REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOHAMED BOUDIAF - M'SILA

FACULTE : Mathématique Et Informatique
DEPARTEMENT : Informatique

N° :……..

DOMAINE : Math **Et Informatique**
FILIERE : Informatique
OPTION : Système Information Et Génie Logicielles

Mémoire présenté pour l'obtention
Du diplôme de Master Académique

Par: BENSBAA Abdelaziz

Intitulé

Classification des textes arabes Basée sur une ontologie de domaine

Soutenu devant le jury composé de :

……..	Université de M'sila	Président
Dr.KADRI Said	Université de M'sila	Rapporteur
……	Université de M'sila	Examinateur

Année universitaire : 2017 /2018

REMERCIEMENTS

*Je tiens à remercier en premier lieu, **Dieu** qui ma a accordé la volonté et le courage pour la réalisation de ce projet.*

J'aimerais aussi d'adresser ma sincère gratitude à mon encadreur Dr.KADRI Said sans qui l'achèvement de ce travail n'aurait jamais été possible. Je le remercie pour n'avoir pas été avare de son temps précieux, pour sa disponibilité et sa générosité intellectuelle.

Mes remerciements s'étendent aux professeurs qui ont assuré notre formation dans le cadre de département d'informatique.

J'exprime ma plus profonde gratitude à ma famille; Mes parents, et à mes chers amis pour leur soutien et leur encouragement.

Je suis redevable à tous ceux qui, de près ou de loin, m'ont soutenue ou aidée dans mon travail.

Table des matières

LISTE DES FIGURES

Table des matières

Introduction Générale

INTRODUCTION

La recherche accorde ces dernières années, beaucoup d'importance au traitement des données textuelles. Ceci pour plusieurs raisons : un nombre croissant de collections mises en réseau et distribuées au plan international, le développement de l'infrastructure de communication et de l'Internet. Les traitements manuels de ces données s'avèrent très coûteux en temps et en personnel, ils sont peu flexibles et leur généralisation à d'autres domaines est presque impossible ; c'est pour cela que l'on cherche à mettre au point des méthodes automatiques.

Le domaine de la fouille de textes (text mining) s'est développé pour répondre à volonté à la gestion par contenu des sources volumineuses de textes. A l'heure actuelle, de nombreux logiciels de classification de textes sont disponibles, ils ont fait l'objet de publications et leurs champs d'application s'élargit de jour en jour. En général, ces systèmes sont basés sur des algorithmes d'apprentissage automatique (approche statistique, approche syntaxique et approche connexionniste).

Nous nous intéressons ici plus particulièrement à la construction automatique d'une ontologie comme une ressource sémantique à partir de textes arabes et aux algorithmes d'apprentissage et nous avons utilisé l'algorithme : les K plus proches voisins (Kppv). Pour pouvoir utiliser cet algorithme, il est nécessaire de transformer les données, initialement en format texte, en une représentation numérique. Nous avons choisi pour ce faire, la méthode de sélection des termes les plus pertinents. Une fois ce prétraitement terminé, nous pouvons effectuer la classification à l'aide de nos algorithmes.

Nous résumons la présentation de notre travail ainsi :

· **Le chapitre I** : est un survol sur le domaine de la classification des textes arabes et ses différents champs d'applications.

· **Le chapitre II** : présente brièvement les ontologies, ses concepts, ses domaines d'application et leur relation avec la classification de textes.

· **Le chapitre III** : Algorithme d'apprentissage et conception du système.

· **Le chapitre IV:** Implémentation et réalisation du système.

Chapitre1

Classification De Textes

CHAPITRE:1

CLASSIFICATION DE TEXTES : CONCEPTS, TECHNIQUES, ET OBJECTIFS

1 Introduction

L'objet d'étude de ce mémoire est la classification automatique de textes, c'est un problème qui intéresse les chercheurs depuis relativement longtemps. On retrouve des travaux portant sur ce sujet depuis au moins le début des années 1960.

La recherche dans ce domaine est toujours très pertinente, car les résultats obtenus aujourd'hui sont encore sujets à amélioration. Pour certaines tâches, les classificateurs automatiques performent presque aussi bien que les humains, mais pour d'autres, l'écart est encore grand. Au premier abord, l'essentiel du problème est facile à saisir. D'un côté, on est en présence d'une banque de documents textuels et de l'autre, d'un ensemble prédéfini de catégories. L'objectif est de rendre une application informatique capable de déterminer de façon autonome, dans quelle catégorie classer chacun des textes, à partir de leur contenu, tel qu'il est lustré à la **figure 1**.1 **[1].**

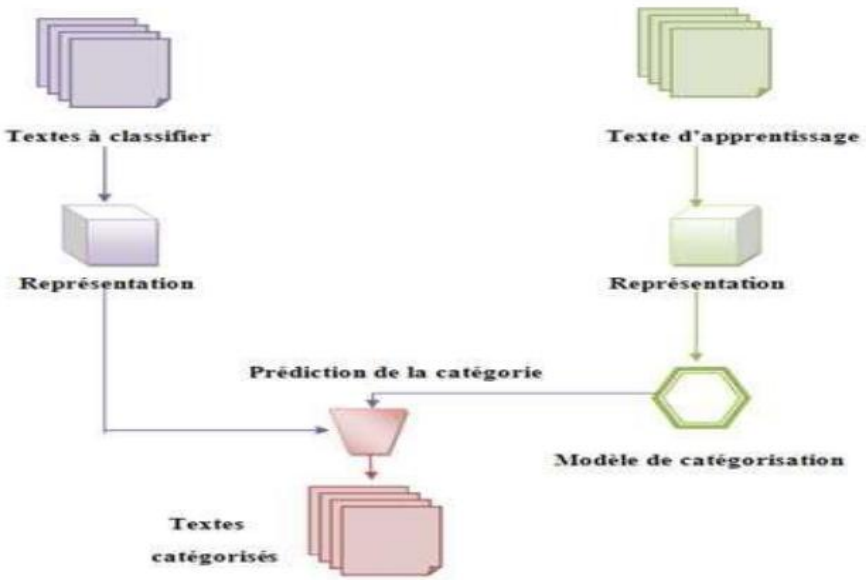

Figure 1.1: Processus de la catégorisation des textes.

La figure I.1 résume le processus de classification des textes qui comporte deux phases : l'apprentissage et le classement.

Malgré cette première définition très simple, la solution n'est pas immédiate et plusieurs facteurs sont à prendre en considération. Ce chapitre vise donc à définir plus en profondeur de quoi il en retourne. En premier lieu, il précise la tâche de catégorisation (classification), et le deuxième point présente le choix d'un mode de représentation adéquat des instances à traiter, en l'occurrence les textes. Il s'agit d'une étape incontournable en apprentissage automatique : on doit opter pour une façon uniforme et judicieuse d'abstraire les données avant de les soumettre à un algorithme. Comme l'apprentissage joue un rôle dans la catégorisation automatique de textes, le choix d'un mode de représentation y devient un enjeu. Par la suite, il sera question de la sélection d'attributs, presque toujours impliqués en catégorisation automatique de textes et on éliminera les attributs jugés inutiles à la classification. **[1].**

2 Définition de la classification de textes

Le but de la classification automatique de textes est d'apprendre à une machine à classer un texte dans le bon domaine en se basant sur son contenu. Habituellement, les catégories font référence aux sujets des textes, mais pour des applications particulières, elles peuvent prendre d'autres formes. En effet, on peut résoudre, par des techniques de catégorisation, des problèmes tels que le filtrage du courrier électronique pertinent ou indésirable (spams), ou encore la désambiguïsation de termes**.** Un autre aspect du problème qui varie selon les applications est la présence ou non d'une contrainte concernant le nombre de catégories assignables à un document donné. Il se peut qu'on désire qu'un même texte ne soit associé qu'à une seule catégorie ou bien on peut permettre que plusieurs catégories accueillent un même document. Aussi, une précision supplémentaire est à faire : dans le cadre de la catégorisation de textes, l'ensemble de catégories possibles est déterminé à l'avance. Il est à noter que le problème consiste à regrouper des documents selon leur similarité.

Dans une catégorisation de texte : la classification s'apparente au problème de l'extraction de la sémantique d'un texte, puisque l'appartenance d'un document à une catégorie est étroitement liée à la signification de ce texte. C'est en partie ce qui rend la tâche difficile puisque le traitement de la sémantique d'un document écrit en langage naturel n'est pas encore solutionné. Une observation mérite aussi d'être faite concernant le fait que la nature des textes

à traiter influence significativement la difficulté de la tâche de classification. Prenons l'exemple d'articles de journaux écrits généralement dans un style direct et contenant de l'information purement factuelle. Le vocabulaire utilisé s'avère précis et souvent relativement restreint pour faciliter la compréhension. A l'opposé, imaginons un corpus de textes d'un style plus littéraire, utilisant un vocabulaire très varié et imagé. On peut aisément prévoir que la classification automatique de ce dernier corpus présentera plus de difficultés que pour l'autre. Entre ces deux extrêmes, on peut aussi retrouver des textes scientifiques (où chaque catégorie aura potentiellement un vocabulaire caractéristique), des pages Web, du courrier électronique, etc. Chacun de ces types de textes possède des particularités qui rendent la tâche de classification plus ou moins ardue.

2.1 type de classification

2.1.1 Classification non-supervisée

L'apprentissage non-supervisé consiste à apprendre sans superviseur. Il s'agit d'extraire des classes ou groupes d'individus présentant des caractéristiques communes. La qualité d'une méthode de classification est mesurée par sa capacité à découvrir certains ou tous les motifs cachés

2.1.2 Classification supervisée

Il s'agit d'apprendre à classer un nouvel individu parmi un ensemble de classes prédéfinies. C.àd, on connaît les classes à priori.

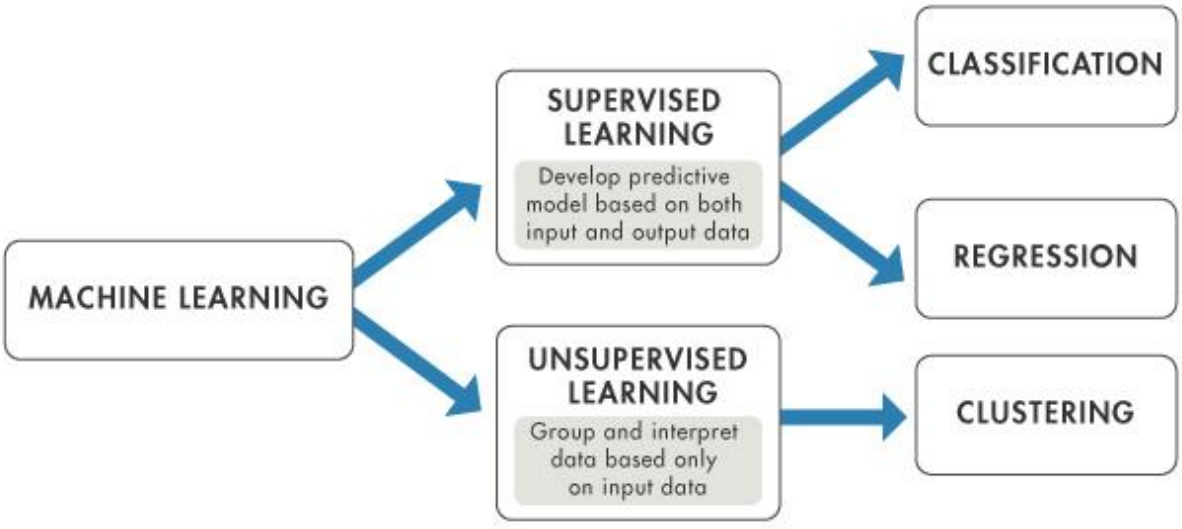

Figure 1.2: type de classification.

3 Comment classer un texte ?

Le processus de classification intègre la construction d'un modèle de prédiction qui, en entrée, reçoit un texte et, en sortie, lui associe une ou plusieurs domaines.

Pour identifier le domaine ou la classe à laquelle un texte est associé, un ensemble d'étapes est habituellement suivies. Ces étapes concernent principalement la manière dont un texte est représenté, le choix de l'algorithme d'apprentissage à utiliser et comment évaluer les résultats obtenus pour garantir une bonne généralisation du modèle appris.

Le processus de classification, intégrant la phase de classement de nouveaux textes, est résumé dans la **figure 1**.2 Il comporte deux phases que l'on peut distinguer comme suit :

3.1 L'apprentissage

Qui comprend plusieurs étapes et aboutit à un modèle de prédiction :

a) Nous disposons d'un ensemble de textes étiquetés (pour chaque texte nous connaissons sa catégorie/classe/groupe).

b) A partir de ce corpus, nous extrayons les **k** descripteurs (ou mots, ou termes) $(t_1;\ldots;t_k)$ les plus pertinents au sens du problème à résoudre.

c) Nous disposons alors d'un tableau « descripteurs × individus », et pour chaque texte nous connaissons la valeur de ses descripteurs et son étiquette.

d) Nous appliquons un algorithme d'apprentissage sur ce tableau afin d'obtenir un modèle de prédiction.

3.2 La classification proprement dite

La classification d'un nouveau texte d_x, comprend deux étapes :

a) Recherche puis pondération des occurrences $(t_1;\ldots;t_k)$. Des termes dans le texte d_x à classer.

b) Application du modèle sur ces occurrences afin de prédire l'étiquette de ce texte d_x..

