REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

N° d'ordre : .....

UNIVERSITE DE M'SILA

FACULTE DES MATHEMATIQUES ET DE L'INFORMATIQUE

Département d'informatique

MEMOIRE de fin d'études

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et Informatique
Filière : Informatique
Spécialité : système d'information avenacés

Par: KHERCHI Asma

SUJET

Catégorisation contextuelle des textes arabes par la méthode Adaboost

Soutenu publiquement le : 19 / 06 /2014 devant le jury composé de :

SAOUDI Alia	**Université de M'silaPrésident**
KADRI Said	**Université de M'silaRapporteur**
BENAZI Makhlouf	**Université de M'silaExaminateur**

Promotion : 2013 /2014

Remerciements

بسم الله الرحمـــن الرحـــيم

الحمد لله رب العالمين

Je tien tout d'abord remercier notre Allah aza wa jal , qui m'avoir donné la santé, la volonté, le courage, la réussite, et de ma avoir aidés à accomplir ce modeste projet.

Nous remercions très sincèrement notre encadreur KADRI SAID pour ses remarques pertinentes et son optimisme qui nous fait parfois défaut. En outre ses qualités d'encadreur, nous ont toujours permis d'avancer à un rythme régulier dans notre travail, nous encourageant à persévérer.

Bien entendu, je tiens très fort à remercier ma mère et mon père, mon mari, pour leurs sacrifices et leur patience, tout au long de mes études ainsi que mes frères et ma sœur, mes collègues et amis pour leur soutien inconditionnel.

Je tiens à remercier mes professeurs à l'université, Ils m'ont donné des connaissances fondamentales qui sont très utiles pour poursuivre mes études.

Je tiens ici à exprimer ma très Sincère gratitude à tous ceux et toute celles qui ont pris sur le temps pour m'aider et m'entourer de leur conseils tout au long de ce travail ; ils ont contribue pour une grande part à l'aboutissement de ce mémoire.

Je termine ces remerciements en saluant vivement l'ensemble des membres de jury pour l'honneur qu'ils me font en acceptant de juger ce travail.

KHERCHI Asma

TABLE DES MATIERES

TABLE DES MATIERES

CHAPITRE 4 : REALISATION *ET EXPERIMENTATION*

LISTE

DES FIGURES, TABLEAUX, ALGORITHMES

Listes des figures

Liste des tableaux

Liste des Algorithmes

INTRODUCTION GENERALE

INTRODUCTION GENERALE

Aujourd'hui, nous vivons dans un monde où l'information est disponible en grande quantité tout en étant de qualité très diverse. Internet s'enrichit continuellement de nouveaux contenus. Par exemple, vidéos, images, articles, journaux, des documents autrefois manuscrits sont aujourd'hui disponibles sous format numérique. Mais toute cette information complexe serait sans intérêt si notre capacité à y accéder efficacement n'augmentait pas elle aussi. Pour cela, nous avons besoin d'outils permettant de chercher, classer, conserver, mettre à jour et analyser les données accessibles. Il est ainsi nécessaire de proposer des systèmes afin d'accéder rapidement à l'information désirée, réduisant ainsi l'implication humaine.

Un des domaines qui tente d'apporter des améliorations et de réduire la tâche de l'humain est la catégorisation automatique de documents. Celle-ci consiste à associer une catégorie à un document pouvant être une phrase, un paragraphe, un texte, etc.

Généralement, une catégorisation de documents est effectuée manuellement et sa réalisation est donc coûteuse en termes de temps. En effet, chaque texte (ou une partie) doit être manuellement lu pour attribuer une catégorie adaptée (classe). C'est la raison pour laquelle le domaine de la catégorisation automatique de documents est en perpétuel développement.

La catégorisation automatique de texte est l'un de ces domaines prometteurs qui a débuté dans les années 1960, les premiers résultats dans ce domaine ont surtout été motivés par les progrès technologiques à partir les années 1980.La majorité des travaux a été réalisée sur des documents écrits en caractères latins (français, anglais, espagnol, etc.). En revanche, très peu de travaux se rapportent à la classification automatique des documents arabes malgré la richesse morphologique de cette langue. Pour construire un modèle efficace de classification automatique des documents à catégoriser, le traitement automatique de ces derniers est un élément essentiel. Lors de ce traitement, la richesse morphologique de la langue arabe entraine des difficultés qui empêchent les chercheurs d'adopter directement les méthodes et les résultats obtenus par les travaux portant sur la classification automatique des documents écrits en caractères latins sans mettre en question leur validité, d'où la nécessité de tester la fiabilité et l'efficacité des solutions existantes sur des documents écrits en caractères arabes avant de pouvoir tirer des conclusions décisives.

La problématique qui nous occupe dans ce travail consiste à utiliser l'apprentissage automatique pour affecter des catégories à des documents écrits en arabe en fonction de leur contenu, afin de trouver une liaison fonctionnelle entre le texte à classer et les catégories.

Pour identifier la catégorie à laquelle un texte est associé, un ensemble d'étapes est généralement suivies : La représentation du texte, le choix de l'algorithme d'apprentissage, et l'évaluation des résultats obtenus pour garantir une bonne généralisation du modèle.

La première étape est elle-même composée de trois phases:

1. Le choix des termes : nous avons choisi une approche qui s'appuie sur les n-grammes, où l'unité du vecteur dans ce cas est considérée comme une suite des caractères consécutifs.

2. La pondération des termes : Nous avons choisi le codage TF IDF pour pondérer les termes. Ce codage est largement utilisé dans le domaine de la classification des données textuelles.

L'algorithme d'apprentissage choisi pour ce traitement est la méthode Adaboost appliqué à un corpus des textes en langue arabe.

Pour la structure du mémoire nous avons le subdiviser en quatre chapitres organisés comme suit :

Le premier chapitre donne une présentation générale sur la catégorisation automatique des textes d'une manière générale, puis la catégorisation de texte arabe en particulier, et se termine par la présentation de quelques travaux antérieurs.

Le deuxième chapitre donne une présentation générale des algorithmes d'apprentissage .

Le troisième chapitre présente d'une manière détaillée l'algorithme d'apprentissage Adaboost qui est la méthode que nous avons utilisé dans notre travail (principe, fonctionnalités).

Le quatrième chapitre est dédié au fonctionnement de l'application réalisée, suivi par une phase d'évaluation des résultats obtenus par notre classifieur implémenté afin de prouver son efficacité.

Enfin, nous concluons ce mémoire en résumant les contributions que nous avons pu apporter, et en évoquant les améliorations futures de ce travail et les perspectives de recherche dans le domaine abordé.

