les tags grammaticaux obtenus précédemment, nous analysons la structure grammaticale des phrases afin d'identifier les relations hiérarchiques entre les mots et construire une structure arborescente (ex : sujet → verbe → complément).

◊ **Elimination des Tags Non Pertinents :**

Cette étape consiste à filtrer ou ignorer les lemmes dont la catégorie grammaticale est considérée comme non pertinente pour notre analyse. Ces éléments n'apportent pas généralement de valeur informative significative et peuvent introduire du bruit dans la représentation finale des documents. Nous avons identifié les tags non pertinents que nous excluons de notre système :

- Les dates.
- Les nombres.
- Les ponctuations et les symboles spéciaux.
- Les espaces.

◊ **Elimination des mots vides :**

Dans cette étape, on élimine les lemmes grammaticalement fréquents mais sémantiquement pauvres c'at-à-dire les mots très présents dans les textes mais qui apportent peu ou pas d'information sémantique utile, tels que les articles, les pronoms et les prépositions. Ces mots sont communément appelés mots vide (ou stop words). Ce prétraitement permet de réduire le bruit linguistique tout en concentrant l'analyse sur les termes porteurs de sens.

Dans la Figure 9 nous montrons la liste des mots vides utilisées dans le prétraitement.

```
a about above according across actually adj after afterwards again against all almost alone
along already also although always among amongst an and another any anyhow anyone anything
anywhere are aren't around as at b be became because become becomes becoming been before
beforehand begin behind being below beside besides between beyond both but by c can can't
cannot caption co co. could couldn't d did didn't do does doesn't don't down during e each
eg eight eighty either else elsewhere end ending enough etc even ever every everyone everything
 everywhere except f few first for found from further g h had has hasn't have haven't he
he'd he'll he's hence her here here's hereafter hereby herein hereupon hers herself him himself
 his how however hundred i i'd i'll i'm i've ie if in inc. indeed instead into is isn't it it's
its itself j k l last later latter latterly least less let let's like likely ltd m made make
makes many maybe me meantime meanwhile might miss more moreover most mostly mr mrs much must
 my myself n namely neither never nevertheless next nine ninety no nobody none nonetheless noone
 nor not nothing now nowhere o of off often on once one one's only onto or other others
otherwise our ours ourselves out over overall own p per perhaps q r rather recent recently s
same seem seemed seeming seems seven several she she'd she'll she's should shouldn't since so
some somehow someone something sometime sometimes somewhere still such t taking than that
that'll that's that've the their them themselves then thence there there'd there'll there're
there's there've thereafter thereby therefore
```

Figure 9 : La liste des mots vides utilisées.

2. Mapping des mots en concepts et désambiguïsation :

Après avoir appliqué les étapes de prétraitement aux documents et les avoir représentés sous forme de vecteurs de mots étiquetés grammaticalement, nous passons à l'étape du mapping entre mots et concepts. Cette étape consiste à substituer chaque mot par le concept le plus approprié, transformant ainsi la représentation lexicale en une représentation conceptuelle. Le mapping repose sur la sélection du ou des concepts les plus probables pour chaque mot, selon l'approche de désambiguïsation lexicale (WSD) utilisée.

Dans notre système, nous avons implémenté cinq approches de WSD. Avant de les appliquer les mots (lemmes) ambigus à désambiguïser sont identifié en fonction de leur catégorie grammaticale (nom, verbe, adjectif ou adverbe). Si un mot ne correspond à aucun concept référencé dans la base lexicale, il est conservé dans sa forme originale, sans mapping.

Voici les principales étapes associées à chaque approche de WSD :

◊ **All Senses :**

« All Senses » est une approche de désambiguïsation lexicale qui ne sélectionne pas un seul concept (ou synsets dans le cas de WordNet) pour chaque mot ambigu, mais conserve lensemble des concepts possibles fournis par la base lexicale utilisée. Elle vise à capturer toute lambiguïté sémantique potentielle dun mot dans son contexte, plutôt que de forcer un e interprétation unique.Par exemple, en utilisant la base lexicale WordNet 3.0, le mot *mouse*,

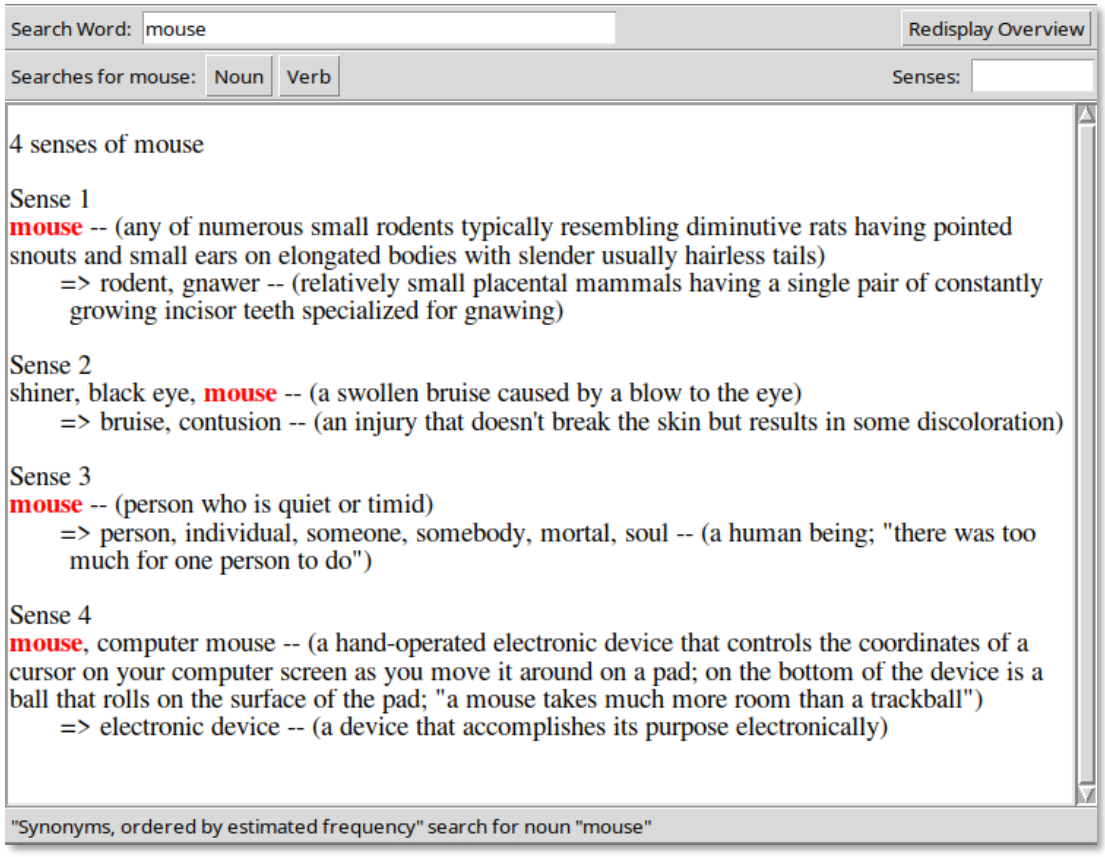


Figure 10 : capture décran de linterface WordNet pour le mot "mouse" .