Notons que les k descripteurs les plus pertinents $(t_1;\ldots;t_k)$ sont extraits lors de la première phase par analyse des textes du corpus d'apprentissage. Dans la seconde phase, celle du classement d'un nouveau texte, nous cherchons simplement la fréquence de ces k descripteurs $(t_1;\ldots;t_k)$ dans ce texte à classer.

Dans la suite nous présentons brièvement ces étapes qui seront développées dans le troisième chapitre.

3.3 Représentation et codage des textes

Un codage préalable du texte est nécessaire, comme pour l'image, le son, etc., car il n'existe pas actuellement de méthode d'apprentissage capable de traiter directement des données non structurées, ni dans la phase de construction du modèle, ni lors de son utilisation en classification. Pour la majorité des méthodes d'apprentissage, il faut transformer l'ensemble des textes en un tableau croisé « Individus-Variables » :

- ✓ L'individu est un texte (un document) d_j, étiqueté lors de la phase d'apprentissage, il est classé dans la phase de prédiction.
- ✓ Les variables sont les descripteurs (les termes) t_k qui sont extraits des données textuelles.

Le **contenu du tableau** (les éléments w_{kj}), au croisement du texte j et du terme k, représente le poids de ce terme k dans le document j.

Le principal enjeu de la catégorisation de texte, par rapport à un processus d'apprentissage classique, réside dans la recherche des descripteurs (ou termes) les plus pertinents pour le problème à traiter. Différentes méthodes sont proposées pour le choix des descripteurs et des poids associés à ces descripteurs. Certains chercheurs utilisent, à titre d'exemples, les mots comme descripteurs, tandis que d'autres préfèrent utiliser les lemmes (racines lexicales); ou encore des *stemmes* (la suppression d'affixes). **[2].**

3.4 Types de coding

3.4.1 Choix du terme

Dans la catégorisation de textes, comme dans la recherche documentaire, on transforme le document d_j en un vecteur $d_j = (w_{1j}, w_{2j}, ..., w_{|T|j})$, où T est l'ensemble de termes (descripteurs) qui apparaissent au moins une fois dans le corpus (la collection) d'apprentissage. Le poids w_{kj} correspond à la contribution des termes t_k à la sémantique du texte d_j.

3.4.2 Représentation en « sac de mots »

La représentation de textes la plus simple a été introduite dans le cadre du modèle vectoriel elle porte le nom de « sac de mots ». L'idée est de transformer les textes en vecteurs dont chaque composante représente un mot. Les mots ont l'avantage de posséder un sens explicite. Cependant, plusieurs problèmes se posent. Il faut tout d'abord définir ce qu'est « un mot » pour pouvoir le traiter automatiquement. On peut le considérer comme étant une suite de caractères

appartenant à un dictionnaire, ou bien, de façon plus pratique, comme étant une séquence de caractères non délimiteurs encadrés par des caractères délimiteurs. Les composantes du vecteur sont une fonction de l'occurrence des mots dans le texte. Cette représentation des textes exclut toute analyse grammaticale et toute notion de distance entre les mots .d'autres auteurs parlent d'« ensemble de mots » lorsque les poids associés sont binaires. **[2].**

3.4.4 Représentation des textes par des phrases

Malgré la simplicité de l'utilisation de mots comme unité de représentation, certains auteurs proposent plutôt d'utiliser les phrases comme unité. Les phrases sont plus informatives que les mots seuls, car les phrases ont l'avantage de conserver l'information relative à la position du mot dans la phrase. Logiquement, une telle représentation doit obtenir de meilleurs résultats que ceux obtenus via les mots. Mais les expériences présentées ne sont pas concluantes car, si les qualités sémantiques sont conservées, les qualités statistiques sont largement dégradées

3.4.5 Représentation des textes avec des racines lexicales et des lemmes

Dans le modèle précédent (représentation en «sac de mots»), chaque flexion d'un mot est considérée comme un descripteur différent et donc une dimension de plus; ainsi, les différentes formes d'un verbe constituent autant de mots. Par exemple: les mots «déménageur, déménageurs, déménagement, déménagements, déménager, déménage, déménagera, etc.» sont considérés comme des descripteurs différents alors qu'il s'agit de la même racine « déménage »; les techniques de désuffixation *(*ou *stemming*), qui consistent à rechercher les racines lexicales, et de « *lemmatisation »* cherchent à résoudre cette difficulté. Pour la recherche des racines lexicales, plusieurs algorithmes ont été proposés; l'un des plus connus pour la langue anglaise est l'algorithme de Porter [10].

La lemmatisation consiste à remplacer les verbes par leur forme infinitive, et les noms par leur forme au singulier. Un algorithme efficace, nommé TreeTagger[11], a été développé pour les langues anglaise, française, allemande et italienne.

L'extraction des *stemmes* repose sur des contraintes linguistiques bien moins fortes; elle se base sur la morphologie flexionnelle mais aussi dérivationnelle. De ce fait, les algorithmes sont beaucoup plus simplistes et mécaniques que ceux permettant l'extraction des lemmes, ils sont donc plus rapides, mais leur précision et leur qualité sont naturellement inférieures.

3.4.6 Codage Term frequency × inverse document frequency (TF × IDF)

Le codage TF × IDF (acronyme pour *«term frequency × inverse document frequency»*) a été introduit dans le cadre du modèle vectoriel, il donne beaucoup d'importance aux mots qui apparaissent souvent à l'intérieur d'un même texte, ce qui correspond bien à l'idée intuitive que ces mots sont plus représentatifs du document. Mais sa particularité est qu'elle donne également moins de poids aux mots qui appartiennent à plusieurs documents : Pour refléter le fait que ces mots ont un faible pouvoir de discrimination entre les classes. **[1].**

Le poids d'un terme t_k dans un document d_j est calculé ainsi :

$$\text{TF} \times \text{IDF}\ (t_k, d_j) = (t_k, d_j).\log\frac{|Tr|}{(tk)} \qquad (1)$$

Où

- (t_k, d_j) est le nombre d'occurrences de t_k dans dj
- $|T_r|$ est le nombre de documents d'apprentissage.
- (t_k) est le nombre de documents d'entraînements dans lesquels *tk* apparaît au moins un fois.

4.4.7 Codage TFC

Le codage TF × IDF ne corrige pas la longueur des documents. Pour ce faire, le codage TFC est similaire à celui de TF×IDF : mais, il corrige la longueur des textes par la normalisation en cosinus, afin de ne pas favoriser les documents les plus longs. **[2]**

$$\text{TFC}(t_k, d_j) = \frac{\text{TF} \times \text{IDF}\ (tk,dj)}{\sqrt{\sum_{s=1}^{|T|}(\text{TF} \times \text{IDF}\ (tk,dj))^2}} \qquad (2)$$

4 Applications de la classification de textes

La classification de textes est utilisée dans de nombreuses applications. Parmi ces domaines figurent : l'identification de la langue, la reconnaissance d'écrivains, la catégorisation de documents multimédia, et bien d'autres.

La classification de textes peut être un support pour différentes applications parmi lesquelles le filtrage, qui consiste à déterminer si un document est pertinent ou non (décision binaire), par exemple la détection de *spams* (les courriers indésirables) pour ensuite les supprimer, le routage diffusion sélective d'information. Lors de la réception d'un document l'outil choisit à quelles personnes le faire parvenir en fonction de leurs centres d'intérêt. Ces centres d'intérêt correspondent à des profils individuels. **[2].**

5 Lien avec la recherche documentaire

La classification de texte est proche de la recherche documentaire. En recherche documentaire, on doit retrouver les documents qui correspondent à une requête, ce qui revient à classer tout le corpus en deux classes : les textes correspondant à la requête d'une part, les autres d'autre part. En catégorisation, il s'agit d'attribuer les documents à un ou plusieurs groupes, en fonction des informations qu'ils contiennent.

6 Choix de classifieurs

La classification de textes comporte un choix de technique d'apprentissage (ou classifieur) disponibles. Parmi les méthodes d'apprentissage les plus souvent utilisées figurent l'analyse factorielle discriminante [3], la régression logistique [4], les réseaux de neurones [5], les plus proches voisins [27], les arbres de décision [6], les réseaux baye-siens [7],les machines à vecteurs supports [8].

Ces classifieurs se différencient selon leur mode de construction de classifieurs (les classifieurs sont-ils construits manuellement, ou bien automatiquement par induction à partir des données ?) et selon leurs caractéristiques, (le modèle appris est-il compréhensible, ou bien s'agit-il d'une fonction numérique calculée à partir de données servant d'exemples ?).

Généralement, le choix du classifieur est fonction de l'objectif final à atteindre. Si l'objectif final est, par exemple, de fournir une explication ou une justification qui sera ensuite présentée à un décideur ou un expert, alors on préférera les méthodes qui produisent des modèles compréhensibles tels que les arbres de décision ou les classifieurs à base de règles.

7 Difficultés particulières de la catégorisation de textes

L'utilisation des méthodes d'apprentissage automatique afin de traiter les données textuelles est plus difficile que le traitement de données numériques. Le langage naturel (par opposition aux langages informatiques) n'est pas univoque : « un langage univoque (ou plus précisément

bi-univoque) est un langage dans lequel chaque mot ou expression a un seul sens, une seule interprétation possible et il n'existe qu'une seule manière d'exprimer un concept donné » [9]. Le langage naturel est équivoque : il y a plusieurs façons d'exprimer la même idée (la redondance), ce qui est exprimé possède souvent plusieurs interprétations (l'ambiguïté) et tout n'est pas exprimé dans le discours (l'implicite). Ajoutant à ces particularités la grande dimensionnalité des descripteurs, et la subjectivité de la décision prise par les experts qui déterminent la catégorie dans laquelle classer un document.

8 Évaluation de la qualité des classifieurs

Afin de pouvoir comparer les résultats produits par différents modèles, et afin de s'assurer que le modèle est généralisable à d'autres textes, nous devons appliquer une méthode d'évaluation. La performance d'un classifieur dans la classification de textes est souvent mesurée via la précision [3] (traduction de *precision*) et le rappel [4] (traduction de *recall*).

Notons que les taux de succès et d'erreur sont rarement utilisés pour mesurer la performance d'un classifieur,

9 Conclusion

Ce chapitre introductif a fixé les objectifs de la classification et a introduit les notions nécessaires à la compréhension des chapitres suivants. Nous avons présenté la notion de classification de textes. Nous avons vu que le processus de classification comporte généralement trois étapes : la représentation, le choix de classifieur et la méthode d'évaluation du modèle.

La phase de représentation est importante et comporte deux choix qui affectent souvent les performances : le choix de termes (mot, lemme, stemm ou n-grammes) et le choix des poids associés à ces termes (absence/présence, nombre d'occurrences, fréquence, ... *etc.*). Le choix de la méthode d'apprentissage est également primordial ; de nombreuses méthodes sont proposées, chacune possédant des avantages, et des inconvénients. L'évaluation des modèles permet de s'assurer de bonnes performances en généralisation, c'est à dire sur d'autres données, qui n'ont pas été utilisées pour l'apprentissage. Dans ce chapitre, nous avons parlé des applications de la classification de textes. L'application des algorithmes d'apprentissage aux données textuelles introduit des difficultés supplémentaires.

Dans le chapitre suivant nous présenterons les ontologies comme une ressource sémantique dans le domaine de la classification de textes.

Chapitre 2
Les ontologies

CHAPITRE 2
LES ONTOLOGIES

1 Introduction

La masse de plus en plus croissante d'information dans tous les domaines a généré un besoin capital d'organisation et de structuration des contenus de documents, disponibles généralement sur le web. Les ontologies en sont un moyen prometteur et qui ne cesse de donner ses preuves. Leurs applications sont multiples : indexation, recherche d'informations, traduction automatique, e-Learning etc. Les principaux buts de la construction des ontologies sont la partageabilité, la portabilité, la réutilisabilité et la capitalisation de la connaissance et de l'expertise d'un domaine. Parce que l'information n'est pas statique, d'où elle se modifie, s'enrichisse, s'altère avec le temps et qu'elle vienne de différentes sources, nous avons besoin d'outils et de modèles qui permettent aux utilisateurs et aux experts du domaine de constituer, consulter et maintenir à jour leurs connaissances du domaine.

2 Définitions

Le mot *ontologie* qui vient du grec ***ontos*** =être et ***logos***= études, appartient à la philosophie ancienne grecque, Aristote le définit comme la science de l'Être en tant qu'être [14]. Il est difficile de définir ce qu'est une ontologie d'une façon définitive. Le mot est en effet employé dans des contextes très différents touchant à la philosophie, la linguistique ou l'intelligence artificielle.

Bien que des débats préexistent, nous parlons plus souvent d'ontologies (au pluriel) afin de refléter les multiples facettes que recouvre cette appellation [15]. Guarino (Guarino, 1996) et Dameron (Dameron, 2003), abordent les différentes définitions de la littérature afin d'examiner le type de représentation des connaissances dénoté par le terme ontologie. En 1993, Gruber propose une première définition « une ontologie est une spécification explicite d'une conceptualisation »[16]. L'expression *spécification explicite* signifie, que la conceptualisation est représentée dans un langage qu'il soit naturel (arabe, français..) ou formel (logique de description, graphes conceptuels..).