CHAPITRE 1

CATEGORISATION DES TEXTES

1.1. Introduction

Avec Les avancées technologiques dans les systèmes de communication électronique, et la révolution de l'Internet rendent disponibles en format électronique des documents autrefois manuscrits .De nos jours, la demande d'outils de recherche permettant de gérer ce quantité d'information, a conduit à un intérêt pour les tâches de catégorisation automatique dans l'espoir de réduire le travail humain de façon significative, et à résoudre des problèmes d'accès à l'information voulu. Dans ce contexte, la catégorisation de texte, est définie comme un processus permettant d'associer une catégorie à un texte, en fonction des informations qu'il contient. Elle a pour but d'automatiser l'association d'un texte à une catégorie.

Ce chapitre présente la définition de la catégorisation des textes ainsi que son processus, il expose aussi quelques applications de la catégorisation des textes, les méthodes d'apprentissage utilisées dans ce domaine et les difficultés qui le caractérise.

1.2. Définition de la Catégorisation de textes

Dans sa forme la plus simple, la catégorisation de documents consiste à assigner à un texte une ou plusieurs étiquettes permettant d'indexer le document dans un ensemble prédéfini de catégories, Originellement conçue pour assister le classement documentaire d'ouvrages ou d'articles dans des domaines techniques ou scientifiques.

Il existe de nombreuses définitions, souvent complémentaires, parfois contradictoires. [1] définit la catégorisation de textes comme étant la recherche d'une relation bijective qui consiste à "chercher une liaison fonctionnelle entre un ensemble de textes et un ensemble de catégories (étiquettes, classes)". [2] ajoute la notion de classes cibles prédéfinies et voit la catégorisation de textes comme étant une tâche de tri. Enfin, [3] définit la classification en ajoutant la notion de hiérarchie de classes à travers certaines propriétés communes.[4] rappelle que dans la littérature scientifique, les termes classification et catégorisation sont indifféremment utilisés. [5] définit la classification et la catégorisation comme étant l'action d'affecter des éléments, qui possèdent des caractéristiques communes, à des catégories préétablies, sans relation d'ordre.

La Catégorisation de Textes (C.T) est le processus qui consiste à assigner une ou plusieurs catégories parmi une liste prédéfinie à un document. L'objectif du processus est d'être capable d'effectuer automatiquement les classes d'un ensemble de nouveaux textes. Dans les systèmes de classification basés sur des méthodes d'apprentissage, la fonction de décision sera évaluée à l'aide d'un corpus d'entraînement. Cette fonction peut faire intervenir un grand nombre de valeurs numériques qu'un humain ne peut pas saisir. La détermination de cette fonction est appelée *phase d'apprentissage*, tandis que l'utilisation de cette fonction pour attribuer une catégorie à un document se fera pendant la phase de test.

1.3. Processus de la catégorisation automatique de textes

Pour réaliser l'opération de catégorisation de textes comme nous l'avons défini, la démarche commune est la suivante :

la première phase consiste donc à formaliser les textes afin qu'ils soient compréhensibles par la machine et utilisables par les algorithmes d'apprentissage.

la deuxième phase est La catégorisation des documents, cette étape est bien entendu décisive car c'est elle qui va permettre ou non aux techniques d'apprentissage de produire une bonne généralisation à partir des couples (Document, Classe).

Pour améliorer la performance des modèles, une évaluation de la qualité des classifieurs et la comparaison des résultats fournis par les différents modèles est effectuée en fin de cycle.

La figure 1.1 illustre la démarche de catégorisation de textes avec ses trois étapes qui peuvent être schématisées comme suit :

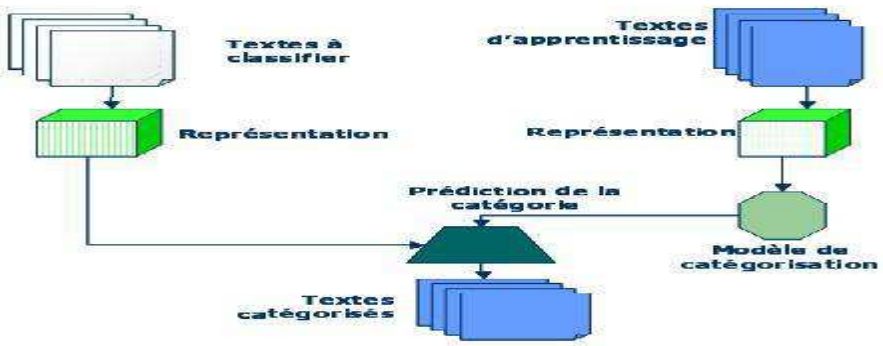

Figure 1.1 *Processus de la catégorisation de textes .*

La démarche d'une approche standard de catégorisation automatique de textes peut être résumée de la manière suivante :

- Eliminer les caractères de séparation, les signes de ponctuations, les mots vides, etc..;
- Les termes restants sont tous des attributs
- Un document devient un vecteur <terme, fréquence>
- Entrainer le modèle de classification à partir des couples (Document, Classe).
- Évaluer les résultats du classifieur sur un ensemble de test.

1.4. Représentation de textes

Afin de bien classifier les textes il est nécessaire d'utiliser une technique de représentation efficace. Les différentes méthodes qui existent pour la représentation des textes sont :

1.4.1. Représentation en sac de mots

La représentation en sac de mot(bag of words en anglais) est la plus simple représentation. Les textes sont transformés simplement en vecteurs dont chaque composante représente un terme. Dans un premier temps, les termes sont les mots qui constituent un texte. Dans les langues comme le français ou l'anglais, les mots sont séparés par des espaces ou des signes de ponctuations ; ces derniers, tout comme les chiffres, sont supprimés de la représentation. On peut choisir de conserver les majuscules pour aider, par exemple, à la reconnaissance de noms propres, mais il faut alors résoudre le problème des débuts de phrase. Les composantes du vecteur sont une fonction de l'occurrence des mots dans le texte .Nous présentons ci-dessous un exemple de représentation par sac de mot (tableau 1.1) :

Marionnaud: Union et Etudes Investissement franchit5% des droits de vote
PARIS, 31 juil (AFP) - La société Union et Etudes Investissement (caisse Nationale de Crédit Agricole) a franchi en hausse le seuil de 5% des droits de vote du groupement français de parfumerie Marionnaud et détient désormais 292.157 actions, soit 8,09% du capital et5,05% des droits de vote, a indiqué vendredi le Conseil des Marchés Financiers.
Ce franchissement de seuil résulte de l'acquisition de 11.460 actions, précise le CMF.