lorsquil est identifié comme un nom commun (*noun*), est associé aux concepts suivants, classés par ordre de fréquence dutilisation (du plus couramment utilisé au moins fréquent), comme le montre la Figure 10 ci-dessous :

Cette illustration démontre la richesse des interprétations potentielles que lapproche « All Senses » prend en compte lors de lanalyse sémantique.

◊ **Most Frequent Sense (MFS) :**

Nous avons également implémenté l'algorithme MFS, qui assigne systématiquement à un mot le concept le plus couramment utilisé dans l'ontologie (décrit au chapitre 2).

Le résultat de cette approche de WSD est le plus restrictif parmi tous les approches que nous avons utilisées, car il ne prend pas en compte la richesse des concepts possibles et retourne systématiquement un seul concept prédéterminé pour chaque mot ambigu, indépendamment du contexte. En reprenant lexemple du mot *mouse* présenté dans la Figure 10, lalgorithme MFS sélectionnerait automatiquement le premier concept listé (le plus fréquent), sans considérer les autres concepts possibles affichés dans linterf ace WordNet.

◊ **Unsupervised Knowledgs-Based (UKB) [31] :**

Notre système intègre l'algorithme UKB, une approche de WSD non supervisé exploitant les connaissances sémantiques de WordNet. Cette approche applique une version adaptée de l'algorithme Personalized PageRank (décrit au chapitre 2) directement sur la structure relationnelle existante de WordNet, sans nécessiter la construction d'un nouveau graphe afin d'identifier les concepts les plus pertinents des mots en contexte. Cette méthode permet de capturer la cohérence sémantique globale en considérant les relations entre tous les termes du contexte, plutôt que de traiter chaque mot indépendamment.

Les principales fonctionnalités de UKB sont :

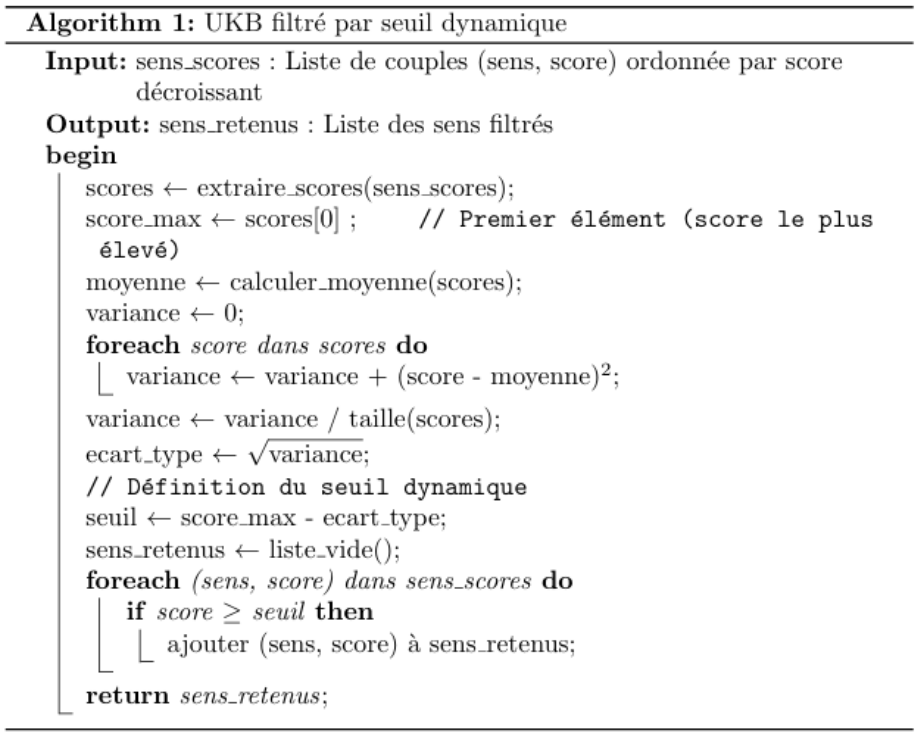
- **Désambiguïsation tous-mots** : UKB fournit une désambiguïsation sémantique de pointe pour tous les mots dune langue disposant dun WordNet disponible.

- **Indépendance linguistique** : Lalgorithme peut être facilement appliqué à nimporte quelle langue avec un wordnet.

- **Classement par PageRank** : Les sens sont ordonnés selon le PageRank assigné par lalgorithme.

- **Flexibilité** : UKB peut être alimenté avec nimporte quel dictionnaire de sens (concept) et graphe de relations.

La désambiguïsation dun mot ambigu avec UKB retourne une liste de concepts ordonnés par



Algorithm 1: UKB filtré par seuil dynamique

Input: sens_scores : Liste de couples (sens, score) ordonnée par score décroissant
Output: sens_retenus : Liste des sens filtrés
begin
scores ← extraire_scores(sens_scores);
score_max ← scores[0] ; `// Premier élément (score le plus élevé)`
moyenne ← calculer_moyenne(scores);
variance ← 0;
foreach *score dans scores* **do**
variance ← variance + (score - moyenne)^2;
variance ← variance / taille(scores);
ecart_type ← $\sqrt{\text{variance}}$;
`// Définition du seuil dynamique`
seuil ← score_max - ecart_type;
sens_retenus ← liste_vide();
foreach *(sens, score) dans sens_scores* **do**
if *score* $\geq$ *seuil* **then**
ajouter (sens, score) à sens_retenus;
return *sens_retenus*;

Figure 11 : Algorithme de filtrage des sens UKB.

score de pertinence décroissant. Dans notre système, nous avons adopté deux méthodes pour exploiter les résultats dUKB :

1. **UKB avec seuil dynamique** : nous utilisons lécart type de lensemble des scores comme seuil dynamique, permettant de sélectionner uniquement les concepts dont les scores sont proches de celui du sens majoritaire (celui ayant le score le plus élevé). Lalgorithme employé pour cett e approche est illustré dans la Figure 11.
2. **UKB sans seuil** : cette méthode sélectionne automatiquement le premier concept dans la liste retournée par UKB, cest -à-dire celui ayant le score de pertinence le plus élevé.