Une autre définition, peut-être plus rigoureuse : « Une ontologie implique une certaine vue du monde par rapport à un domaine donné. Cette vue est souvent conçue comme un ensemble de

concepts (entités, attributs, processus, leurs définitions et leurs interrelations). On appelle cela une conceptualisation » [17]. Une ontologie peut prendre différentes formes mais elle inclura nécessairement un vocabulaire de termes et une spécification de leur signification [15]. En résumé, nous pouvons définir une ontologie comme l'ensemble représentatif et exhaustif des termes d'un domaine donné avec toutes les relations qui les relient.

3 Constituants d'une ontologie

Une ontologie en tant qu'artefact informatique, comme nous l'avons défini dans la section2, est donc constituée de termes, leur définition, leurs propriétés, des contraintes sur les propriétés, des individus, et des relations. Dans ce qui suit, nous allons aborder ces concepts plus en détails.

3.1 Les concepts

Un concept, également appelé *classe* dans certains travaux ou outils, représente l'idée que l'on se fait d'un terme : le contenu. Il est porteur d'une connaissance. Il peut désigner un objet concret comme :(**كتاب** =*livre)* ou abstrait comme *:(* إحساس = Sensation)

3.2 Les propriétés

La propriété est une caractéristique qui qualifie un concept et qui peut généralement être dotée d'une valeur. Si nous prenons l'exemple précédent (**رياضة**=sport) nous pouvons designer quelques propriétés comme : (**اسم_رياضة** = nom_du_sport), (**نوع_الرياضة** = type_de_sport) (**عدد_اللاعبين**= nombre_de_joueurs), ceci bien sûr, dans un certain contexte.

3.3 Les facettes

Les facettes sont des restrictions sur les valeurs des propriétés, si nous prenons l'exemple cité dans la section 3.2, La facette de (**رياضة** = sport) sera une liste de tous les sports, en l'occurrence

(السباحة = Natation ; كرة_القدم= football ; ….) ,(nombre_de_joueurs), ce sera tout simplement un entier.

3.4 Les instances

Les instances d'un concept concret sont des éléments singuliers de ce concept, aussi appelées individus dans certains travaux. (السباحة = Natation ; كرة_القدم= football ; التنس=tennis) sont des

instances de (**رياصة** =sport). Les instances ne sont nécessaires que lorsque l'objectif de l'ontologie est de servir à la construction d'une base de connaissances.

3.5 Les relations

Les relations sont un type d'interaction entre deux concepts. La relation la plus utilisée est sans doute celle qui établit la hiérarchie de la structure ontologique, c'est la relation de subsomption

(**عبارة_عن**= *est_un).* B *est_un* A, exprime le fait que le concept B est un sous concept du concept A, dans le sens où B hérite de toutes les propriétés de A et a forcément des propriétés spécifiques.

Exemple : (إنسان عبارة_عن **رجل) et (إنسان** عبارة_عن **إمرأة)**

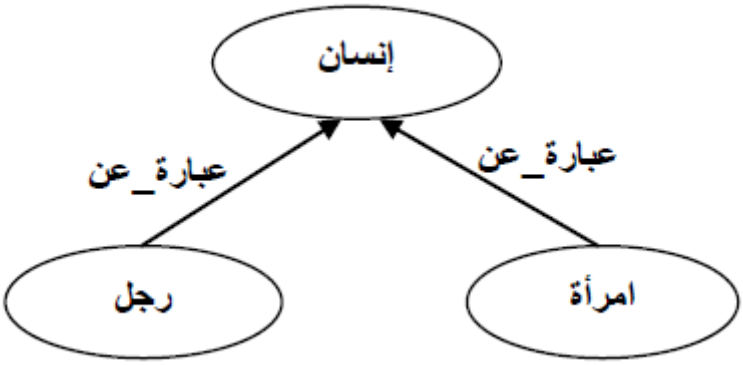

Figure 2.1: La relation de subsomption

Il existe deux types de relations : celles qui sont indépendantes du domaine et celles qui sont étroitement liées au domaine choisi. Les relations indépendantes du domaine sont générales et peuvent être utilisées dans n'importe quel champ de spécialité, les plus connues sont (**عبارة_عن** = *est_un*), (**نوع_من** = *sorte_de*), (**جزء_من**= *partie_de*) etc. Les relations dépendantes du corpus, spécifiques à un domaine, ont un sens précis dans le domaine utilisé.

Exemple : (يلعب_كرة =jouer_la_bale) dans (ليلعب_كرة_القدم)**.**

Cette relation ne peut être utilisée que dans certains domaines et elle a un sens bien précis par rapport à ce domaine.

4 Typologies des ontologies

Il existe plusieurs typologies des ontologies, proposées par des groupes de recherche selon l'objectif principal pour lequel l'ontologie a été conçue. Nous présentons dans ce qui suit les plus importantes.

4.1 Typologie de Uschold et Grüninger

Uschold et Grüninger [**22**] ont classé les ontologies selon leur degré de formalisme.

4.1.1 Les ontologies hautement informelles

Ce sont des ontologies écrites en langage naturel, elles ne sont pas traitées par une machine.

4.1.2 Les ontologies semi-informelles

Elles utilisent un langage naturel structuré et limité, elles se situent entre les ontologies formelles et informelles.

4.1.3 Les ontologies rigoureusement formelles

Elles sont définies dans un langage contenant une sémantique formelle, elles sont facilement traitées par une machine peuvent être intégrées dans des applications.

4.2 Typologie de Gómez-Pérez

Cette typologie a été proposée par Gomez-Pérez, à l'université de Madrid [**26**]. Elle s'intéresse aux objets que modélisent les ontologies, elle les classe ainsi :

4.2.1 Ontologies pour la représentation des connaissances

Elles sont utilisées généralement lors de la construction d'un système à base de connaissance.

4.2.2 Ontologies de domaine

Elles servent à fournir une modélisation d'un domaine de connaissance donné comme la médecine ou une spécialité en médecine comme l'ophtalmologie.

4.2.3 Ontologies de haut niveau

Elles sont généralement exprimées en termes de scenarios, événements, temps et objets. On peut y greffer n'importe quelle autre ontologie du domaine.

4.2.4 Ontologies génériques

Elles ne sont pas spécifiques à un domaine précis, Aussi appelée méta-ontologie, véhicule des connaissances génériques qui, bien que moins abstraites que celles modélisées dans l'ontologie de haut niveau, doivent être assez générales pour être réutilisées dans différents domaines. Elles organisent des connaissances factuelles ou des connaissances visant à résoudre des problèmes génériques d'un ou de plusieurs domaines**.**

5 Formalismes de représentation des ontologies

La formalisation en elle-même n'est pas une étape incontournable de la méthodologie de construction. Une utilisation simple d'une ontologie en recherche d'information par exemple ne nécessite pas forcément une formalisation. Cependant certaines tâches complexes ont besoin de cette étape, qui assure entre autre, l'intégrité, la consistance et la complétude du contenu ontologique surtout si celui-ci se trouve volumineux, elle permet par la suite l'opérationnalisation et l'intégration de l'ontologie dans les différentes applications visées. Pour pouvoir raisonner sur une ontologie, il faut formaliser la connaissance. Formaliser les connaissances d'un domaine, consiste à les coder en un langage formel de description de connaissances, pour qu'elles puissent être traitées par une machine. Formaliser une ontologie c'est permettre de vérifier sa hiérarchie, sa consistance et pouvoir l'intégrer facilement dans d'autres systèmes et raisonner dessus. Il existe plusieurs formalismes de représentations des connaissances, mais les plus utilisés dans les ontologies sont principalement :

- Les graphes conceptuels
- Les logiques de description

5.1 Graphes conceptuels

Les graphes conceptuels sont introduits par [23] et appartiennent à la famille des réseaux sémantiques. Le modèle des graphes conceptuels comprend deux parties :

a) Une partie terminologique pour les concepts, les relations et les instances. Le niveau terminologique comprend donc, trois ensembles disjoints : L'ensemble des types de concepts (T_c), l'ensemble des types de relations (T_r) et l'ensemble des marqueurs individuels (M)

b) Une partie assertionnelle pour la représentation des assertions du domaine de connaissances.

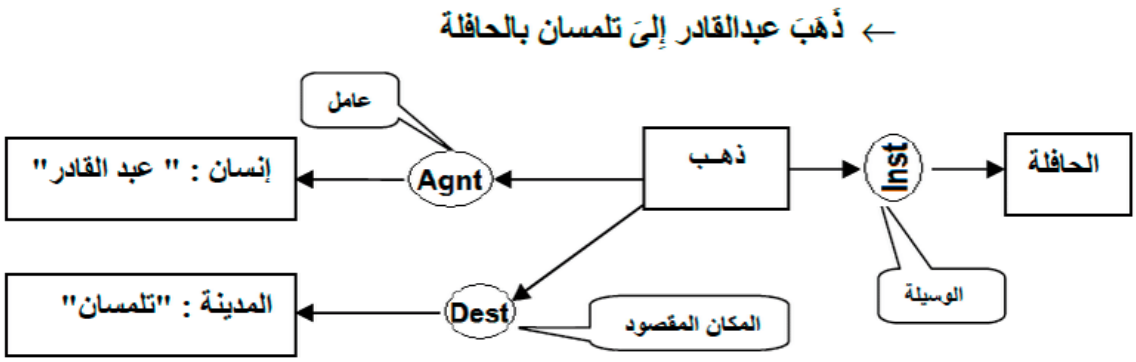

Figure 2.2: Un exemple de graphe conceptuel.

5.2 Les logiques de description

Les logiques de description sont des langages formels permettant de représenter des propriétés pour des ensembles d'objets. Elles se basent sur la logique du premier ordre. Elles ont pour vocation d'après leurs auteurs [24], d'enrichir les représentations sous forme de réseaux sémantiques ou frames, notamment au niveau des relations. Les logiques de descriptions étant le formalisme que nous avons choisi pour représenter la connaissance dans ce travail, nous y reviendrons dans la section dédiée à la formalisation de l'ontologie.

6 Langages de représentation des ontologies

L'un des principaux avantages d'une ontologie est la portabilité. Pour pouvoir exploiter une ontologie et la partager par un grand nombre d'utilisateur, il faut l'exprimer dans un langage permettant son utilisation sur différentes applications et plateformes. Ce langage doit répondre aux exigences des utilisateurs potentiels de cette ontologie. Il existe un grand nombre de langages développés à cet effet.

Tous sont basés sur la syntaxe XML, bien que XML lui-même ne soit pas un langage pour représenter les connaissances ontologiques. Nous allons citer dans cette section les plus utilisés et donc les plus enclins à respecter une certaine standardisation.

6.1 RDF & RDFs

Les initiales RDF correspondent à « Resource Description Framework », ou cadre de description de ressources en français, le « s » de schémas est une extension de RDF. Une ressource est simplement une *chose* : Une personne, un livre, un clavier, un article de publication, un bureau, une idée, toute *chose* qui peut être décrite. RDF est un cadre d'applications utilisant l'architecture du Web pour décrire une ressource. Tel HTML qui permet de relier des documents à d'autres documents sur le Web, RDF permet de relier une ressource à d'autres ressources sur le Web. Comme tous ses prédécesseurs, ce langage se base sur la syntaxe d'XML. Doté d'un schéma de représentation riche, incluant des classes, sous-classes, propriétés, sous-propriétés et des règles d'héritage de propriétés.

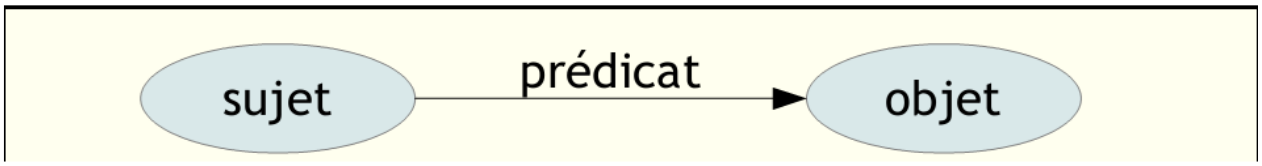

Figure 2.3: Le triplet RDF.

Dans un graphe, chaque triplet représente l'existence d'une relation entre les choses symbolisées par les nœuds qui sont joints. La structure objet-classe des RDFS permet de définir des objets du domaine et leurs relations pour rendre compte d'une ontologie [15]

6.2 OIL

OIL[13] (*Ontology Inference Layer*) est un langage de représentation d'ontologie qui dérive de RDF. Les principaux fondements du langage OIL sont les langages de frame (tels que OKBC, XOL ou RDF) et les logiques de descriptions. OIL a été défini dans l'objectif de permettre la spécification et l'échange d'ontologies.

6.3 DAML et DAML+OIL

DAML+OIL[14] (DARPA[15] *Agent Markup Language*) est un langage permettant la représentation d'ontologies. DAML est une combinaison de XML et de RDF permettant de spécifier des objets mais également les relations entre ces objets. La dernière version de DAML se combine avec OIL (DAML+OIL). Ce nouveau langage supporte désormais les types de données primitifs (tels qu'on les trouve dans la norme XML Schéma) et la définition d'un certain nombre d'axiomes comme l'équivalence de classes ou de propriétés [15].