Mot	Occur.	Mot	Occur.	Mot	Occur.
A	2	détient	1	le	3
acquisition	1	en	1	marchés	1
actions	2	et	4	marionnaud	2
agricole	1	études	2	nationale	1
caisse	1	financiers	1	parfumerie	1
capital	1	franchi	1	précise	1
ce	1	franchissement	1	résulte	1
cmf	1	franchit	1	seuil	2
conseil	1	français	1	société	1
crédit	1	groupement	1	soit	1
de	9	hausse	1	union	2
des	4	indiqué	1	vendredi	1
droits	3	investissement	2	vote	3
du	2	l	1		
désormais	1	la	1		

Table 1.1 *Exemple de la représentation de textes en « sac de mots »*

Les chiffres et dates sont supprimés de la représentation, Cette représentation des textes exclut toute analyse grammaticale et toute notion de distance entre les mots : c'est pourquoi cette représentation est appelée "sac de mots".

1.4.2. Représentation par des phrases

Un certain nombre de chercheurs proposent d'utiliser les phrases comme unité de représentation au lieu des mots comme le cas dans la représentation « sac de mot », puisque les phrases sont plus informatives que les mots seuls, par exemple « recherche d'information », « world wide web », ont un degré plus petit d'ambiguïté que les mots constitutifs, et aussi que les phrases ont l'avantage de conserver l'information relative à la position du mot dans la phrase [6][7] .

1.4.3. Représentation avec des racines

Cette méthode consiste à remplacer les mots du document par leurs racines lexicales, qui peut être réalisée en utilisant un des algorithmes les plus connus pour la langue anglaise qui est l'algorithme de Porter [8] de normalisation de mots qui

sert à supprimer les affixes de ces derniers pour obtenir une forme canonique. Cette méthode a comme avantage de regrouper les différentes flexions d'un mot dans une seule composante, et comme inconvénient la perte de sens car la racine extraite peut être commune à des mots se rapportant à des concepts différents. A titre d'exemple : les mots vol, volant, vole ont la même racine vol mais se rendent à trois notions différentes.

1.4.4. Représentation avec des lemmes

Cette méthode (lemmatisation) consiste à remplacer les mots du document par leurs lemmes, elle doit utiliser l'analyse grammaticale afin de remplacer les verbes par leurs formes infinitives et les noms par leurs formes au singulier. En effet, Un mot donné peut avoir différentes formes dans un texte, mais leur sens reste le même. Par exemple, les mots vol, volant et vole seront remplacés par leurs lemmes : vol, volant et voler selon le contexte. Cette représentation est simple mais elle peut causer une perte d'informations donnée par le contexte nécessaire à la distinction des lemmes polysémiques (possèdent plusieurs sens) et la présence de synonymes, considérés comme des lemmes différents même s'ils font référence au même concept [9].

1.4.5. Représentation basées sur les n-grammes

La notion de n-grammes et plus particulièrement bi-grammes et trigrammes (c'est -à-dire avec respectivement n=2 et n=3) est apparue à l'origine dans [10] selon [11]. Ce dernier a introduit la notion de n-grammes dans le cadre de systèmes de prédiction de caractères en fonction des autres caractères précédemment entrés. La notion n-gramme de X définit comme une séquence de n X consécutifs. X peut alors être un caractère ou bien un mot [12].

La figure 1.2 illustre la construction de n-grammes de caractères et de mots par la notion de déplacement de fenêtre. Ce déplacement se fait par étape, une étape étant soit un caractère ou bien un mot. Les caractères (ou mots) contenus dans la fenêtre ainsi définie constituent les descripteurs d'un corpus. Nous avons par exemple dans la figure 1.2 les descripteurs de la phrase sous forme de bi-grammes de mots qui sont : "Le choix", "choix du" et "du descripteur".

Nous présentons ci-dessous les deux types de n-grammes, caractères et mots

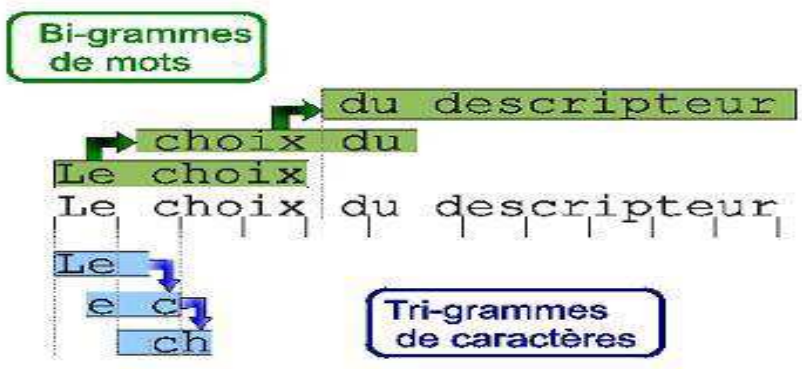

Figure 1.2 *Exemple de N-grammes de mots et de caractères.*

Les n-grammes de caractères prennent en considération les espaces. En effet, il est assez trivial de montrer que la non prise en compte des espaces introduit du bruit. Les n-grammes de caractères sont les premiers à avoir été utilisés pour une tâche utilisant des données textuelles. Les n-grammes de caractères sont très utilisé dans l'identification de la langue ou encore la recherche documentaire.

1.4.6. Représentation conceptuelle

Cette méthode consiste à représenter le document sous forme d'un ensemble de concepts, ces concepts peuvent être capturés en utilisant les réseaux sémantiques ou les sous arbres (un sous arbre représente une hiérarchie de concepts). Cette méthode a comme avantage selon REHEL dans [4] de réduire l'espace de travail car les mots qui sont synonymes partagent au moins un concept. Cependant, l'inconvénient majeur de cette représentation est qu'il n'existe pas des bases lexicales pour toutes les langues.

1.5. Pondération des termes

La pondération des termes permet de mesurer l'importance d'un terme dans un document. Cette importance est souvent calculée à partir de considérations et interprétations statistiques (ou parfois linguistiques). L'objectif est de trouver les termes qui représentent le mieux le contenu d'un document. Il existe plusieurs méthodes des pondération des termes comme : Représentation binaire, TF , TF IDF , TFC , Lnu, L'entropie, Les méthodes les plus populaires sont :

1.5.1. Représentation binaire

Elle est appelée représentation « par mots clés ». La méthode consiste à transformer le texte en un vecteur dont les éléments renseignent sur la présence (valeur égale à 1) ou l'absence (valeur égale à 0) d'un terme dans le texte. Cette façon de représenter un texte, est peu informative car elle ne donne pas les informations nécessaires ni sur les occurrences d'un terme dans le document qui peut être une information importante pour l'opération de classification, ni sur la longueur du texte.

1.5.2. Codage TF

Cette mesure est proportionnelle à la fréquence du terme dans le document (pondération locale). selon l'une des formules suivantes :

$$TF\ (t,\ d) = f(t,d)/Max[f(t,d)]$$

$$TF = log(f(t,d)) \qquad (1.1)$$

$$TF = log(f(t,d) + 1)$$

avec :

f(t,d) : fréquence du terme t dans le document d (nombre d'apparition de t dans d).