Lapproche avec seuil dynamique basée sur lécart type permet déviter de retenir tous les concepts possibles tout en conservant ceux ayant une probabilité raisonnable dêtre pertinents, minimisant ainsi le bruit produit par les concepts peu probables, et assurant une couverture optimale des concepts potentiellement valides. Lapproche sans seuil, quant à elle, offre une solution plus restrictive similaire à MFS mais basée sur les scores de pertinence contextuels dUKB.

◊ **Simplified Lesk :**

Notre implémentation inclut la version simplifiée de l'algorithme Lesk, qui compare les mots du contexte avec les définitions (glosses) des synsets WordNet (concepts), et choisit le concept dont le gloss partage le plus de mots communs en calculant le chevauchement lexical (voir le chapitre 2).

Cette méthode retourne un ensemble de concept ordonnés par un score de pertinence décroissant, nous sélectionnons donc le ou les concepts ayant obtenu le score maximal.

◊ **Approche Hybride :**

A partir des quatre approches, nous avons implémenté une méthode hybride par vote qui combine les résultats de l'algorithme UKB (avec un seuil dynamique) et de l'algorithme Lesk simplifiée. Le fonctionnement de cette approche repose sur un principe simple :

- Chaque concept proposé reçoit un vote de chaque algorithme qui le suggère.
- Seuls les concepts ayant reçus des votes des deux méthodes (intersection) sont retenus comme résultats final.
- Si aucune intersection n'est trouvée, cette approche utilise automatiquement les résultats de UKB comme solution de repli (méthode alternative lorsque la méthode principale échoue).

La Figure 12 présente l'algorithme de cette approche :

Algorithm 2: Approche Hybride par vote

```
Input: offsets produits par UKB, offsets produits par Lesk Simplifié
Output: Offsets sélectionnés par vote d'intersection
begin
    Extraire et normaliser les offsets de UKB et Lesk;
    Créer l'ensemble de tous les offsets uniques;
    intersection ← faux;
    foreach offset unique do
        votes ← 0;
        if offset présent dans UKB then
            votes ← votes + 1;
        if offset présent dans Lesk then
            votes ← votes + 1;
        if votes = 2 then
            intersection ← vrai;
            Sélectionner cet offset;
    if intersection = vrai then
        return Offsets ayant reçu 2 votes;
    else
        return Résultats de UKB;
```

Figure 12 : Algorithme de l'approche Hybride par vote.

Avant de sélectionner UKB comme solution de repli, nous avons effectué une petite comparaison. Sur la base de quelques critères, nous avons déterminer que cette méthode était la plus appropriée pour ce rôle.

Le Tableau 3 illustre que UKB présente une meilleure adaptabilité pour ce type de fonctionnement :

Tableau 3: Comparaison entre Lesk simplifiée et UKB.

Critère	**UKB**	**Lesk Simplifié**
Approche	Graphe de connaissance	Chevauchement lexical
Gestion synonymie	Excellente	Reste limité
Performance générale	Stable	Dépendante du contexte
Contexte lexicale	Large	Limitée aux définitions

3.2.2. Vectorisation et représentation des données :

Cette phase cruciale transforme les concepts désambiguïsés en structures de données exploitables par les algorithmes dapprentissage automatique :

◊ **Création du vecteur des documents :** Génération dune représentation vectorielle où chaque document est encodé comme un vecteur multidimensionnel dans un espace conceptuel.

◊ **Construction de la matrice Document-Concept :** Élaboration dune structure bidimensionnelle où les lignes représentent les documents, les colonnes représentent les concepts, et les valeurs encodent les relations pondérées entre eux. Cette matrice constitue lentrée principale des algorithmes de classi fication afin de faire lapprentissage.

3.2.3. Modèle de catégorisation :

Cette phase consiste à exploiter les représentations vectorielles désambiguïsées afin de construire un modèle de classification performant. Pour ce faire, on a utilisé plusieurs algorithmes d'apprentissage selon différentes configurations.

Avant l'entraînement des modèles, on a appliqué un filtrage TF-IDF aux données, permettant de transformer les fréquences brutes des termes en scores pondérés, reflétant leur importance relative dans l'ensemble du corpus. Les algorithmes utilisés sont les suivants (Voir chapitre 1) :

- ◊ **SVM (support vector machine) :** méthode à marge maximale, bien adaptée aux espaces vectoriels de grande dimension. Elle vise à optimiser la séparation entre les classes.

- ◊ **K-NN** : méthode par similarité locale testant différentes valeurs de k (nous avons utilisé k=5). Elle classe un document en fonction des catégories majoritaires parmi ses voisins les plus proches.

- ◊ **Naïve Bayes** : méthode probabiliste offrant de bonnes performances malgré sa simplicité, particulièrement efficace avec les représentations textuelles.

- ◊ **Random Forest** : méthode ensembliste basée sur une combinaison de plusieurs arbres de décision (dans notre système nous avons utilisé 150 arbres) pour une robustesse accrue et une meilleure généralisation.

3.2.4. Documents classifiés :

Contrairement aux systèmes de désambiguïsation lexicale qui sappuient traditionnellement sur des évaluations intrinsèques via des corpus annotés (SENSEVAL/SemEval), nous avons opté pour une approche dévaluation extrinsèque en mesurant directement limpac t des approches de WSD sur une tâche applicative réelle : la classification automatique de documents du corpus Reuters.

Cette méthodologie nous permet dobserver concrètement lamélioration des performances de classification apportée par les différentes approches de désambiguïsation, fournissant ainsi une mesure plus orientée vers l'application pratique que les évaluations traditionnelles.