6.4 OWL

Nous avons vu que RDF et RDFS permettent de définir, sous forme de graphes de triplets, des données ou des métadonnées. Cependant, de nombreuses limitations bornent la capacité d'expression des connaissances établies à l'aide de RDF/RDFS. On peut citer, par exemple, l'impossibilité de raisonner et de mener des raisonnements automatisés sur les modèles de connaissances établis à l'aide de RDF/RDFS. C'est ce manque que se propose de combler OWL.OWL [18] **(***Ontology Web Language*) a été créé en 2001 par le W3C, hérite du langage DAML+OIL et doit permettre de représenter des ontologies sur le Web. OWL fournit en fait trois sous-langages, d'expressivité croissante, nommés OWL Lite, OWL DL et OWL Full.

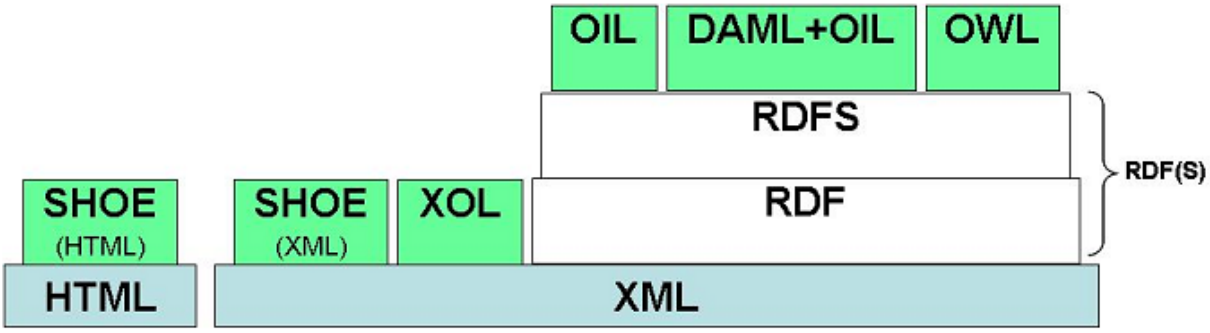

Figure 2.4: Les langages d'exploitation des ontologies [26]

OWL est devenu un standard du Consortium W3C qui a publié en 2004 une recommandation définissant le langage OWL fondé sur le standard RDF et en spécifiant une syntaxe XML. Plus expressif que RDFS, il tend à détrôner les autres langages et à s'imposer de plus en plus en maitre absolu.

7 Editeurs d'ontologies

Éditer une ontologie avec un outil adéquat permet son affichage sous forme arborescente, en outre l'intégration de plugins appropriés, permet la visualisation des différents concepts avec toutes les relations qui les relient, cela donne une vue plus globale de la disposition des concepts les uns par rapport aux autres. Certains éditeurs vont plus loin en permettant d'importer ou d'exporter une ontologie d'un format vers un autre, ce qui facilite largement sa portabilité et la génération automatique de fichiers OWL/XML ou RDF **[25].** Nous donnons dans ce qui suit un survol sur les éditeurs les plus importants, certains d'entre eux représentent de véritables plateformes avec de multiples plugins permettant de soumettre des requêtes de vérifier la consistance et de fusionner des ontologies existantes dans différents formats.

7.1 PROTEGE

PROTEGE est un éditeur d'ontologies, distribué en open source par l'institut d'informatique médicale de Stanford. C'est un éditeur hautement extensible, capable de manipuler des formats très divers. Il existe deux moyens pour modéliser une ontologie avec PROTEGE, PROTEGE-Frame et PROTEGE-OWL. Une ontologie en PROTEGE peut être exportée dans différent formats incluant RDF(s), OWL, XML schémas. PROTEGE est une plateforme Java, il est flexible et supporte plusieurs langues dont l'Anglais, le Français, l'Arabe, le Chinois le Russe etc. Une large communauté de développeurs académiques, de gouvernements et d'entreprises

utilise PROTEGE dans divers domaines. L'interface permet de créer, supprimer, modifier et mettre à jour les concepts, les propriétés, les instances et les relations.

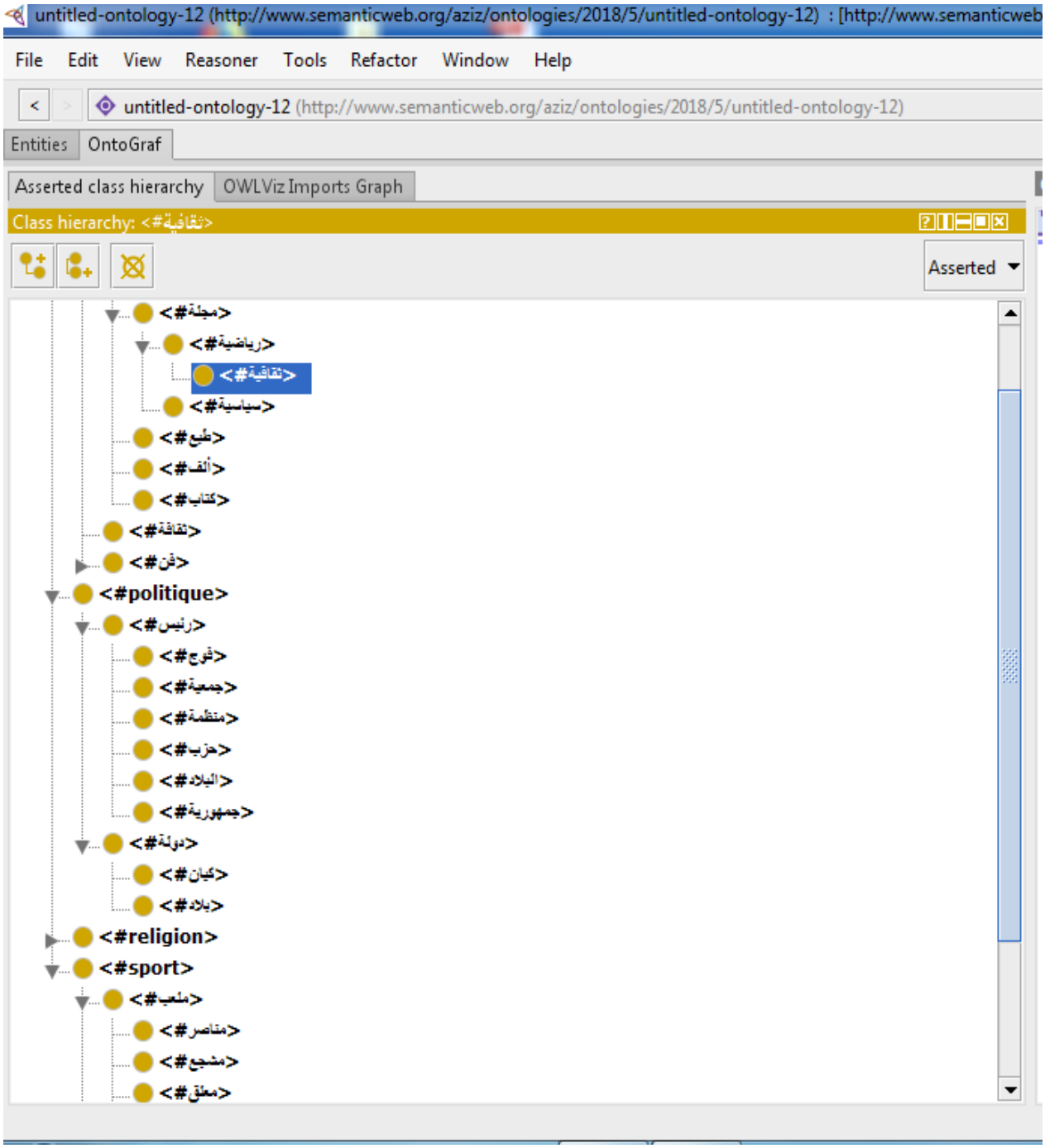

Figure 2.5: La hiérarchie Ontologique sous PROTEGE.

En plus de la visualisation de la hiérarchie ontologique, PROTEGE permet une visualisation graphique à l'aide de plugins comme OntoGraph ou OWL-Viz, il dispose de raisonneurs

comme Racer, Fact++, Hermitt, Pellet. OntoGraph, Fact++, Hermitt, Pellet sont fournis avec PROTEGE.

7.2 JENA

JENA est un environnement de travail open source en Java, pour la construction d'application web sémantique. JENA permet de manipuler des documents RDF, RDFS, OWL et SPARQL. Il fournit un moteur d'inférences permettant des raisonnements sur les ontologies. JENA est maintenant sous Apache Software Licence.

7.3 OilEd

OilEd est un éditeur qui a été développé par l'université de Manchester pour éditer des ontologies dans les langages de représentation OIL. Les versions disponibles de OilEd ne constituent pas un environnement complet pour le développement d'ontologies d'envergure.

7.4 ONTOLINGA

ONTOLINGA fournit un environnement collaboratif distribué pour chercher, créer, éditer, modifier, et utiliser des ontologies en ligne. Le serveur peut supporter jusqu'à 150 utilisateurs actifs. Certains fournissent une description de leurs projets. L'environnement assiste l'utilisateur dans les taches de développement et le maintien et leur permet de partager leur ontologie avec d'autres utilisateurs.

8 Conclusion

Nous avons abordé dans ce chapitre un tour d'horizon sur les différentes technologies utilisées dans le développement d'ontologies. Nous n'avons présenté qu'une liste succincte mais nous avons tenu à ce qu'elle soit la plus représentative possible des outils existant pour chaque phase de la création d'ontologie. Nous avons mis l'accent autant que cela était possible sur les outils libres et open source, qui peuvent servir beaucoup plus, dans le domaine de la recherche. Nous n'avons pas développé les sections concernant les outils ou techniques que nous avons adopté dans ce travail et que nous allons abordé dans les prochains chapitres puisqu'elles y seront exposés avec plus de détail comme la logique de description par exemple.

CHAPITRE3

Apprentissage Et Conception

CHAPITRE3
ALGORITHME D'APPRENTISSAGE, CONCEPTION

1 Introduction

En apprentissage automatique, différents types de classificateurs ont été mis au point, et cela dont le but d'atteindre un degré maximal de précision et d'efficacité, chacun ayant ses avantages et ses inconvénients. Mais, ils partagent toutefois des caractéristiques communes. Parmi la panoplie de classificateurs existants, on peut faire des regroupements et distinguer des grandes familles.

Dans les pages qui suivent, nous allons exposer quelques algorithmes en détail, le classificateur bayésien naïf algorithme surpassé par d'autres mais il est souvent utilisé comme point de référence à cause de sa simplicité, l'algorithme de Rocchio. Puis, les machines à support vectoriel et l'algorithme des k plus proches voisins les, qui représentent vraisemblablement à l'heure actuelle les deux meilleurs choix en catégorisation de textes. Dans ce chapitre nous décrivons aussi la conception de l'application, les outils utilisés et les différentes étapes de la réalisation.

2 Naïve bayes

L'algorithme Naïve Bayes (NB), est une autre méthode bien connue en apprentissage, elle est également utilisée dans la catégorisation de documents. Elle se base sur un modèle probabiliste, qui vise à estimer la probabilité conditionnelle d'une catégorie sachant un document et affecte au document la (ou les) catégorie(s) la (les) plus probable(s). La partie naïve de ce modèle est l'hypothèse d'indépendance des mots, c'est-à-dire que la probabilité conditionnelle d'un mot sachant une catégorie est supposée indépendante de cette probabilité pour les autres mots. Cette hypothèse fait que la catégorisation par NB est plus efficace que la complexité exponentielle des approches bayésiennes non naïves qui utilisent des combinaisons de mots comme prédicateurs [28].

2 Méthode de Rocchio

Le classificateur de Rocchio est l'un des plus simples et plus ancien algorithmes de classification issu du modèle vectoriel. Ce classificateur a été largement utilisé dans la

communauté de la catégorisation textuelle. Il se distingue par sa rapidité d'apprentissage et de classification grâce au fait qu'il nécessite très peu d'espace mémoire, contrairement à d'autres algorithmes [25].Un profil prototypique [c] est calculé pour chaque classe selon :

$$C_w = \frac{t}{Nc} \sum_{d \epsilon C} dw \; - \; \frac{1-t}{N_c} \sum_{d \notin c} dw \qquad (3)$$

3 Machine à vecteur support (SVM) :

Ces machines, proposées par *Vapnik* 1995 ont été utilisées avec succès dans plusieurs tâches d'apprentissage, elles sont actuellement en plein essor. Elles offrent en particulier une bonne approximation du principe de minimisation du risque structurel. Ce sont des méthodes de classification binaire par apprentissage supervisé. Ces méthodes sont une alternative récente pour la classification. Elles reposent sur l'existence d'un classificateur linéaire dans un espace approprié. Il s'agit d'un problème de classification à deux classes. Cette méthode fait appel à un jeu de données d'apprentissage pour apprendre les paramètres du modèle. Elle est basée sur l'utilisation de fonction dite noyau (*kernel*) qui permet une séparation optimale des données [29].