1.5.3. Codage TF IDF

Le poids d'un terme ***T*** dans un document ***D*** est calculé comme suit :

$$TFIDF(T, D) = TF(T, D) * log\ (N / DF(T)) \qquad (1.2)$$

Avec :

TF(T, D) : la fréquence du terme dans le document,

N : le nombre total de documents de la base documentaire et

DF(T) : le nombre de documents contenant le terme.

1.6. Réduction de la taille du vocabulaire

Vu la taille impressionnante des bases textuelles, il est difficile de prendre l'ensemble de tous les mots comme étant des attributs, en effet cela engendre une perte de mémoire et de temps de calcul.

Plusieurs techniques de réduction existent pour réduire la dimension de vocabulaire qui se divise en deux grandes familles :

- **Sélection d'attributs**: (feature selection) prend les attributs (ou mots) d'origine et conserve seulement ceux jugés utiles à la catégorisation, selon une certaine fonction d'évaluation et les autres sont rejetés.

- **Extraction d'attributs**: (feature extraction) à partir des attributs de départ, elles créent de nouveaux attributs en faisant soit des regroupements ou des transformations.

1.7. Méthodes de classification

La variété des domaines d'application de la classification a donné naissance à plusieurs méthodes de classification afin d'obtenir des résultats satisfaisants. Elles ont un but commun qui consiste à minimiser la distance entre deux individus d'une même classe et à maximiser la distance entres deux individus de classes distinctes. Les méthodes de classification se décomposent de la manière suivante :

1.7.1. Classification non supervisée

Dans la classification non supervisée (***clustering***), aucune information n'est fournie à la méthode (les classes ne sont pas étiquetées). De ce fait, nous cherchons à découvrir de nouvelles classes ex : K-means , la Segmentation Hiérarchique Ascendante,…).

1.7.2. Classification supervisée

Dans la classification supervisée (***catégorisation***), les classes sont connues d'avance, il suffit de répartir les nouveaux objets sur les différentes classes.

Il existe plusieurs méthodes de classification supervisée ; les plus connues sont : les SVM , la Méthode de Rocchio, Naive bayes, Les arbes de décision, les Réseaux de neurones, la méthode de plus proches voisins K-ppv , AdaBoost…,qui nous allons présenter dans la prochain section.

1.8. Catégorisation du texte arabe

La langue arabe est une langue orientale, sémitique et qui s'écrit de droite à gauche. Elle est composée de 28 consonnes, de 8 signes diacritiques ainsi que de

quelques symboles supplémentaires (<~, Madda>, <ء, Hamza>, ...). L'arabe est une langue très riche et différente des langues occidentales et ceci à plusieurs niveaux. Par exemple, l'écrit de cette langue est sensible au contexte. En effet, certaines lettres sont écrites différemment selon leurs positions dans le mot ou selon les lettres qui les entourent. De plus, la signification d'un mot n'est décelée que par son contexte, vue qu'un même mot peut avoir plusieurs sens. Ci pour ça que la classification des textes arabe est difficile par rapport les autres langues, nous présentons ici quelques propriétés morphologiques de l'arabe et quelques problèmes qui permettent de traiter les textes arabes pour avoir une meilleure classification.

1.8.1. Les propriétés morphologiques de l'arabe

La langue arabe a une morphologie riche et différente, par rapport aux langues occidentales. L'analyse morphologique d'un mot arabe, consiste principalement à déterminer la structure générale de ce mot, s'il existe, et les autres éléments utilisés pour construire ce mot . Les éléments essentiels de la morphologie de la langue arabe sont :

1.8.1.1. Les racines

Les racines sont à l'origine de la plupart de mots arabes. Elles sont des verbes formés de trois à cinq lettres consonnes Elles sont aux alentours de 10000 racines dont la grande majorité (85%) sont trilatérales. Les restes sont des racines quadrilatérales ou quintilatérales. Une racine définit la signification fondamentale des mots dérivés en utilisant différents diacritiques et affixes avec les lettres de la racine pour créer l'inflexion de la signification, Par exemple, la racine " كتب "(il a écrit) Plusieurs mots sont dérivés à partir de cette racine (tableau 1.2)

كتب	il a écrit	كاتب	auteur
كتابة	écriture	كتاب	livre
مكتوب	lettre	مكتب	bureau
إكتتاب	enregistrement		

Table 1.2 *Quelques dérivations de la racine "كتب".*

1.8.1.2. Les modèles

Le modèle arabe permet essentiellement de déterminer la structure de la plupart des mots (les noms, les verbes conjugués, etc.). Les modèles sont des déclinaisons du mot <فعل , faire> qui sont obtenus en utilisant des diacritiques ou en y ajoutant des affixes. Par exemple, le modèle < فُعل , a été fait> est obtenu en utilisant les diacritiques , le modèle <مستفعل > est obtenu en y ajoutant le préfixe" مستـ ". Par exemple Le mot "كتب"(tableau 1.3).

كتب	كُتب	مستكتب
écrit	a été écrit	une personne employée pour écrire

Table 1.3 *Quelques* structures de mots " كتب ".

1.8.1.3. La forme de mot

La morphologie de la langue arabe est très compliquée puisque la forme des mots arabes peut avoir 4 catégories d'affixes : les antéfixes (sont généralement des prépositions), les préfixes (représentés par une seule lettre indiquent la personne de la conjugaison des verbes au présent(<"الـ" , Le >,<"بـ", Avec >,<"لـ", Pour >,…)), les suffixes (sont les terminaisons de conjugaison des verbes et les signes du pluriel et du féminin pour les noms(<"كمـا" , Votre/vous >,< " ه ,ـه ", lui/son >,<" ـني ", Moi/mon >,…)),) et les post fixes (représentent des pronoms).

Antéfixe	**Préfixe**	**Noyau**	**Suffixe**	**Pos fixe**
ل	**ي**	**صنف**	**ون**	**هم**
préposition signifiant (pour que)	**Une lettre signifiant le temps et la personne de conjugaison**	/	**terminaisons de conjugaison**	**Pronom Signifiant (eux)**

Table 1. 4 *exemple d'un mot arabe composé de 4 affixes.*

1.8.1.4. Signes diacritiques

Les signes diacritiques sont des signes ajoutés au dessus ou en dessous des lettres arabes afin de spécifier la prononciation du mot. Ce rôle phonologique influe aussi sur le sens de ce mot. En effet, deux mots peuvent être écrits de la même manière mais différencié par l'ajout des signes diacritiques différents. (Par exemple le mot "درس" ,"دَرْسٌ" leçon,"دَرَسَ" il a étudie).