Notre **processus dévaluation** sarticule autour de **la validation croisée à 10 plis** (10-fold cross validation) pour éviter les problèmes de surapprentissage entre les ensembles dentraînement et de test. Après lentraînement du modèle de classification, nous procédons à la prédiction des catégories sur les documents de test, puis nous évaluons les performances obtenues à laide de différents indicateurs de performance détaillés au chapitre 1. Cette démarche rigoureuse nous permet de quantifier précisément la contribution des techniques de WSD dans le contexte spécifique de la classification documentaire automatisée.

3.3. Corpus et outils de développement :

Notre cadre expérimental repose sur des ressources standardisées et des outils reconnus, afin de garantir la reproductibilité des résultats et de faciliter les comparaisons avec dautres travaux du domaine. Les éléments constitutifs de ce cadre sont détaillés ci-après :

3.3.1. Corpus utilisés :

- **Corpus Reuters-21578 :**

Le corpus Reuters-21578 est une collection de 21 578 dépêches financières, réparties en 90 catégories thématiques, rédigées en anglais et publiées par lagence Reuters au cours de lannée 1987. Ce jeu de données est largement adopté dans les domaines de la classification automatique de textes et de lapprentissage machine [32]. Cette ressource offre un ensemble riche, structuré, et annoté, où les documents sont associés à des catégories thématiques prédéfinies, facilitant ainsi les tâches d'évaluation et de comparaison des performances des systèmes.

La Figure 13 illustre un exemple d'un document du corpus :

```
TITRE: ALLEGHENY <AI> SUED OVER PROPOSED BUYOUT

TEXTE:
Allegheny International Inc said it and First Boston Inc's <FBC>
Sunter Holdings Corp subsidiary have been named as defendants in a
class action filed in the Court of Common Pleas for Allegheny
County, Pa., which seeks an injunction against Allegheny's proposed
merger into Sunter. The company said its board and some former
directors and First Boston were also named as defendants. It said
it and Sunter intend to vigorously oppose the action. Allegheny
said the class action suit alleges the price to be paid in the
transaction is grossly unfair. The company said the suit's
allegations are similar to those contained in an earlier federal
court suit. Reuter
```

Figure 13 : Exemple d'un document du corpus Reuters.

- **Corpus équilibré :**

La version précédente du corpus Reuters-21578 comporte 90 catégories, dont la distribution est fortement déséquilibrée. Certaines classes, telles que earn, acq, money-fx, sont très fréquentes, tandis que d'autres apparaissent rarement, ce qui pose des difficultés pour l'apprentissage, notamment en ce qui concerne la reconnaissance des classes rares.

Afin d'atténuer ce déséquilibre et de faciliter les comparaisons, on a basé uniquement sur les 10 classes les plus couramment utilisées.

Le Tableau 4 suivant présente leur distribution dans le corpus :

Tableau 4 : La distribution des classes majoritaires du corpus Reuters

Classe	Nombre de fichier	Pourcentage %
Earn	3926	42.36%
Acq	2427	26.18%
Moneyfx	777	8.38%
Trade	475	5.12%
Crude	456	4.92%

Classe	Nombre de fichier	Pourcentage %
interest	300	3.24%
ship	291	3.14%
corn	252	2.72%
wheat	207	2.23%
moneysupply	158	1.70%

Selon le Tableau 4, le déséquilibre entre les classes persiste, même lorsqu'on ne considère que les 10 classes les plus fréquentes. Cette situation engendre des problèmes lors de l'apprentissage, notamment un biais en faveur des classes majoritaires, le modèle ayant tendance à prédire ces dernières pour minimiser l'erreur globale.

Afin d'atténuer cet effet, nous avons effectué un **rééquilibrage manuel** de ce corpus. Cette opération consiste à uniformiser le nombre de documents entre **les 10 classes**, afin d'atteindre un même nombre de fichiers par classes, en se basant sur la classe la moins représentée (la **classe money-supply** qui a **158 fichiers**).

3.3.2. Bibliothèques utilisées :

- **JFreeling 4.2 :**

JFreeling est une bibliothèque open-source en java dédiée au traitement automatique du langage naturel (TALN). Cette bibliothèque offre un pipeline complet danalyse linguistique incluant :

- Tokenisation et segmentation en phrases.
- Étiquetage grammatical avec une précision de 97% sur langlais.
- Lemmatisation intégrant la gestion des formes irrégulières

- ❖ Analyseur de dépendances basé sur des grammaires statistiques.
- ❖ La désambiguïsation lexicale avec la méthode UKB.

Lavantage principal de JFreeling réside dans son intégration native avec WordNet, facilitant ainsi le mappage entre les tokens du texte et les concepts de lontologie [33][31].

- **DKPro Core [34] :**

DKPro est une bibliothèque java open-source et une infrastructure modulaire de traitement du langage naturel développée à lUniversité technique de Darmstadt, fournissant des composants réutilisables pour diverses tâches linguistiques, elle s'appuie sur le framework Apache UIMA.

Cette bibliothèque intègre un module de désambiguïsation lexicale (WSD) implémentant une version simplifiée de l'algorithme de Lesk, offrant les fonctionnalités clés suivantes :

- ❖ Compare les mots du contexte avec les définitions(glosses) de WordNet.
- ❖ Utilise des fonctions configurables de calcul de similarité textuelle.
- ❖ Permet des prétraitements personnalisables (lemmatisation, filtrage de mots vides).
- ❖ Sintègre facilement avec dautres composants danalyse linguistique.
- ❖ Offre une API standardisée pour lévaluation et la comparaison avec dautres algorithmes de WSD.

- **ExtJWNL [35] :**

ExtJWNL (extended Java WordNet Library) est une bibliothèque java open-source permettant d'interagir avec la bases lexicales WordNet de manière programmatique. C'est une version améliorée de JWNL. Elle sert d'interface complète pour charger, manipuler et interroger les données de WordNet.

- **Weka 3.8.6 [36] :**

Weka (Waikato Environnent for Knowledge Analysis) est une suite logicielle open-source en java, dédiée à l'apprentissage automatique, à l'exploration de données et au prétraitement des données. Développée à l'université de Waikato, elle offre une collection d'algorithmes prêts à l'emploi pour des tâches de classification, régression, etc. Elle est accessible via une interface graphique interactive ou une API java.