Dans le cas de la classification de textes, les entrées sont des documents et les sorties sont des catégories. En considérant un classificateur binaire, on voudra lui faire apprendre l'hyperplan qui sépare les documents appartenant à la catégorie et ceux qui n'en font pas partie. Les SVM conviennent bien pour la classification de textes [29]**.** Une dimension élevée ne les affecte pas puisqu'ils se protègent contre le sur-apprentissage. Dans le même sens, il affirme que peu d'attributs sont totalement inutiles à la tâche de classification et que les SVM permettent d'éviter une sélection agressive qui aurait comme résultat une perte d'information. On peut se permettre de conserver plus d'attributs. Egalement, une caractéristique des documents textuels est que lorsqu'ils sont représentés par des vecteurs, une majorité des entrées sont nulles. Or, les SVM conviennent bien à des vecteurs dits clairsemés. [1].

4 K plus proches voisins

k-PPV (K Plus Proches Voisins, ou KNN pour *K Nearest Neighbours*) a prouvé son efficacité face au traitement de données textuelles. La phase d'apprentissage consiste à stocker les exemples étiquetés. Le classement de nouveaux textes s'opère en calculant la distance entre la représentation vectorielle du document et celle de chaque exemple du

corpus. Les K éléments les plus proches sont sélectionnés et le document est assigné à la classe majoritaire (le poids de chaque exemple dans le vote étant éventuellement pondéré par sa distance).

L'algorithme ci-dessous montre comment classer un nouvel exemple par la méthode KPPV :

Algorithme : algorithme de classification par K-PPV

paramètre : le nombre K de voisin

contexte : un échantillon de l textes classés en $C=c_1, c_2,\ldots c_n$ classes

1 : **pour** chaque texte *t* **faire**

2 : transformer le texte t en vecteur $t=(x_1, x_2,\ldots, x_m)$.

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.

Le choix de la distance et du paramètre k est primordial pour le bon fonctionnement de la méthode. Nous présentons maintenant les différents choix possibles pour la distance et pour le mode de sélection de la classe du cas présenté. [2].

4.1 Définition de la distance

Soit E : un ensemble de textes.

Soient a, b, c trois points de l'espace E (Les points sont des textes).

Une distance est une application de $E \times E$ dans R^+ qui vérifie les propriétés suivantes :

$d (a , a) = 0$;

$d (a , b) = 0 \quad ; \ a=b$

$d (a , b) = d (b , a)$;

d (a , b) = d (a , c) + d (c , b)

Dans notre cas en utilise cet algorithme pour la classification des textes selon les corpus déjà annoté manuellement, on retiendra la classe la plus représentée parmi les *k* textes les plus proches.

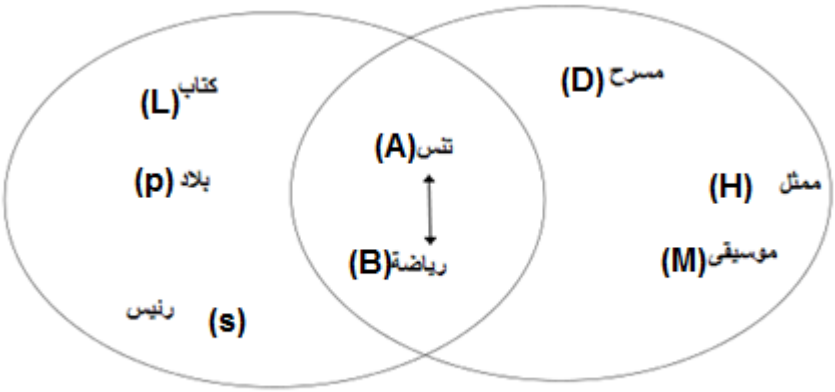

Figure 3.1 distance entre document.

On peut noter qu'un point **A** peut avoir un plus proche voisin **B** qui, lui-même, possède de nombreux voisins plus proches que **A** (**Figure 3.1**).

La distance entre un texte et ses voisins se fait via une métrique de distance. Cette métrique peut être la métrique de Minkowski **[2].**

$$d_p(a,b) = |a_i - b_i|^{p\,\frac{1}{p}} \qquad (4)$$

Selon la valeur de p, on retrouve plusieurs distances connues :

1. Si p = 1 cette distance est la distance de Manhattan définie par

$d_m(a,b) = \{|a_1 - b_1| + |a_2 - b_2| + \ldots + |a_n - b_n|\}$

2. Si p = 2 c'est la distance euclidienne définie par :

$$d_e(a,b) = \sqrt{a - b} \qquad (5)$$

4.2 Choix de k

La valeur de *k* est un des paramètres à déterminer lors de l'utilisation de ce type de classificateur. La valeur que l'on choisit pour *k* va être plus critique, plus déterminante en

rapport avec la performance du classificateur. On peut se permettre de considérer un plus grand nombre de voisins, sachant que plus ils diffèrent du document à classer, moins ils ont d'impact sur la prise de décision. Cependant, il demeure nécessaire de limiter le nombre de voisins pour s'en tenir à un temps de calcul raisonnable. [1].

4.3 Mesure de similarité

Une des caractéristiques fondamentales de ce type de classificateur est l'utilisation d'une mesure de similarité entre les documents. Les textes étant représentés sous forme vectorielle, donc comme des points dans un espace à *n* dimensions.

1) On peut au premier abord penser à déterminer les voisins les plus proches en calculant la distance euclidienne entre ces points.

Distance euclidienne entre deux documents ***a et b*** = $\sqrt{\boldsymbol{t}\quad(\boldsymbol{p_t(a)-p_t(b)})^2}$ **(6)**

Où :

· *T* est l'ensemble des attributs

· $p_t(a)$ est le poids du terme *t* dans le document *a*

· $p_t(b)$ est le poids du terme *t* dans le document *b*

2) Une autre façon de calculer la similarité des documents :

Similarité cosinusoïdale entre deux documents *a et b :* $\frac{p_t(a).p_t(b)}{\sqrt{T^{p}{}_{t(a)^2}.pt^{(b)^2}}}$ **(7)**

Où :

· *T* est l'ensemble des attributs.

· $p_t(a)$ est le poids du terme *t* dans le document *a*

· $p_t(b)$ est le poids du terme *t* dans le document *b*

La **similarité cosinusoïdale** est préférable en classification de textes pour plusieurs raisons :
- Elle permet de comparer des textes de longueurs différentes en normalisant leur vecteur.
- Elle met l'accent plutôt sur la présence de mots que sur l'absence de mots. (La présence de mots est probablement plus représentative de la catégorie du texte que l'absence de mots).

Voir l'exemple suivant : Pourquoi la similarité cosinusoïdale est préférable à la distance euclidienne. Nous allons considérer le vocabulaire et les trois documents suivants :

· Vocabulaire : < فريق, كتاب ,ملف , عبادة>

· Document 1 : <1, 1, 1, 0>

· Document 2 : <0, 0, 1, 0>

· Document 3 : <0, 0, 0, 0>

1 : signifie que le mot et présent dans le document.

0 : signifie l'absence de mot.

Par exemple le mot « فريق » est présent dans le document 1 **(**indiqué par 1**)** et absent dans les documents 2 et 3 **(**indiqué par 0**)**

$$d_c(doc2, doc1) = \sqrt{(1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2} = \mathbf{1.41}$$

$$d_c(doc3,doc2) = \sqrt{(0-0)^2 + (0-0)^2 + (1-0)^2 + (0-0)^2} = \mathbf{1}$$

La mesure de distance euclidienne indique que le document 3 est moins distant (plus similaire) du document 2 (distance de 1) que le document 2 ne l'est du document 1 (distance de 1.41).

Pourtant, les documents 2 et 3 n'ont aucun mot en commun, alors que les documents 1 et 2 partagent le mot «ملف».

$$\text{Sim (doc 2 , doc3)} = \frac{0.0+0.0+1.0+0.0}{\sqrt{(0^2+0^2+1^2+0^2)(0^2+0^2+0^2+0^2)}} = 0$$

$$\text{Sim (doc 1 , doc2)} = \frac{0.0+0.0+1.1+0.0}{\sqrt{(1^2+1^2+1^2+0^2)(0^2+0^2+1^2+0^2)}} = 0.57$$

La mesure de similarité cosinusoïdale, pour sa part, retourne une similarité nulle (0) entre les documents 2 et 3, mais positive (0.57) entre les documents 1 et 2.

Ces derniers résultats sont plus représentatifs de la réalité, puisque les documents 1 et 2 partagent au moins un mot, donc présentent une certaine similarité et sont plus susceptibles d'être classés dans la même catégorie que les documents 2 et 3 qui ne partagent aucun mot.

4.4 Mise en place de la méthode

La méthode ne nécessite pas de phase d'apprentissage. Le modèle est constitué de trois éléments :

1) l'échantillon d'apprentissage.

2) la distance ou Similarité.

3) la méthode de combinaison des voisins.

L'efficacité de la méthode dépend de ces trois éléments.

- Il faut choisir l'échantillon, c'est-à-dire les attributs pertinents pour la tâche de classification considérée et l'ensemble des enregistrements. Il faut veiller à disposer d'un nombre assez grand d'enregistrements par rapport au nombre d'attributs et à ce que chacune des classes soit bien représentée dans l'échantillon choisi.
- Une heuristique fréquemment utilisée pour le choix du nombre k de voisins est de prendre k égal au nombre d'attributs.
- L'emploi de *k* voisins, au lieu d'un seul, assure une plus grande robustesse à la prédiction. Classiquement, dans le cas où la variable à prédire comporte deux étiquettes, ce paramètre k est une valeur impaire afin d'avoir une majorité plus facilement décidable.

5 Conception de l'application

5.1 Construction d'une ontologie à partir d'un thesaurus

Un thesaurus fournit une cartographie d'un champ de connaissance en indiquant comment les concepts ou les idées associées sont reliés. Un thesaurus aide un indexeur ou une personne qui recherche de l'information à comprendre la structure d'un domaine. Plus précisément, un thésaurus est un ensemble de termes normalisés et contrôlés, descripteurs et non-descripteurs obéissant à des règles terminologiques propres. Les termes d'un thésaurus sont reliés entre eux par des relations sémantiques (hiérarchiques, associatives, ou d'équivalence).

5.1.1 Thésaurus vs Ontologie

Thésaurise contient des ressources lexicaux, collection de termes organisée hiérarchiquement et des relations entre les termes.

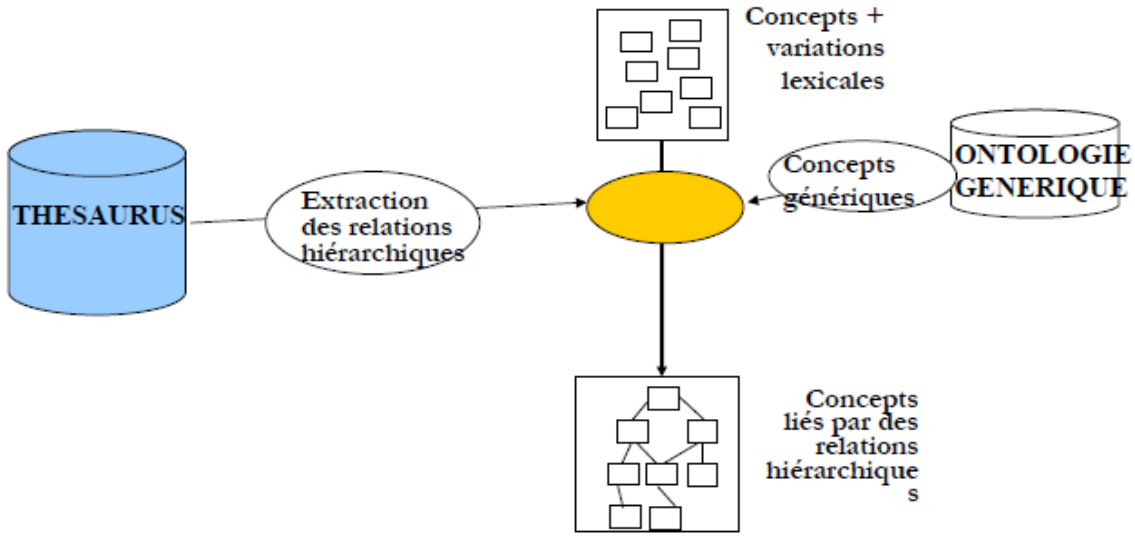

Figure 3.2 Thesaurus vers les ontologies.

5.1.2 Enrichir l'ontologie à partir du thesaurus

L'avantage principale de thésaurise c'est en cas de capture de nouveaux relations entre concepts non représentes dans l'ontologie on peut l'ajouter simplement.