1.8.1.5. Mots isolés

Les mots isolés sont les mots qui n'ont pas des racines. Les mots sont en général, les noms propres, les noms communs et les particules.(Exemple : <"بلد", pays>, <"إنسان", personne>, <"حيوان", animal>, etc. » , les noms propres <"محمد", Mouhamed>..., les particules : <"بعد" ,après> <"فوق", au dessus>..., les cinq noms<"أب", un père > …).

1.8.2. Les problèmes liés au traitement automatique de l'arabe

Vu ses particularités, le traitement automatique de l'Arabe, fait face à un certain nombre de problèmes, les plus importants sont le problème de la voyellation, l'agglutination et l'extraction de la racine.

1.8.2.1. Ambiguïtés dérivationnelles et flexionnelles

La flexion est la variation de la forme des mots en fonction de facteurs grammaticaux tel que la conjugaison pour les verbes (exemple : le mot " يتـأثّــرون " (ils s'influencent) est le résultat de la concaténation du préfixe "ي"indiquant le présent et du suffixe " و ن " indiquant le masculin pluriel du verbe " تـأثّـر "). Le problème en analyse morphologique de l'arabe se rapporte surtout au niveau de la dérivation qui est un phénomène plus complexe que la flexion. En effet, la dérivation est la formation de nouveaux mots à partir de mots existants. Dans le cas de la langue arabe, la plupart des mots sont dérivés à partir de racines trilitères ou quadrilatères. Le mot arabe n'est pas le résultat d'une simple concaténation de morphèmes comme c'est le cas pour l'anglais (exemple : unfailingly = un+fail+ing+ly), mais c'est à partir d'une racine, d'une combinaison de voyelles, de préfixes, d'infixes, de suffixes et d'un schème morphologique qu'on obtient un mot (exemple : à partir de la racine " أثـر "

(choisir/citer à) on peut dériver plusieurs verbes tel que " تأثّـر " (s'influencer) et plusieurs noms tel que " متأثّـر " (ému)).

1.8.2.2. Ambiguïtés d'agglutination

Contrairement aux langues latines, en arabe, les articles, les prépositions, les pronoms, etc. collent aux adjectifs, noms, verbes et particules auxquels ils se rapportent. Comparé au français, un mot arabe peut parfois correspondre à une phrase française [13]exemple : le mot arabe " أتتذكّـروننا " correspond en français à la phrase « Est-ce que vous vous souvenez de nous »). Cette caractéristique engendre une ambiguïté morphologique au cours de l'analyse. En effet, il n'est pas toujours facile de distinguer un proclitique ou enclitique d'un caractère original du mot. Par exemple, le caractère " و " dans le mot " وصل " (il est arrivé) est un caractère original alors que dans le mot " وفتـح " (et il a ouvert), il s'agit plutôt d'une proclitique.

1.8.2.3. Problèmes d'absence des voyelles

La morphologie arabe est assez régulière lorsque les mots sont présentés sous leurs formes voyellées. Cependant, la majorité des documents arabes sont non voyellés sauf pour le Coran et pour certains ouvrages scolaires pour débutants et donc c'est pour cette raison que nous nous sommes intéressés à l'arabe non voyellé. En fait, les mots non voyellés engendrent beaucoup de cas ambigus au cours de l'analyse (exemple : le mot non voyellé " فصل " pris hors contexte peut être un verbe au passé conjugué à la troisième personne du singulier" فَصَلَ " (il a licencié), ou un nom masculin singulier "فَصْلٌ"(chapitre/ saison), ou encore une concaténation de la conjonction de coordination "فَـ " (puis) avec le verbe" صـل" : impératif du verbe lier conjugué à la deuxième personne du singulier masculin).

1.8.2.4. Problèmes de l'ordre des mots dans la phrase

En passant à un niveau supérieur au mot, à savoir la phrase nous remarquons que l'ordre des mots en arabe est variable. D'une manière générale, on met au début de la phrase le mot sur lequel on veut attirer l'attention et l'on termine sur le terme le plus long ou le plus riche en sens ou en sonorité. Cet ordre, relativement libre des mots, provoque des ambiguïtés syntaxiques artificielles dans la mesure où il faut prévoir dans

la grammaire toutes les règles de combinaisons possibles d'inversion de l'ordre des mots dans la phrase.

1.9. Les difficultés de la catégorisation de textes

Plusieurs difficultés peuvent s'opposer au processus de catégorisation de textes. Des problèmes connus dans la discipline liés à l'apprentissage automatique supervisé comme la subjectivité de la décision prise par les experts, le sur-apprentissage, etc.. mais aussi des problèmes particuliers liés à la nature des données traitées à savoir des données textuelles comme la polysémie, la redondance, Les variations morphologiques ou même L'homographie, etc..

Dans ce qui suit, nous décrivons brièvement quelque difficulté qui s'oppose à la catégorisation de textes :

1.9.1. Les pièges de la langue naturelle

On précise rarement le sens des mots que l'on emploie d'autant plus qu'un mot peut avoir plusieurs sens : «*pollachôs legetaï*» disait Aristote, les choses « *se disent en plusieurs sens* ». Selon [14], la langue naturelle est équivoque, d'où naissent les problèmes de :

- *L'implicite*, puisque tout n'est pas exprimé dans le discours et par la suite il est impossible de le prendre en compte par des logiciels.
- *La redondance*, puisqu'il existe plusieurs façons d'exprimer la même idée du fait qu'il existe beaucoup de mots ou d'expressions différentes ayant le même sens, ou des sens voisins (la synonymie), des expressions équivalentes mais de structures ou de termes différents (la paraphrase), et à l'incompatibilité entre le sens propre d'un mot (dénotation) et son sens dans un contexte particulier (connotation).
- *L'ambigüité*, car ce qui est exprimé possède souvent plusieurs interprétations comme dans les cas de l'homonymie, la polysémie, et l'homotaxie (une même syntaxe recouvrant des réalités différentes).

1.9.2. Subjectivité de la décision

Contrairement, à d'autres situations où l'appartenance à une classe est objective (un client achète, ou non, tel produit ; un patient a, ou n'a pas, tel microbe), l'attribution d'une catégorie à un texte est subjective [15]. En effet, la catégorie est attribuée en

fonction du contenu sémantique de ce texte, qui est une notion subjective, et dépend du jugement d'un expert. Souvent les experts ne sont pas d'accord sur la classe d'appartenance d'un document [16] , on parle de « inter-indexer inconsistency » [14].