3.3.3. Ressources lexicales :

- **WordNet 3.0 :**

WordNet est une base de données lexicale pour langlais où les mots sont organisés en ensembles de synonymes (synsets) reliés par des relations sémantiques et lexicales, formant un réseau hiérarchique de concepts. Cette version, publiée en 2006, contient environ 155 000 mots organisés en 117 000 synsets avec 207 000 paires mot-sens [37][38].

Tableau 5 : Statistiques du WordNet 3.0 [38].

Partie du discours	Mots uniques	Synsets	Paires Mot-Sens
Noms	117,798	82,115	146,312
Verbes	11,529	13,767	25,047
Adjectifs	21,479	18,156	30,002
Adverbes	4,481	3,621	5,580

3.3.4. Environnement de développement :

- **NetbBeans Apache :**

Nous avons opté pour une implémentation en java, réalisée dans l'environnement de développement intégré (IDE) Apache NetBeans, version 21.

NetBeans Apache est un IDE open source maintenu par la Apache Software Foundation. Ce projet, initialement développé par Sun Microsystems puis racheté par Oracle, NetBeans a été transféré à la Apache Software Foundation en 2016 pour devenir un projet Apache de niveau supérieur.

Lenvironnement offre des fonctionnalités avancées pour le développement Java Enterprise Edition (JEE), avec un support natif pour les serveurs dapplications, la gestion des bases de données, et lintégration de frameworks populaires. NetBeans inclut également des outils de profilage de performance et da nalyse de code.

- **Langage JAVA :**

Pour le développement de ce projet, nous avons choisi d'utiliser le langage de programmation Java, principalement en raison des avantages suivants :

❖ **Robustesse et fiabilité** :

Vous pourriez ajouter que Java dispose dune gestion automatique de la mémoire et dun système de gestion des exceptions robuste qui contribuent à cette fiabilité.

❖ **Support de linformatique distribuée** :

Il serait intéressant de mentionner les APIs spécifiques comme RMI (Remote Method Invocation) ou les technologies Java EE qui facilitent le développement distribué.

❖ **Gestion du multithreading** :

Préciser que Java offre des mécanismes de synchronisation avancés et des classes utilitaires (comme java.util.concurrent) pour gérer efficacement la concurrence.

❖ **Programmation orientée objet** :

Vous pourriez souligner lencapsulation, lhéritage et le polymorphisme comme piliers facilitant la modularité.

3.4. Protocol expérimental :

Cette section présente les résultats détaillés de notre expérimentation comparative des approches de désambiguïsation du sens des mots (WSD) combinées aux quatre modèles de catégorisation. Pour chaque approche de WSD, nous analysons les performances obtenues avec les différents modèles de classification.

3.4.1. Ajustement des hyperparamètres :

Avant de commencer l'évaluation, nous avons configuré deux paramètres de filtrage et de sélection des caractéristiques afin de trouver un équilibre optimal entre la richesse sémantique du modèle et sa capacité de généralisation. Les paramètres sont les suivants :

- **setWordsToKeep** : Ce paramètre contrôle la dimensionnalité du modèle en limitant le nombre de termes distincts conservé dans la représentation. Il prend des valeurs comprises entre 1000 et 150000 mots.
- **setMinTermFreq** : Ce seuil définit la fréquence minimale qu'un terme doit atteindre dans le corpus pour être inclus dans le modèle. Il varie entre 2 et 15 occurrences.

Pour chaque jeu de données conceptuelles généré par les approches de WSD, nous avons testé toutes les combinaisons possibles entre les valeurs **[3000, 5000, 6000, 8000, 10000, 12000, 15000]** pour **setWordsToKeep** et **[3, 5, 6, 7, 8, 10]** pour **setMinTermFreq**. Cette approche nous a permis d'évaluer 42 configurations différentes pour chaque approche de WSD. Nous avons ensuite comparé les F-mesure obtenus (nous avons choisi la F-mesure car elle combine entre la précision et le rappel) afin d'identifier la combinaison de paramètres offrant les meilleures performances de classification.

Le tableau 6 présente pour chaque approche de WSD la meilleure combinaison de paramètre obtenue.

Tableau 6 : La meilleure combinaison de paramètre pour chaque jeu de donnée.

La méthode	All senses	UKB avec seuil	UKB sans seuil	Lesk simplifié	MFS	Hybride
Configuration	[3000, 10]	[3000, 3]	[3000,3]	[3000, 10]	[3000, 3]	[3000, 8]
F-mesure	86.89%	88.90%	88.78%	86.42%	88.71%	87.75%

3.4.2. Résultats et discussion :

Dans cette partie, nous avons analysé les résultats de l'évaluation des 4 modèle de catégorisation en les comparant avec les résultats de la représentation conceptuelle traditionnelle « All senses » afin d'observer s'il y a eu amélioration ou non. Enfin, nous avons effectué une comparaison globale entre les résultats des différents modèles.

1. Performances avec la classification SVM :

Les résultats ci-dessous (Tableau 7) présentent l'évaluation des approches de WSD appliqués à la classification avec l'algorithme SVM :

Tableau 7 : L'évaluation des approches de WSD avec la classification SVM.

	All Senses	**UKB avec seuil**	**UKB sans seuil**	**Lesk Simplifié**	**MFS**	**Hybride**
Précision	87.07%	89.02%	88.86%	86.51%	88.78%	87.82%
Rappel	86.84%	88.86%	88.73%	86.39%	88.67%	87.72%
F-mesure	86.89%	88.90%	88.78%	86.42%	88.71%	87.75%

Par rapport à la référence « All senses » qui a un F-mesure de 86.89%, la plupart des approches de WSD testés améliorent les performances de classification, à l'exception de Lesk simplifié. UKB avec seuil obtient les meilleurs résultats (+2.01%), suivi UKB sans seuil (+1.89%) et de MFS (+1.82%) et de l'approche Hybride (+0.86%). Seul Lesk simplifié affiche une légère baisse des performances (-0.47%).

2. Performances avec la classification K-NN :

Voici les résultats (Tableau 8) de l'évaluation de la classification K-plus proche voisin (K-NN) :

Tableau 8 : L'évaluation des approches de WSD avec la classification K-NN.

	All Senses	UKB avec seuil	UKB sans seuil	Lesk Simplifié	MFS	Hybride
Précision	65.71%	73.85%	72.38%	75.67%	72.65%	73.51%
Rappel	59.94%	61.08%	59.05%	60.57%	60.63%	62.91%
F-mesure	59.62%	60.40%	58.04%	60.88%	59.66%	62.33%

Par rapport « All senses » qui a un F-mesure de 59.62%, toutes les approches de WSD améliorent les performances de classification K-NN, à l'exception de UKB sans seuil.