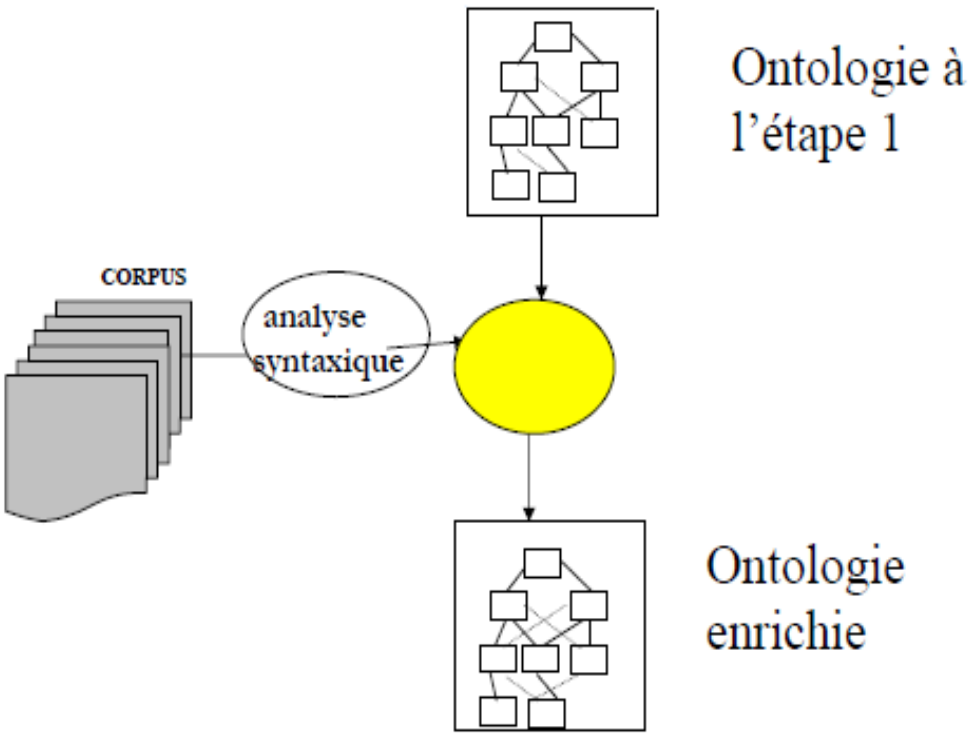

Figure 3.3 Mise à jour L'ontologie à partir de thésaurus.

5.2 Classification d'un nouveau texte

La tâche de classification d'un nouveau texte faite par quelque étapes son :

5.2.1 Prétraitement

Le prétraitement de texte est l'étape préliminaire pour identifier les données lexicales à partir de ce texte. (Les prétraitements des données textuelles consistent à normaliser les diverses manières d'écrire un même mot, à corriger les fautes d'orthographe évidentes ou les incohérences typographiques et à expliciter certaines informations lexicales exprimées implicitement dans les textes. Les traitements consistent, par exemple, à extraire la structure superficielle des textes à partir d'indices comme une ligne vide pour délimiter les paragraphes.

5.2.2 La segmentation

La segmentation est une étape quasiment obligatoire avant l'extraction d'information. Elle permet de découper le texte en unités linguistiques suffisamment élémentaires pour qu'elles soient traitées. C'est une étape qui permet de découper un texte d'abord en section puis en phrase et enfin en mot.

Exemple **(الصحة /تاج / رؤوس /الاصحاء)**

5.2.3 Lemmatiseur

C'est un programme de traitement automatique du langage qui permet de passer d'un mot portant des marques de flexion (pluriel, forme conjuguée d'un verbe...) à sa forme de référence (lemme ou forme canonique).

exemple : يستعبدونه ⟶ عبادة

5.2.4 Extracteur de relations sémantiques

Il s'agit de repérer des relations entre des termes préalablement extraits, telle la synonymie, l'hyponymie ou encore la métonymie.

Exemple : Une phrase telle que: العلم نور و الجهل ظلام.

Génère une relation de similitude entre les *العلم* et النور.

5.2.5 Stemming

Le Stemming est une technique qui a pour but d'extraire la racine ou le radical d'un mot, d'où les racineurs sont des systèmes qui font l'extraction des racines des mots entrants et retourne comme résultat les préfixes, les suffixes, le schème utilisé, ainsi que la racine et le radical. Ce type d'analyse «simpliste », traite de façon identique affixes flexionnels et dérivationnels.

Le Stemming arabe présenté par Khoja est considéré par un certain nombre de chercheurs comme un Stemming standard **[30]** pour Modern Standard Arabe (MSA) et aussi pour les dialectes arabes, une des étapes principales dans le Stemming est la normalisation qui vise à transformer une copie du document original dans un format standard plus facilement manipulable. Cette étape est considérée nécessaire à cause des variations qui peuvent exister alors de l'écriture d'une même unité lexicale. Donnons comme exemple :

- ✓ Suppression des caractères spéciaux.
- ✓ Remplacement de la lettre (ة) par (ه).
- ✓ Remplacement des lettres ئ, آ, أ, إ par ا
- ✓ Remplacement de la lettre finale ي par ى.

En effet, plusieurs mots, ayant des sens différents, peuvent avoir la même forme normalisée. Nous distinguons deux types de Stemming :

- **Light Stemming** : s'intéresse seulement au Stem.

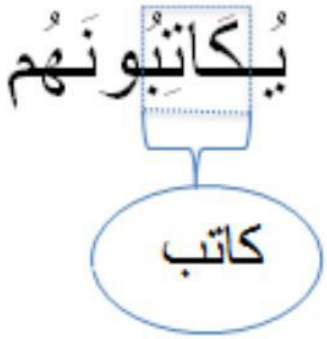

Figure 3.4 : Exemple de light Stemming

- **Heavy Stemming :** son objectif est d'extraire le Stem et la racine.

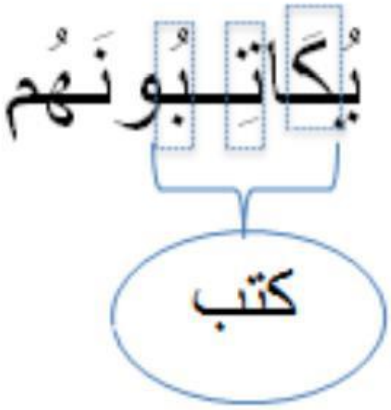

Figure 3.5 : Exemple de heavy Stemming.

5.2.6 Extracteur de termes « MAPING »

L'extracteur terminologique analyse le contenu d'un document et recherche la terminologie disponible dans les ontologies choisi.

Comme illustre la figure 3.3

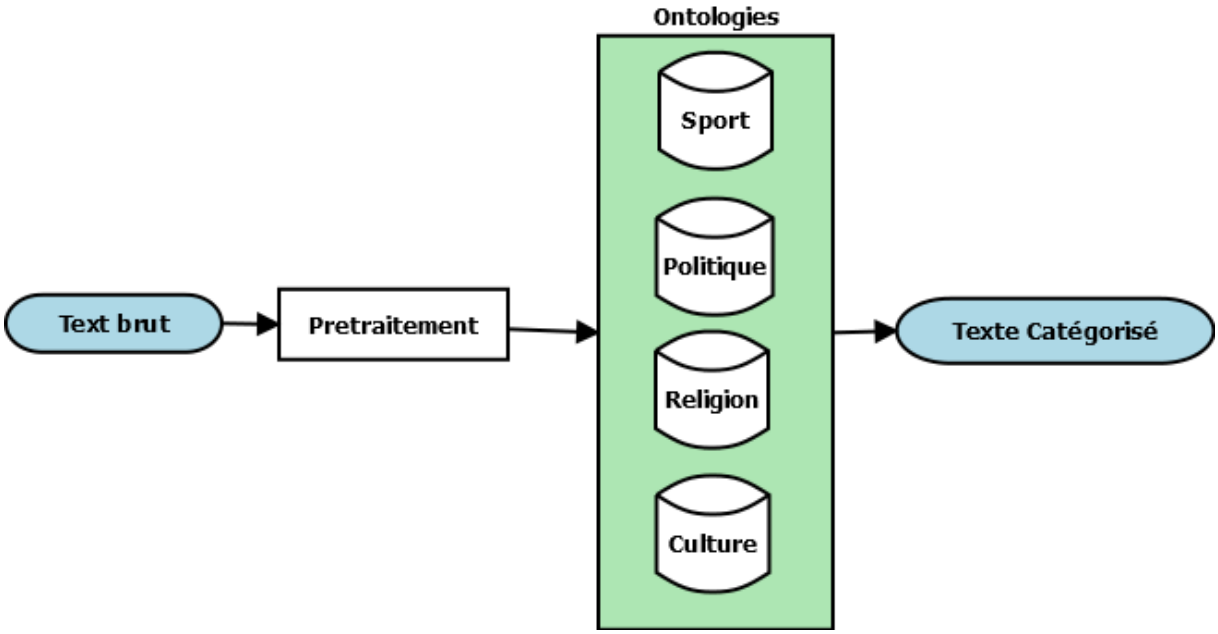

Figure 3.6 étapes de classification d'un nouveau texte.

6 Conclusion

Nous avons décrit dans ce chapitre l'architecture globale de l'application développée. Cette application nous permet de classer les textes à partir des ontologies de domaine elle offre la création et l'enrichisse des ontologies et le classement des nouveaux textes.

Chapitre04

Implémentation Et Réalisation

CHAPITRE4
IMPLEMENTATION ET REALISATION

1 Introduction

Pour répondre à la problématique posée dans le premier chapitre on a développé deux applications l'une pour la création des ontologies de domaine à partir des thesaurus et l'autre pour la classification de textes arabes basée sur ces ontologies.

Dans ce chapitre nous décrivons les outils utilisés et les différentes étapes de la réalisation de l'application.

2 Description des outils

Cette application offre de multiples fonctionnalités qui exigent l'utilisation de différents outils pour les implémenter, certains de ces outils sont communs sur tous les projets java mais d'autres sont particuliers à ce type de projets.

2.1 Java

Java est un langage de programmation qui a été créé par Sun Microsystems en 1995. Beaucoup d'applications et de sites Web ne fonctionnent pas si Java n'est pas installé, et leur nombre ne cesse de croître chaque jour. Java est rapide, sécurisé et fiable. Des ordinateurs portables aux centres de données, des consoles de jeux aux superordinateurs scientifiques, des téléphones portables à Internet, la technologie Java est présente sur tous les fronts.

En a choisi java avec sa version 8 comme langage de programmation du projet.

2.2 NetBeans

Est un environnement de développement intégré (EDI), placé en *open source* par Sun en juin 2000 sous licence CDDL (Common Development and Distribution License).

En plus de Java, NetBeans permet la prise en charge native de divers langages tels que : C, C++, JavaScript, XML, PHP et HTML. Il offre toutes les facilités d'un IDE moderne (éditeur en couleurs, projets multi-langages, refactoring, éditeur graphique d'interfaces et de pages Web).

Compilé en Java, NetBeans est disponible sous Windows, Linux, Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). Un environnement Java Development Kit JDK est requis pour les développements en Java.

2.3 Dia

Dia est un logiciel libre de création de diagramme développé en tant que partie du projet GNOME. Conçu par Alexander Larsson, il poursuit des buts similaires à Microsoft Visio et fait partie du projet GNU.

Dia est conçu de manière modulaire avec plusieurs paquetages de formes pour des besoins différents : diagramme de flux, diagramme de circuit électrique, diagramme UML, etc. L'ajout d'un paquetage se fait par l'écriture de fichiers XML, en utilisant un sous-ensemble du SVG pour dessiner les formes.

Figure 4.1 Dia, NetBeans, Java

2.4 Choix du premier corpus

Nous avons utilisé un simple corpus de textes arabe dans les domaines : Sport, Politique, Culture et Religion. Qui contient presque 1000 mots par chaque domaine. Pour construire notre ontologie de domine.

2.5 Choix du deuxième corpus « corpus de test »

Pour les besoins de certaines étapes de notre application, nous avons eu besoin de travailler sur un autre corpus, parce qu'alors nous avions besoin de comparer ou de tester les termes de notre corpus par rapport aux autres, nous avons eu recours à celui proposé par Al-KHALIDJ C'est un corpus divisé en 6 catégories dont le Sport, Religion, Local, International, Culture et économie. Il est disponible sur le web gratuit. [16]**.** Nous avons utilisé ce corpus pour tester les résultats de classement de notre application.