1.9.3. Dimension de l'espace d'apprentissage

Puisque l'espace d'apprentissage est composé dans la plupart du temps d'un très grand nombre de documents, cette approche génère généralement, une énorme matrice. A présent, il n'existe aucun algorithme d'apprentissage capable d'exploiter une gigantesque matrice telle qu'elle est sans que cette dernière n'affecte négativement sa performance et sa fidélité et même parfois le rendre inopérable. Pour cela, il est indispensable de réduire la taille de cette matrice, avant de pouvoir l'utiliser. Néanmoins, la réduction de dimension ne doit pas être sur-appliquée afin d'éviter de supprimer des attributs pertinents [1]. Comme il n'existe aucune règle générale qui décrit à quel point il faut réduire et quelle est la dimension qui donne les meilleurs performances et résultats, seules les expérimentations par tâtonnement peuvent nous l'indiquer.

1.9.4. Sur-apprentissage

Le nombre de termes très important et très varié qui ne se répètent dans tous les textes va causer énormément de creux dans le tableau de grande dimension (textes*termes) qui peut provoquer du sur-apprentissage qui s'explique par le fait que le modèle n'arrive pas à bien classer les nouveaux textes, pourtant il l'a bien fait dans la phase d'apprentissage en classant correctement les textes de la base d'apprentissage.
Pour limiter le sur-apprentissage, on doit sélectionner des termes pour réduire la dimensionnalité. D'après les expériences antérieures, le nombre de termes doit être limité par rapport au nombre de textes de la base d'apprentissage.

Quelques auteurs recommandent d'utiliser au moins 50 à100 fois plus de textes que de termes. En général le nombre de textes d'apprentissage est limité, c'est pour cela on cherche à agir sur le nombre des termes utilisés en les diminuant, pour éviter ce sur-apprentissage. Sans bien sûr pénaliser le système en supprimant des termes pertinents [18].

1.9.5. L'imprécision des fréquences

Puisqu'on traite des milliers de documents, on se retrouve face à un très grand nombre d'attributs appartenant à des documents. Ces attributs se reproduisent rarement dans chacun de ces documents. Par conséquent, les cellules du tableau croisé contiennent souvent de petites valeurs, voire dans la plupart du temps, une valeur de 0. Ces valeurs correspondent aux poids de chaque mot dans le document qui le contient. Le calcul du poids d'un mot dépend souvent de sa fréquence. Selon [18], les poids *« suivent approximativement des lois de Poisson; le coefficient de variation (CV = écart-type/ moyenne) donne une indication sur la précision relative de l'estimation de la fréquence dans une cellule du tableau croisé; pour une loi de Poisson de moyenne m, la variance est aussi égale à m; le coefficient de variation est donc* $CV = \sqrt{\frac{m}{m}} = \frac{1}{m}$ *si m est petit, le CV est grand et la fréquence est donc imprécise ».*

1.10. Les applications de la catégorisation de texte

Habituellement, les catégories font référence aux sujets des textes, mais pour des applications particulières, elles peuvent prendre d'autres formes et on peut résoudre, par des techniques de catégorisation des problèmes tels que l'identification de la langue d'un document, le filtrage et la détection des spams (les courriers indésirables) afin de les supprimer, par exemple :

[19] le classificateur naïve Bayes qui est utilisé pour détecter automatiquement le spam.

Actuellement, les grand entreprise ont besoin de gérer rapidement et efficacement le flux d'information pour la satisfaction des clients, plusieurs travaux de recherche existent, destinées à créer des outils informatiques et des ressources génériques pour la classification, le routage et l'acheminement des courriels vers leurs destinataire ces travaux cherchent aussi à développer un processus puissant de filtrage [20] ,

il existe également d'autres applications comme la désambiguïsation des termes [21],

la catégorisation des documents multimédia, l'indexation automatique des textes [22], et l'organisation des documents [21] ,

[23] Ce travail intéressé à l'analyse des textes arabe. En premier lieu une segmentation est effectuée, puis appliqué une analyse morphologique détaillée spécifique à la langue arabe avec comme résultats les tableaux lexicaux ; ces derniers sont affinés par une

analyse syntaxique puis des calculs statistiques sont appliqués pour favoriser certains mots par rapports à d'autres. Les tableaux lexicaux sont exploités par les techniques de fouille de données.

1.11. Conclusion

Ce chapitre introductif a présenté et défini le processus de la catégorisation qui se déroule en trois phases dont la première est la représentation de texte, c'est une étape très importante, deux variables de cette phase affectent souvent les performances des résultats : le choix du terme (mot, lemme, N-gramme, concepts) et le choix du codage. La deuxième phase concerne le choix du l'algorithme d'apprentissage et la troisièmes est représente par la méthode d'évaluation du modèle. la notion de classification automatique des documents ainsi que ses applications qui sont devenues des aspects majeurs et distingués dans le domaine de la recherche d'information.

CHAPITRE 2

ALGORITHMES D'APPRENTISSAGE

2.1. Introduction

L'évolution technologique amène à des acquisitions de données de plus en plus volumineuses (document, images, résultats de mesure, etc.) qui nécessitent l'utilisation de techniques permettant d'en extraire la connaissance utile. La classification et l'apprentissage automatique qui cherchent à transformer les données brutes en connaissances plus structurées, fournissent des outils adaptés à ce type de problème qui sont nommés les algorithme d'apprentissage (méthodes d'apprentissage).

Dans le domaine de la catégorisation de textes, plusieurs types de classificateurs ont été mis au point. Dans ce chapitre, nous présentons l'apprentissage automatique, ainsi que quelques algorithmes d'apprentissage automatique utilisés dans la catégorisation de textes.

2.2. L'apprentissage automatique et la classification de textes

Puisque l'approche manuelle de classification de textes est coûteuse en temps de travail, peu générique, et relativement peu efficace, l'autre solution a été admise, qui consiste à faire apprendre automatiquement à l'ordinateur, sur la base d'un corpus de textes qui servent d'exemples, les paramètres de la fonction de classement. Ainsi depuis une quinzaine d'années la classification de textes a été considérée comme un problème d'apprentissage automatique et est rapidement devenue un champ d'essai sollicité par les différentes techniques de classification.

De toute façon, quelle que soit l'approche retenue, une des particularités de cette tâche est la très grande dimensionnalité de l'espace dans lequel les textes sont représentés, qui comprend généralement plusieurs milliers de termes.

L'apprentissage automatique s'intéresse aux méthodes inductives permettant d'acquérir des connaissances à partir d'observations d'un phénomène. Cette connaissance peut être exploitée pour des tâches de décision ou de prévision : c'est le cadre de l'apprentissage supervisé ; ou à des fins d'analyse exploratoire ou de structuration d'un ensemble de données : c'est le cadre de l'apprentissage non supervisé [24].