L'approche hybrides obtient les meilleurs résultats avec un gain significatif de (+2.71%), suivi de Lesk simplifié qui présente une amélioration de (+1.26%) et de UKB avec seuil avec un gain de (+0.78%). La méthode MFS affiche une très légère amélioration de seulement (+0.04%), restant pratiquement au même niveau de performance que la référence « All senses ». Seulement l'approche UKB sans seuil affiche une légère baisse des performances (-1.58%).

3. Performances avec la classification Naïve Bayes :

Voici les résultats (Tableau 9) de l'évaluation de la classification Naïve Bayes :

Tableau 9 : L'évaluation des approches de WSD avec la classification Naïve Bayes.

	All Senses	UKB avec seuil	UKB sans seuil	Lesk Simplifié	MFS	Hybride
Précision	73.07%	80.01%	81.42%	81.04%	80.41%	80.18%
Rappel	71.27%	79.75%	81.14%	80.95%	80.06%	79.81%
F-mesure	71.39%	79.75%	81.15%	80.88%	80.02%	79.84%

Par rapport à la référence « All Senses » qui a un F-mesure de 71,39%, toutes les approches de WSD testées améliorent significativement les performances de classification. UKB sans seuil obtient les meilleurs résultats avec un gain remarquable (+9.76%), suivi Lesk Simplifié avec un gain important de (+9,49%), suivi de près par MFS qui présente une amélioration substantielle de (+8,63%) et dUKB avec seuil avec un gain important de (+8,36%). L'approche Hybride affiche également une amélioration notable de (+8,45%).

4. Performances avec la classification Random Forest :

Voici les résultats (Tableau 10) de l'évaluation de la classification Random Forest :

Tableau 10 : L'évaluation des méthodes de WSD avec la classification Random Forest.

	All Senses	UKB avec seuil	UKB sans seuil	Lesk Simplifié	MFS	Hybride
Précision	85.64%	88.48%	88.63%	88.15%	88.09%	88.85%
Rappel	85.37%	88.04%	88.35%	87.72%	87.78%	88.61%
F-mesure	85.37%	88.04%	88.35%	87.70%	87.78%	88.59%

Par rapport à la méthode de référence « All Senses » qui obtient un F1-score de 85,37%, toutes les méthodes de WSD testées améliorent significativement les performances de classification avec Random Forest. L'approche Hybride obtient les meilleurs résultats avec un gain remarquable de (+3,22%), suivie de près par UKB sans seuil qui présente une amélioration substantielle de (+2.98%), suivi de près UKB avec seuil avec un gain de (+2,67%) et MFS avec un gain de (+2,41%). Lesk Simplifié affiche également une amélioration notable de (+2,33%).

5. Comparaison globale entre les méthodes de WSD :

Dans cette partie, nous évaluons les performances globales de chaque approche de WSD en calculant la moyenne des F-mesures obtenus avec l'ensemble des algorithmes de classification, afin d'avoir une vision d'ensemble des performances globales de chaque approche.

Pour chaque approche de WSD, nous avons calculé la moyenne selon la formule :

$$(\boldsymbol{F-mesure\_SVM + F-mesure\_KNN + F-mesure\_NB + F-mesure\_RF})/\boldsymbol{4}$$

Les résultats de l'analyse comparative sont présentés dans le tableau ci-dessous :

Tableau 11 : Evaluation globale.

Méthode	All Senses	UKB avec seuil	UKB sans seuil	Lesk Simplifié	MFS	Hybride
F-mesure moyenne	75.82%	79.27%	79.04	78.97%	79.04%	79.63%

Cette évaluation révèle des performances distinctes entre les différentes approches de WSD :

- L'approche **Hybride (par vote)** se distingue comme la plus performante avec un score moyen de 79.63%, démontrant lefficacité de la combinaison entre UKB et Lesk.

- **UKB avec seuil** suit de très près avec 79.27%, confirmant la robustesse des approches basées sur les graphes sémantiques et représentant un avancement potentiel pour les recherches sur UKB.

- **UKB sans seuil** et **MFS** obtiennent des scores identiques de 79.04%, ce qui confirme que ces deux méthodes produisent des résultats équivalents car elles sélectionnent toutes deux systématiquement un seul concept par mot ambigu.

- **Lesk Simplifié** obtient un score respectable de 78.97%, montrant des performances très proches des approches à concept unique.

- **All Senses** présente logiquement le score le plus faible avec 75.82%, servant de référence de base pour l'évaluation.

La différence entre la méthode la plus performante **« Hybride »** et la référence **« All senses »** est de (+**3.81%)**, ce qui représente une amélioration relative significative.

Ces résultats soulignent que toutes les approches de désambiguïsation lexicale apportent une valeur ajoutée par rapport à l'approche traditionnelle « All senses », avec un avantage particulier pour les approches hybrides et basées sur les graphes sémantiques qui tirent de structures de connaissance plus riches.

6. Sélection de la méthode la plus optimale pour chaque classifieur :

Bien quil soit difficile de déduire l 'approche la plus optimale de désambiguïsation pour chaque classificateur en se basant sur des expérimentations menées sur un seul corpus, cette analyse exploratoire des performances en variant les méthodes de désambiguïsation permet néanmoins didentifier des tenda nces comportementales significatives qui enrichissent la

compréhension théorique des interactions WSD-classification. Les résultats de cette observation sont illustrés dans le tableau 12.

Tableau 12 : La méthode de WSD la plus optimale pour chaque classifieur.

Méthode de classification	Méthode de WSD plus adapté	F-mesure
SVM	UKB avec seuil	**88.90%**
K-NN	Hybride	**62.33%**
Naïve Bayes	UKB sans seuil	**81.15%**
Random Forest	Hybride	**88.59%**

Après une analyse de ce tableau, laquelle met en évidence des écarts notables selon les différentes combinaisons, nous constatons que le classificateur **SVM** atteint sa meilleure performance **(88,90 %) avec lapproche UKB avec seuil** , ce qui témoigne de sa compatibilité avec les méthodes de désambiguïsation structurées. De son côté, **Random Forest** se distingue **avec lapproche Hybride (88,59 %),** obtenant des résultats similaires à SVM bien quutilisant une stratégie différente. Pour Naïve Bayes, cest lapproche UKB sans seuil qui donne les meilleurs résultats (81,15 %), ce qui laisse penser que ce classificateur sadapte mieux aux méthodes probabilistes moins contraignantes. Quant à K-NN, il affiche sa performance la plus élevée (62,33 %) également avec lapproch e Hybride, bien que ses scores restent globalement inférieurs aux autres classificateurs.