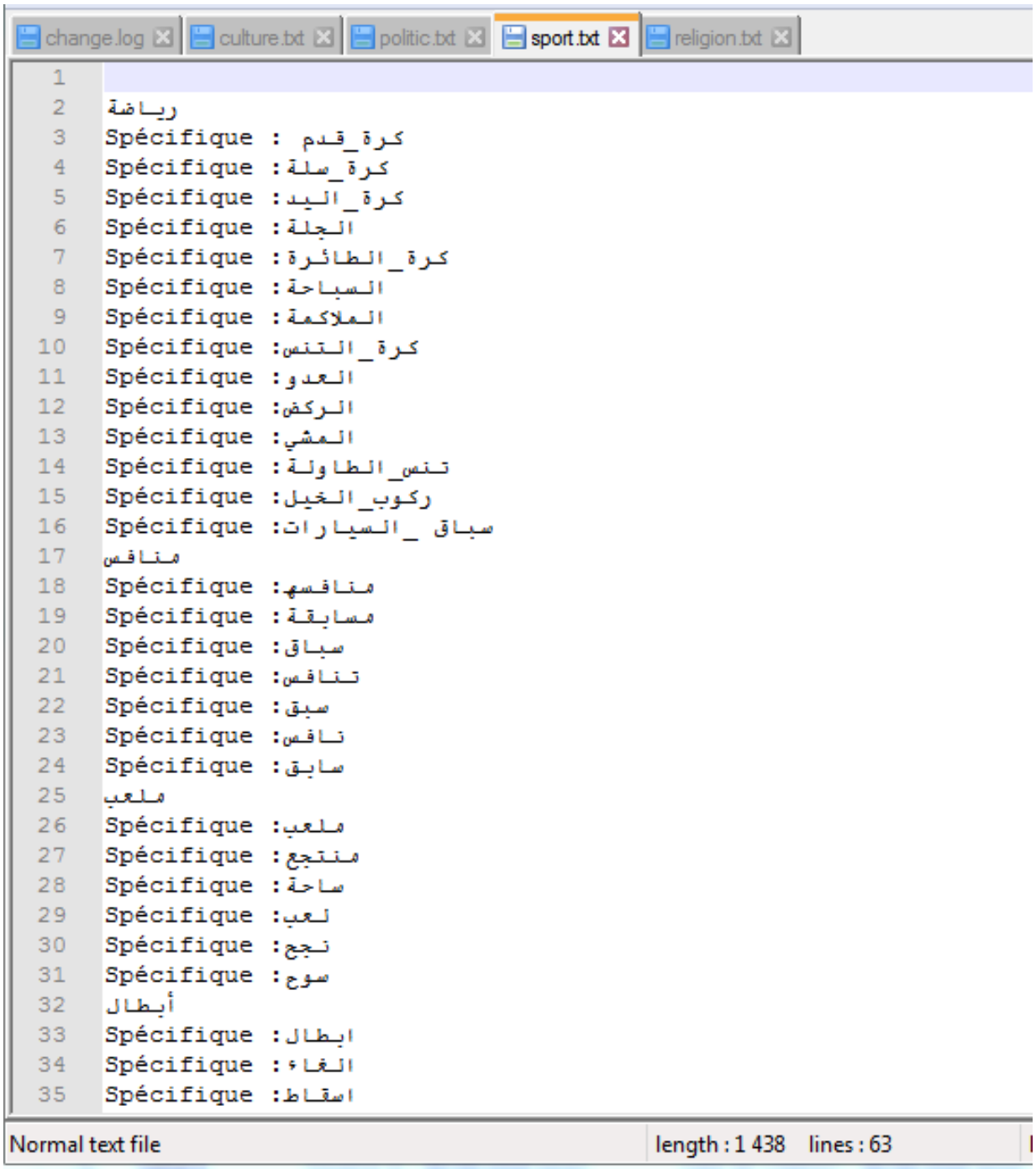

Figure 4.2 Corpus choisi.

2.6 Les mots vides

La table suivante contient les prépositions, les particules qui ont un effet de rection sur les verbes à l'accompli et l'inaccompli, les particules de coordination et d'autres particules, en résumé, tous les mots qui restent invariants quel que soit leur contexte. Notre application les éliminer à la phase de Prétraitement.

Mots Vide	Taille
وله	3
ذات	3
اي	2
بدلا	4
اليها	5
انه	3
الذين	5

Figure 4.3 Tableau Mots Vides.

3 Interfaces Graphiques

3.1 Ontologie à partir d'un Thesaurus

Cette application permet de charger un thesaurus sur format d'un fichier « txt » et le convertir en fichier XML puis en Ontologie OWL. Et nous donne le graphe résultat de l'ontologie créé.

1er étape : Chercher l'emplacement et charger le fichier : Domaine.txt

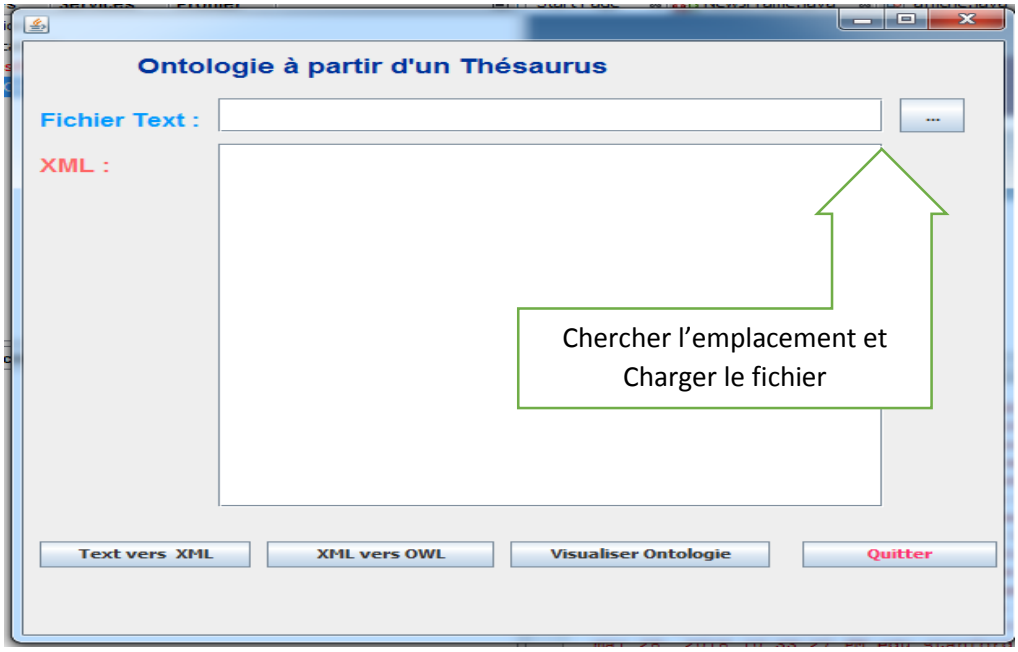

Figure 4.4 fenêtre principale de l'application.

2em étape : Convertir le fichier chargé en format XML puis en OWL

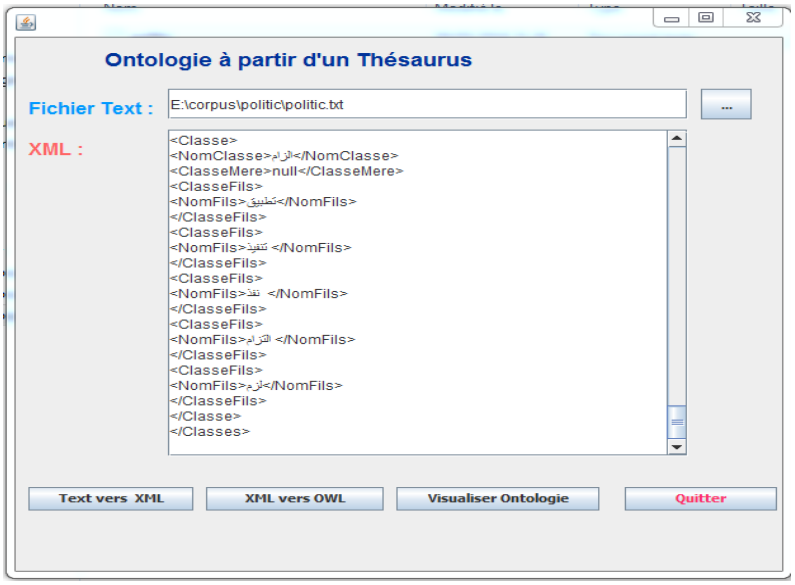

Figure 4.5 texte ver XML.

4eme la création et la sauvegarde du fichier OWL faite avec succès.

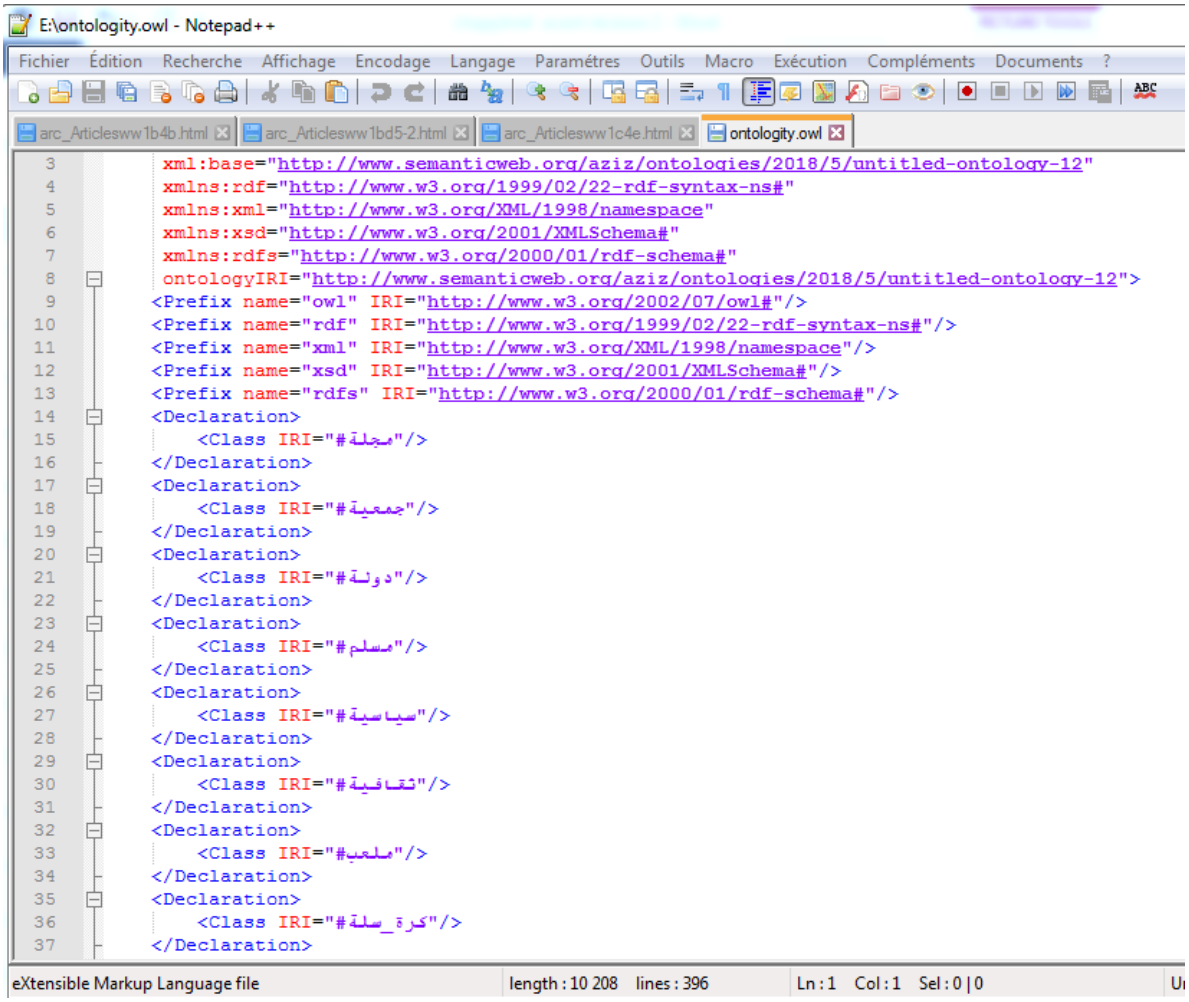

Figure 4.6 Création du fichier OWL résultat.

3.2 La 2eme application : Classification de texte arabe

3.2.1 Fenêtre principale de l'application

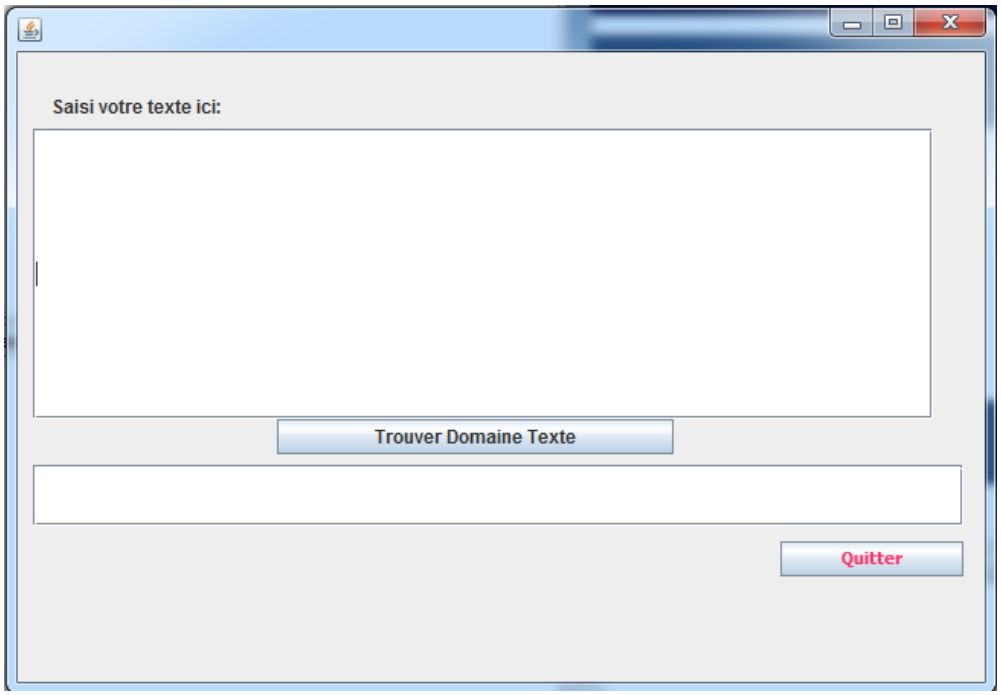

Figure 4.7 fenêtre principale.