2.3. L'apprentissage non supervisé

L'apprentissage non supervisé consiste à apprendre à classer sans supervision. Au début de processus nous ne disposons ni de la définition des classes, ni de leurs nombres. C'est l'algorithme de classification qui va déterminer ces informations. Nous ne disposons pas non plus de données en entrée qui sont déjà classées, c'est aussi à l'algorithme de découvrir par lui-même la structure plus ou moins cachée des données et de former des groupes d'individus dont les caractéristiques sont communes. [25]

il existe plusieurs types d'algorithmes d'apprentissage non supervisé tels que :

- les algorithmes de classification hiérarchique ,il existe deux types de classification hiérarchique Ascendante et descendent.
- les algorithmes de partitionnements :Le partitionnement consiste au regroupement des données suivant leur degré de similarité. L'algorithme le plus célèbre appartenant à cette classe est K-means

K-means : c'est un algorithme qui permet de partitionner un ensemble de données automatiquement en K clusters. Il consiste tout d'abord à choisir k points qui représentent les centres des groupes à créer, puis à affecter les autres points aux centres les plus proches. Cette affectation est faite par le calcul de distance entre les points. Plusieurs distances peuvent être définies telles que la distance euclidienne ou la distance de Manhattan. Par la suite nous procédons à une étape de raffinement des groupes de façon itérative, le raffinement se fait par le recalcule des centres des groupes après chaque itération et par une réaffectation des points aux groupes. L'algorithme s'arrête quand aucun point ne bouge. [26]

2.4. Apprentissage supervisé

L'apprentissage supervisé est une catégorie d'algorithme par apprentissage qui vise à produire des règles en s'appuyant sur une base de données d'apprentissage (souvent des résultats d'expériences précédentes). Dans la plupart des catégories d'algorithmes vues dans la partie précédente, la première étape est celle de la classification durant laquelle on « étiquette » chaque donnée en l'associant à une classe. C'est le cas de l'apprentissage supervisé. Dans un premier temps, un « professeur » (personne apte à étiqueter les exemples) génère un ensemble de couples entrée-sortie : c'est la phase d'apprentissage. Ensuite, dans une phase de test, le système tente de prédire

la sortie connaissant l'entrée. On retrouve ce comportement lors de diagnostics médicaux. Ceux-ci s'appuient sur une classification de couple symptômes – maladie qui est utilisée dans le but de trouver une maladie (sortie) associée à des symptômes (entrée). Voici un schéma résumant l'apprentissage supervisé :

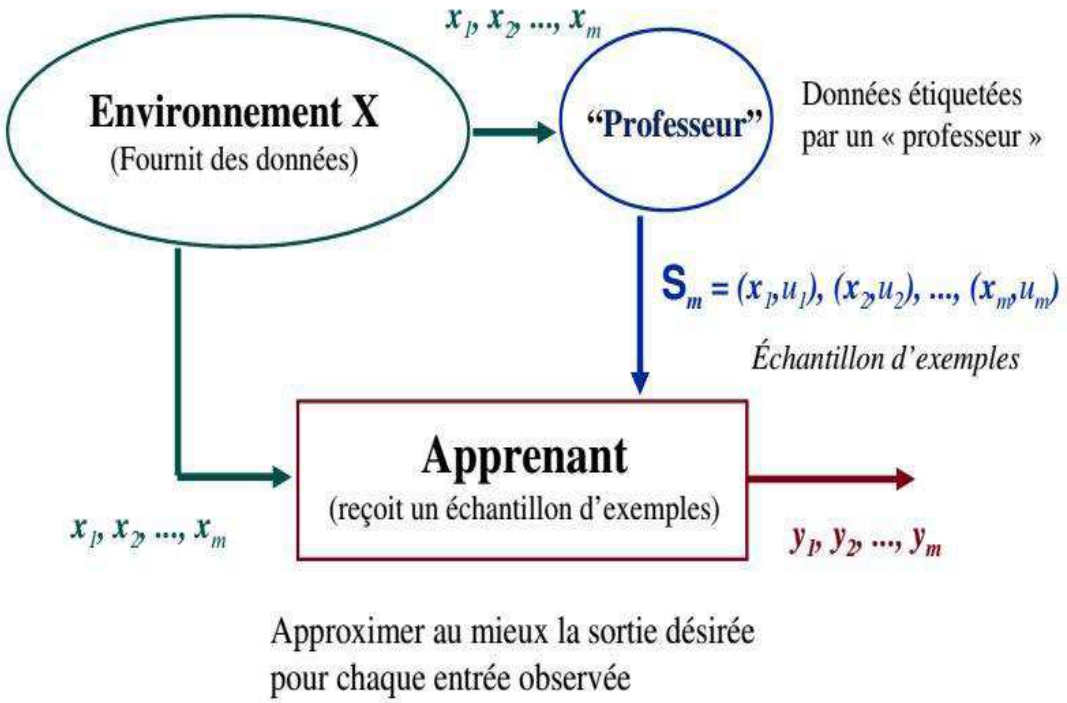

Figure 2.1 *Schéma de l'apprentissage supervisé .*

Cette forme d'apprentissage est utilisée dans de nombreuses méthodes. En voici quelques exemples : k-PPV, arbres de décision, réseaux de neurones…etc.

2.5. La catégorisation est un problème de classification supervisée

Pour construire un filtre relatif à une classe donnée, il faut donc disposer de couples (Document, catégorie), ces exemples de chaque catégorie, préalablement étiquetés constituent le corpus d'apprentissage.

On fait appel aux méthodes d'apprentissage supervisées pour ajuster un modèle qui crée une association entre les documents d'entrée et les catégories de sortie. Ainsi, par ces méthodes d'apprentissage, il est possible de construire un modèle de classification, à partir de ces exemples connus à priori (Document, catégorie). Ce qui

affirme clairement que la catégorisation de textes est bien un problème de classification supervisée.

2.6. Algorithmes d'apprentissage (méthodes d'apprentissage)

Le domaine de l'apprentissage automatique s'intéresse aux méthodes inductives permettant d'acquérir des connaissances à partir d'observations d'un phénomène. Cette connaissance peut être exploitée pour des tâches de décision ou de prévision : c'est le cadre de l'apprentissage supervisé ; ou à des fins d'analyse exploratoire ou de structuration d'un ensemble de données : c'est le cadre de l'apprentissage non-supervisé. Nous présente dans la suite quelque méthode d'apprentissage

2.6.1. 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).

Les k-PPV ont été utilisés en CT pour la première fois par B. Masand. Y. Yang a montré que l'algorithme est parmi les meilleurs actuellement [27]. Une des caractéristiques fondamentales de ce type de classificateur est l'utilisation d'une mesure de similarité entre les documents.