Ces résultats traduisent la sensibilité des performances à linteraction spécifique entre chaque algorithme de classification et la méthode de désambiguïsation employée. Cette analyse révèle un constat fondamental : lefficacité des algorithmes de classifi cation ne résulte pas seulement de la sophistication et de la complexité de la méthode de WSD choisie mais de leur compatibilité avec les propriétés des classificateurs.

Cette observation montre lexistence de **couplages algorithmiques optimaux.**

3.5. Conclusion :

En conclusion, cette étude expérimentale a permis d'évaluer l'impact des différentes approches de Désambiguïsation lexicale sur les performances de catégorisation de textes. Les résultats obtenus démontrent que l'intégration de techniques de WSD influence significativement la qualité de la classification automatique, chaque approche présentant des caractéristiques distinctes en termes de précision et de couverture.

Cette expérimentation révèle l'importance du choix méthodologique dans le traitement de l'ambiguïté lexicale et met en évidence les défis inhérents à la désambiguïsation dans des contextes applicatif réels. Les observations tirées de cette évaluation comparative constituent un apport significatif pour l'amélioration des systèmes de traitement automatique du langage naturel et ouvrent des perspectives prometteuses pour de futures recherches dans ce domaine.

Conclusion générale

Ce travail visait à évaluer l'effet des approches de désambiguïsation lexicale sur les performances des systèmes de catégorisation automatique de textes, à travers une analyse comparative fondée sur des métriques d'évaluation standardisées.

Létude approfondie des processus de catégorisation de texte présentée dans le premier chapitre a permis de saisir les enjeux et les défis inhérents à cette tâche fondamentale du traitement automatique du langage naturel. La compréhension des différentes étapes du processus de classification, depuis le prétraitement jusquà lévaluation finale, a constitué un socle théorique essentiel pour appréhender les problématiques liées à lambiguïté lexicale.

Lexploration des approches de la WSD dans le deuxième chapitre a révélé la diversité des approches disponibles, chacune présentant ses propres avantages et limitations. Cette analyse théorique a mis en évidence la complexité du problème de la polysémie et limportance de disposer doutils ef ficaces pour résoudre lambiguïté sémantique dans les textes.

Lévaluation empirique menée dans le troisième chapitre constitue le cœur de notre contribution. Les résultats obtenus apportent un éclairage quantitatif sur lefficacité relative des différentes approches de désambiguïsation lorsquelles sont intégrées aux systèmes de catégorisation. Cette analyse comparative a permis didentifier les approches les plus performantes et de hiérarchiser les méthodes selon leur capacité à améliorer la précision de classification.

Cependant, ce travail présente certaines limitations qu'il convient de reconnaître. L'évaluation s'est concentrée sur des métriques spécifiques et un corpus particulier, ce qui peut limiter la généralisation des résultats à d'autres contextes applicatifs.

Ces limitations ouvrent naturellement sur plusieurs perspectives de recherches futures :

- Evaluer l'efficacité des méthodes de WSD sur des corpus spécialisés (médical, juridique, technique) pour valider la généralisation des résultats obtenus.
- Développer et tester des métriques plus fines qui prendraient en compte la complexité sémantique et contextuelle des textes traités.
- Incorporer les méthodes de WSD basée sur les modèles de langage pré-entraînés (BERT, GPT) et évaluer leur impact sur la catégorisation.
- Tester la stabilité des performances des différentes approches face à des variations dans la qualité des textes d'entrée (fautes d'orthographe, multilinguisme).

En définitif, cette recherche apporte une contribution significative au domaine du traitement automatique du langage naturel en évaluant de manière systématique l'efficacité de la WSD dans la catégorisation de texte. Les résultats obtenus offrent des éléments concrets pour guider les choix méthodologiques et orienter les recherches futures dans ce domaine.

Références bibliographiques

1. Références Web

[1] Kabla, H. (2025, 25 juin). *Combien y a-t-il de pages indexées par Google ?*. https://www.hervekabla.com/wordpress/combien-y-a-t-il-de-pages-indexes-par-google/

2] FasterCapital. (2025, 25 juin). *Explosion des données : alimenter une croissance exponentielle à l'ère numérique*. https://fastercapital.com/fr/contenu/Explosion-des-donnees---alimenter-une-croissance-exponentielle-a-l-ere-numerique.html

[6] Blent. (2025, 25 juin). *Word Embedding & NLP : définition, exemples*. https://blent.ai/blog/a/word-embedding-nlp-definition

[17] *Support de cours – Apprentissage automatique*. (2025, 25 juin). https://www.mcours.net/cours/pdf/leilclic3/leilclic929.pdf

[27] *Supervised WSD Approaches – Word Sense Disambiguation*. (2025, 25 juin). https://1library.net/article/supervised-wsd-approaches-word-sense-disambiguation.y9g2x1vq

[29] TutorialsPoint. (2025, 25 juin). *Natural Language Processing - Word Sense Disambiguation*. https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_word_sense_disambiguation.htm

[33] *FreeLing NLP suite – Universitat Politècnica de Catalunya*. (2025, 25 juin). https://nlp.lsi.upc.edu/freeling/

[35] *Extended Java WordNet Library (extJWNL)*. (2025, 25 juin). https://sourceforge.net/projects/extjwnl/

[3] *WordNet Statistics – Princeton University*. (2025, 25 juin). https://wordnet.princeton.edu/documentation/wnstats7wn

2. Articles scientifiques, ouvrages et thèses

[3] Sebastiani, F. (2002). *Machine Learning in Automated Text Categorization*. *ACM Computing Surveys*.

[4] Jalam, R. (2003, 04 juin). *Apprentissage Automatique et Catégorisation de Textes Multilingues* (Thèse de doctorat). Université Lumière Lyon 2.

[5] Stricker, M. (2004, 23 avril). *Réseau de neurones pour le traitement automatique du langage : conception et réalisation de filtres d'informations* (Thèse de doctorat). Université de Paris.