3.2.2 Exemple de classification d'un texte :

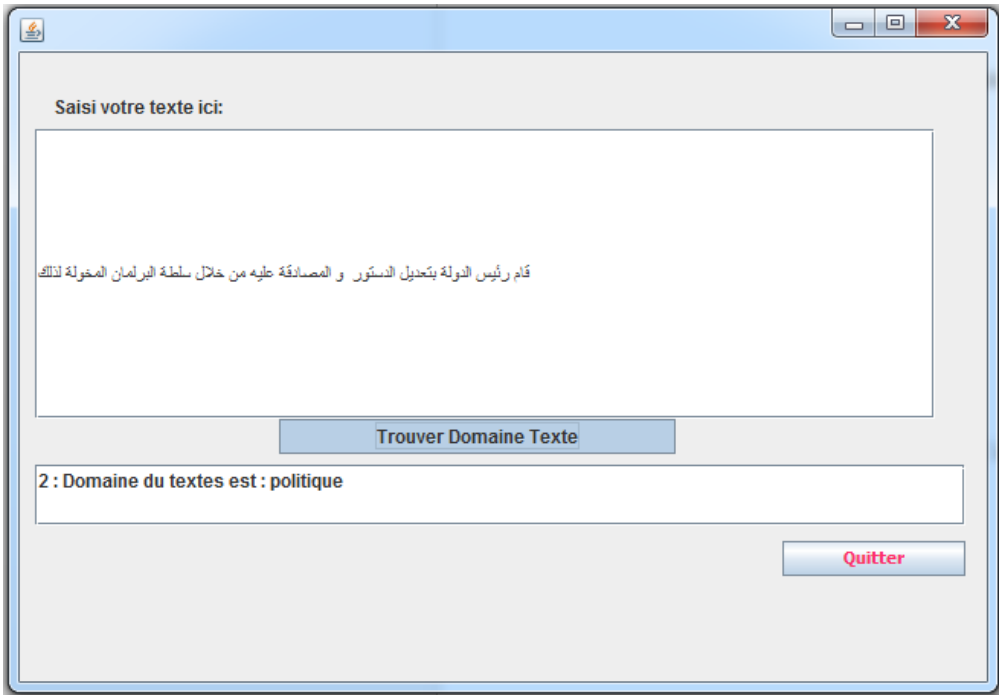

Figure 4.8 texte de domaine «politique ».

3.2.3 calculer les distances entre le nouveau texte à classifier et les différentes catégories :

```
Distance euclidienne
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     1.7976931348623157E308  0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     1.7976931348623157E308  0.0
```

Figure 4.9 Calcule distance.

4 Performance

4.1 Test et évaluation de temps

Nous avons fait un test sur le corpus de test en fonction du temps d'exécution en classifiant quelle que documents et enregistre le temps d'exécution par l'éditeur:

Document (mots)	10	20	25	30	35	40	80	160
Temps (s)	1.29	1.93	2.14	2.4	2.7	3.6	6.8	12.4

Table 4.1 test temps.

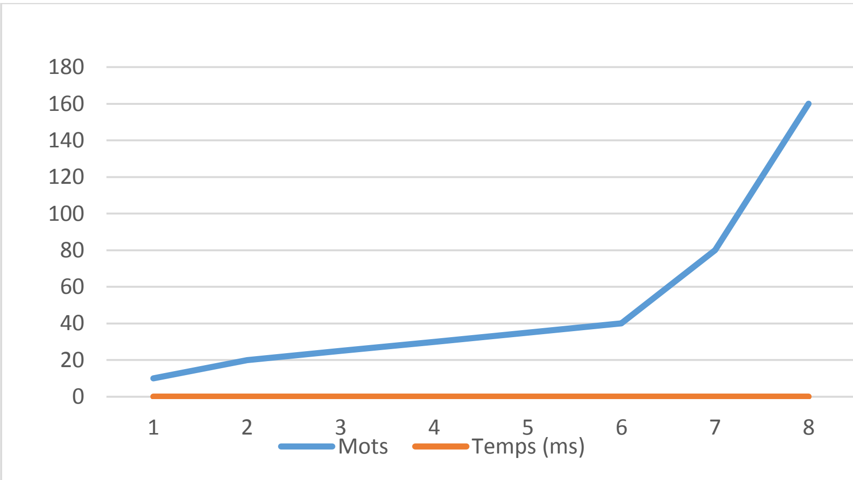

Figure 4.10 le temps d'exécution en fonction nombre de mots.

4.2 Test et évaluation de Précision

Nous avons testé notre programme avec 100 ensembles de textes dans chaque domaine dans le corpus de teste et nous avons enregistré les résultats dans le tableau suivant :

	Sport	Politique	Religion	Culture
Vrais	**72**	**75**	**86**	**77**
faux	**28**	**25**	**14**	**23**
Précision	**72%**	**75%**	**86%**	**77%**

Table 4.2 test précision

*Précision (total)= (72+75+86+77)/4=**77.5%***

4.3 Test de volume

Nous avons testé notre programme aussi avec déférents volume des textes saisi par l'utilisateur et nous observons et remarquons les taux d'erreur comme illustre la table suivant :

Nombre de mots	**10**	**40**	**100**	**150**
Taux d'erreur	0.40	0.35	0.2	0.15

Table 4.3 test volume.

4.4 Observations et conclusions

- ✓ La méthode utilisée est efficace pour tester le domaine d'un texte.
- ✓ Pour des bons résultats il faut que le texte saisi ait un volume acceptable.
- ✓ Pour optimiser la décision il faut bien enrichir notre ontologie.

5 Conclusion

Nous avons discuté les différentes fonctionnalités de l'application développée et nous avons présenté dans ce chapitre l'essentiel de notre travail qui se résume en deux parties

- ✓ La création de l'ontologie de domaine à partir du thesaurus
- ✓ La classification d'un texte arabe à partir de l'ontologie créée.

CONCLUSION GENERALE

À la lumière des résultats obtenus à partir du notre travail, nous pouvons conclure que les nouvelles approches d'analyse intelligente de documents qui utilisent les techniques d'apprentissage automatique peuvent faciliter le domaine de recherche.

Nos premières expériences dans ce travail conduisent à la représentation de texte dans un format adapté aux algorithmes d'apprentissage en utilisant souvent la représentation vectorielle. A cet effet, nous avons utilisé la méthode K-ppv, qui a donné de bons résultats.

Ce projet nous a permis de découvrir l'étendue de l'utilisation du langage java et la façon d'utiliser les différents Algorithmes d'apprentissage. D'autre part, c'était une expérience qui nous a permis d'voir comment nous pouvons utiliser des données non structurées pour mieux prendre de meilleures décisions, et le plus important nous étudié le phénomène de classification de textes arabes et les déférentes façons dont il peut être réalisé, nous avons essayé aussi de mettre en œuvre des méthodes de classification et comment nous pouvons les utiliser de manière optimale.

Conclusion Générale

CONCLUSION GENERALE

À la lumière des résultats obtenus à partir du notre travail, nous pouvons conclure que les nouvelles approches d'analyse intelligente de documents qui utilisent les techniques d'apprentissage automatique peuvent faciliter le domaine de recherche.

Nos premières expériences dans ce travail conduisent à la représentation de texte dans un format adapté aux algorithmes d'apprentissage en utilisant souvent la représentation vectorielle. A cet effet, nous avons utilisé la méthode K-ppv, qui a donné de bons résultats.

Ce projet nous a permis de découvrir l'étendue de l'utilisation du langage java et la façon d'utiliser les différents Algorithmes d'apprentissage. D'autre part, c'était une expérience qui nous a permis d'voir comment nous pouvons utiliser des données non structurées pour mieux prendre de meilleures décisions, et le plus important nous étudié le phénomène de classification de textes arabes et les déférentes façons dont il peut être réalisé, nous avons essayé aussi de mettre en œuvre des méthodes de classification et comment nous pouvons les utiliser de manière optimale.

Bibliographie

Livre

[1] Simon Réhel : « Catégorisation automatique de textes et cooccurrence de mots provenant de documents non étiquetés » Québec, Janvier 2005

[2] Radwan jalam : « Apprentissage automatique et catégorisation de textes multilingues » Université Lumière Lyon2, Juin 2003.

[3] Lebart, L. and Salem, A. (1994). Statisque textuelle. Dunod, Paris.

[4] Editors, Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval, pages 282– 289, Dublin, IE. Springer Verlag, Heidelberg, DE.

[5] Wiener, E. D., Pedersen, J. O., and Weigend, A. S. (1995). A neural network approach to topic spotting. In Proceedings of SDAIR-95.

[6] Lewis, D. D. and Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, pages 81–93, Las Vegas, US.

[7] Borko, H. and Bernick, M. (1964). Automatic document classification. part ii : additional experiments. the Association for Computing Machinery,page:138–151.

[8] Joachims,T. (1997). A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In Fisher, D. H., editor, Proceedings of ICML-97, 14th International Conference on Machine Learning, pages 143–151, Nashville, US.

[9] Lefèvre, P. (2000). La recherche d'information - du texte intégral au thésaurus. Hermès Science, Paris.

[14] Welty C., & Guarino, N., (2001). Supporting Ontological Analysis of Taxonomic Relationships Data et Knowledge Engineering (39), pages 51-74, 2001.

[15] Baneyx A., (2007). Construire une ontologie de la pneumologie, aspects théorique, modèles et expérimentations. Thèse de doctorat, Université Pierre et Marie Curie.

[16] Gruber T. (1993). A translation approch to portable ontology specifications. Knowledge acquisition, 5(2), 199–220.

[17] Charlet J., Bachimont B., Bouaud J., Zweigenbaum P., (1996). Ontologie et réutilisabilité : expérience et discussion. In N. Aussenac-Gilles, P. Laublet & C. Reynaud,

Coordinateurs,Acquisition et ingénierie des connaissances : tendances actuelles, chapitre 4, p. 69–87. Cepaduès-éditions.

[22] Uschold M. & Grüninger M. (1996). Ontologies: Principles, Methods and Applications. Knowledge Engineering Review, 11(2).

[23] Sowa J. (2000). Ontology, Metadata and Semiotics. In 8th International Conference On Conceptual Structures (Iccs'2000), Volume 1867, P. 55–81: Springer-Verlag Lncs.

[24] Nardi d., brachman r. (2003). the description logic handbook : theory, implementation and applications, chapitre an introduction to description logics., p. 544. cambridge university press.

[25] patil l., dutta d., sriram r. (2005).ontology formalization of product semantics for product lifecycle management. proc. asme/idetc cie conf., long beach, ca

[26] Gomez-pérez a. (2004). ontology evaluation, in s. staab & r. studer, coordinateurs, handbook on ontologies, chapitre, p. 251–275. handbooks in information systems. springer.

[27] Yang, Y. and Chute, C. G. (1994). An example-based mapping method for text categorization and retrieval. ACM Transactions on Information Systems, 12(3) ,252.. 277.

[28] Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998).Inductive learning algorithms and representations for text categorization. In Gardarin,G., French, J. C., editors, Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management, pages 148–155.

[30] LARKEY, BALLESTEROS, & CONNELL. Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis. . (2002).

These

[29] Mohamadally Hasan, Fomani Boris ,2006 « SVM : Machines à Vecteurs de Support ou Séparateurs à Vastes Marges » Versailles St Quentin, France URL *georges.gardarin.free.fr/Surveys_DM/Survey_SVM.pdf*

Site web

[10] Fabienpoulard : http://fabienpoulard.info/post/2008/02/21/Lalgorithme-de-Porter.html, consuler le 28/01/2018

[11] Developer : http:/developer.net/forum/generaldeveloppement/algorihtme/treeragger-ssous.html, consuler le 20/02/2018

[12] Lingue : https://www.linguee.fr/francaisanglais/traduction/ pprentissage.html, consuler le 30/01/2018

[13] Le big data : https://www.lebigdata.fr/deep-learning-definition.html, consuler le 10/02/2018

[18] Ontoknowlege : **http://www.ontoknowledge.org/oil/ ,** Consuler Le 19/01/2018

[19] Daml : **http://www.daml.org/2001/03/daml+oil-index,** Consuler Le 17/2/2018

[20] DARPA: Defense Advanced Research Projects Agency.html**,** Consuler Le 15/03/2018

[21] Manchester : http://owl.cs.manchester.ac.uk/**,** Consuler Le 12/4/2018

ملخص

الهدف من المشروع هو وضع نهج جديد لتصنيف النصوص العربية المتعلقة بمجال معين .هذا النهج هو تصميم وإنشاء أنتولوجيات المجال واستخدامها كمصدر للمعرفة ، باستغلال واحدة من خوارزميات التعلم الآلي لتحسين نتائج التصنيف.

الكلمات المفتاحية: تصنيف الوثيقة ، التصنيف الخاضع للمتابعة ، خوارزميات التعلم ، انتولوجيات المجال.

ABSTRACT

The objective of the project is to establish a new approach to classify Arabic texts related to a specific field. The approach is to design and implement a domain ontology and use it as a source of knowledge, to exploit one of the algorithms of machine learning to improve classification results.

Keywords: document classification, supervised classification, learning algorithms, domain ontology.

RESUME

L'objectif du projet est d'établir une nouvelle approche permettant de classifier des textes arabes liés à un domaine spécifique. L'approche consiste à concevoir et implémenter une ontologie de domaine et l'utiliser comme source de connaissances, d'exploiter l'un des algorithmes de l'apprentissage automatique pour améliorer les résultats de classification.

Mots-clés : classification de documents, classification supervisée, algorithmes d'apprentissage, ontologie de domaine.