[7] Bentaallah, M. A., & Malki, M. (2007). *WordNet-based Multilingual Text Categorization*. EEDIS Laboratory, Université de Sidi Bel Abbès.

[8] Barigou, F., Atmani, B., Bouziane, Y., & Barigo, N. (2020). *Accélération de la méthode de K plus proches voisins pour la catégorisation de textes*. Université d'Oran.

[9] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer.

[10] Duda, R. O., & Hart, P. E. (1973). *Pattern Classification and Scene Analysis*.

[11] Langley, P., Iba, W., & Thompson, K. (1992). *An Analysis of Bayesian Classifiers*.

[12] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.

[13] Hand, D. J., & Yu, K. (2001). *Idiot's Bayes—Not So Stupid After All?*

[14] Breiman, L. (2001). *Random Forests*. *Machine Learning*.

[15] Cherfi, S. (2020). *Détection d'intrusions via des réseaux de neurones optimisés par des métaheuristiques*. Thèse, Université de Jijel.

[16] Kohavi, R. (1995). *A Study of Cross-validation and Bootstraps for Accuracy Estimation and Model Selection*. IJCAI.

[18] Navigli, R. (2009). *Word Sense Disambiguation: A Survey*. *ACM Computing Surveys*, 41(2), Article 10.

[19] Segonne, V. (2021). *Désambiguïsation lexicale automatique des verbes du français*. Thèse, Université Paris Cité.

[20] Lesk, M. (1986). *Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone*. SIGDOC.

[21] Kilgarriff, A., & Rosenzweig, J. (2000). *Framework and Result for English SENSEVAL*.

[22] Yarowsky, D. (1995). *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods*.

[23] Fellbaum, C. (1998). *WordNet – An Electronic Lexical Database*. MIT Press.

[24] Agirre, E., & Soroa, A. (2009). *Personalizing PageRank for Word Sense Disambiguation*. EACL.

[25] Agirre, E., Soroa, A., & Stevenson, M. (2010). *Graph-based Word Sense Disambiguation using Random Walks*.

[26] Agirre, E., Lopez de Lacalle, O., & Soroa, A. (2014). *Random Walks for Knowledge-based Word Sense Disambiguation*.

[28] Loureiro, D., & Rezaee, L. (2021). *Analysis and Evaluation of Language Models for Word Sense Disambiguation*.

[30] Purandare, A., & Pederson, T. *Word Sense Discrimination by Clustering Contexts in Vector and Similarity Spaces*.

[31] Padró, L., Reese, S., Agirre, E., & Soroa, A. (2024). *Semantic Services in Freeling 2.1: WordNet and UKB*. TALP, ISAE-Supaero, IXA NLP Group.

[32] Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). *RCV1: A New Benchmark Collection for Text Categorization Research*. *Journal of Machine Learning Research*, 5, 361–397.

[34] Miller, T., & Gurevych, I. (2014). *DKPro WSD: A Generalized UIMA-based Framework for Word Sense Disambiguation*. *ACL*.

[36] Frank, E., Hall, M. A., & Witten, I. H. (2016). *The WEKA Workbench. Online Appendix for Data Mining: Practical Machine Learning Tools and Techniques*.

[37] Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press. | Miller, G. A. et al. (2006). *WordNet: A Lexical Database for the English Language*.

Abstract

Automatic text classification is a fundamental task in natural language processing, often confronted with the challenge of lexical ambiguity in polysemous words. This research examines the impact of Word Sense Disambiguation (WSD) methods on classification performance by evaluating five WSD approaches applied to four distinct classification algorithms. The study demonstrates that there is no universally optimal WSD method, but each classification algorithm benefits from a specific WSD approach. Experimental results show significant performance variations according to the tested combinations, highlighting the importance of adequacy between disambiguation strategy and learning algorithm. This research contributes to a better understanding of the interaction between semantic ambiguity resolution and automatic classification, offering perspectives for optimizing text processing systems. This study suggests the necessity of an adaptive approach in choosing WSD methods according to the specific classificatory context.

الملخص

تُعتبر تصنيف النصوص التلقائي مهمة أساسية في معالجة اللغة الطبيعية، والتي تواجه غالباً تحدي الغموض المعجمي في الكلمات متعددة المعاني. يدرس هذا البحث تأثير طرق إزالة الغموض المعجمي على أداء التصنيف من خلال تقييم خمس مناهج لإزالة الغموض المعجمي مطبقة على أربع خوارزميات تصنيف مختلفة. تُظهر الدراسة أنه لا توجد طريقة مثلى عالمياً لإزالة الغموض المعجمي، بل تستفيد كل خوارزمية تصنيف من نهج محدد لإزالة الغموض المعجمي. تُظهر النتائج التجريبية تباينات كبيرة في الأداء حسب التركيبات المختبرة، مما يؤكد أهمية التوافق بين استراتيجية إزالة الغموض وخوارزمية التعلم. يساهم هذا البحث في فهم أفضل للتفاعل بين حل الغموض الدلالي والتصنيف التلقائي، مما يوفر آفاقاً لتحسين أنظمة معالجة النصوص. تقترح هذه الدراسة ضرورة اتباع نهج تكيفي في اختيار طرق إزالة الغموض المعجمي حسب السياق التصنيفي المحدد.

Résumé

La classification automatique de textes constitue une tâche fondamentale en traitement du langage naturel, souvent confrontée au défi de l'ambiguïté lexicale des mots polysémiques. Cette recherche examine l'impact des méthodes de désambiguïsation lexicale (WSD) sur les performances de classification en évaluant cinq approches de WSD appliquées à quatre algorithmes de classification distincts. L'étude démontre qu'il n'existe pas de méthode de WSD universellement optimale, mais chaque algorithme de classification bénéfice d'une approche WSD spécifique. Les résultats expérimentaux montrent des variations significatives de performance selon les combinaisons testées, soulignant limportance de ladéquation entre la stratégie de désambiguïsation et lalgorithme dapprentissage. Cette recherche contribue à une meilleure compréhension de linteraction entre résolution dambiguïté sémantique et classification au tomatique, offrant des perspectives pour loptimisation des systèmes de traitement de textes. Cette étude suggère la nécessité dune approche adaptative dans le choix des méthodes de WSD selon le contexte classificatoire spécifique.