2.6.2. Naive bayes

ce classificateur se base sur le théorème de Bayes permettant de calculer les probabilités conditionnelles. Son utilisation lorsqu'il est appliqué à la classification de textes est résumé comme suit: on cherche la classification qui maximise la probabilité d'observer les mots du document. Lors de la phase d'entraînement, le classificateur calcule les probabilités qu'un nouveau document appartienne à telle catégorie à partir de la proportion des documents d'entraînement appartenant à cette catégorie. Il calcule aussi la probabilité qu'un mot donné soit présent dans un texte, sachant que ce texte appartient à telle catégorie. Quand un nouveau document doit être classé, on calcule les probabilités qu'il appartienne à chacune des catégories à l'aide de la règle de Bayes.[4]

La probabilité à estimer est donc : $P\ (c_j \mid a_1 a_2,\ a_3,\ \ldots,\ a_n)$ où : c_j est une catégorie et a_i est un descripteur. A l'aide du théorème de Bayes, on obtient :

$$P(c_j/a_i) = \frac{P\ (a_1,a_2,a_3,\ldots,a_n/c_j)\ P\ (c_j)}{P\ (a_1,a_2,a_3,\ldots,a_n)} \qquad (2.1)$$

Ce classificateur a comme avantage: possibilité en ligne et comme inconvénient: lorsque le modèle est mal spécifié, on aura intérêt à utiliser une méthode discriminative. [28]

2.6.3. Les arbres de décision

Les arbres de décision sont les plus populaires des méthodes d'apprentissage. Ils sont également populaires pour la classification de documents. Du point de vue de la classification automatique de documents, les arbres de décisions sont un algorithme qui permet de répartir les documents récursivement en groupes homogènes selon des attributs discriminants afin de pouvoir les classer dans les catégories qui leur correspondent. Le résultat est un enchaînement hiérarchique de règles où chaque nœud interne de l'arbre indique un attribut discriminant et les branches liant un nœud à ses fils représentent les valeurs discriminantes de l'attribut du nœud. Enfin, les feuilles de l'arbre représentent les catégories dans lesquelles les documents doivent être classés. En partant de la racine jusqu'une feuille de l'arbre on constitue une règle d'affectation de genre « *SI condition(s) ALORS décision*». L'ensemble de ces règles constitue le modèle de prédiction [29]

2.6.4. 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 [14] .Un profil prototypique [c] est calculé pour chaque classe selon :

$$c_w = \frac{t}{N_c} \sum_{d \in c} d_w - \frac{1-t}{N_{\bar{c}}} \sum_{d \notin c} d_w \qquad (2.2)$$

Où N_c est le nombre de documents dans c, $N_{\bar{c}}$ est le nombre de documents n'appartenant pas à c, et t est un paramètre du modèle compris entre 0 et 1.

Dans les situations où un document peut être attribué à une seule classe, t est souvent positionné à 1. Ces profils correspondent au barycentre des exemples (avec un coefficient positif pour les exemples de la classe et négatif pour les autres). Ces vecteurs sont également normalisés de la même façon que les documents.

Le classement des nouveaux documents s'opère en calculant la distance euclidienne (équivalente au produit scalaire et à la similarité en cosinus puisque tous les vecteurs sont de norme 1) entre la représentation vectorielle du document et celle de chacune des classes ; le document est assigné à la classe la plus proche [30] .

2.6.5. Réseaux de neurones

Les réseaux de neurones sont généralement optimisés par des méthodes d'apprentissage de type statistique grâce à leur capacité de classification et de généralisation, tels que la classification automatique de codes postaux ou la prise de décision concernant un achat boursier.

Un réseau de neurone est en général composé d'une succession de couches dont chacune prend ses entrées sur les sorties de la précédente. Chaque couche i est composée de N_i neurones, prenant leurs entrées sur les N_{i-1} neurones de la couche précédente. À chaque synapse est associé un poids synaptique, de sorte que les N_{i-1} sont multipliés par ce poids, puis additionnés par les neurones de niveau i, ce qui est équivalent à multiplier le vecteur d'entrée par une matrice de transformation. Mettre l'une derrière l'autre, les différentes couches d'un réseau de neurones revient à mettre en cascade plusieurs matrices de transformation et pourrait se ramener à une seule matrice produit des autres, s'il n'y avait à chaque couche, la fonction de sortie qui introduit un non linéarité à chaque étape.

Ceci montre l'importance du choix judicieux d'une bonne fonction de sortie : un réseau de neurones dont les sorties seraient linéaires n'aurait aucun intérêt.[31]

2.6.6. Machines à support de vecteurs (ou SVM)

Cette technique initiée par Vapnik tente de séparer linéairement les exemples positifs des exemples négatifs dans l'ensemble des exemples. Chaque exemple doit être représenté par un vecteur de dimension n. La méthode cherche alors l'hyperplan qui sépare les exemples positifs des exemples négatifs, en garantissant que la marge entre le plus proche des positifs et des négatifs soit maximale. Intuitivement, cela garantit un bon

niveau de généralisation car de nouveaux exemples pourront ne pas être trop similaires à ceux utilisés pour trouver l'hyperplan mais être tout de même situés franchement d'un côté ou l'autre de la frontière. L'efficacité des SVM est supérieure à celle de toutes les autres méthodes sur la classification de textes (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). Son efficacité est aussi très bonne pour la reconnaissance de formes. Un autre intérêt est la sélection de Vecteurs Supports qui représentent les vecteurs discriminant grâce auxquels est déterminé l'hyperplan. Les exemples utilisés lors de la recherche de l'hyperplan ne sont alors plus utiles et seuls ces vecteurs supports sont utilisés pour classer un nouveau cas. Cela en fait une méthode très rapide.

2.6.7. Algorithme génétique

C'est une méthode générale qui peut être utilisée après n'importe qu'elle méthode parmi celles citées précédemment, par exemple avec les arbres de décisions. En entrée, un algorithme génétique reçoit une population de classifieurs non optimaux. Le but du programme génétique est de produire un classifieur plus optimal que chacun de ceux de la population d'origine. D'une façon simple, cela consiste à extraire les meilleures parties de chaque classifieur d'origine et de les mettre ensemble pour produire un nouveau classifieur. Cela suppose de pouvoir comparer l'efficacité d'un classifieur. Un résultat important de la méthode est qu'après chaque itération on obtient un classifieur meilleur qu'avant. On peut donc arrêter les itérations à tout moment, même si le résultat n'est pas l'optimum.

2.6.8. Adaboost

Ce méthode Connue aussi sous le nom adabter boosting **,** Adaboost est Une méthode envisagée pour classifier automatiquement des textes est d'interroger différents classificateurs et de combiner leurs décisions de classification en pondérant chaque classificateur selon sa performance testée sur des exemples de validation semblables aux documents en question.

Il est clair qu'une telle approche est très coûteuse sur le plan informatique étant donné qu'il faut entraîner et faire fonctionner les différents classificateurs puis combiner

leurs résultats. Un algorithme AdaBoost permet d'obtenir des résultats intéressants. Son fonctionnement général revient à entraîner un comité de classificateurs en passant par plusieurs itérations, en donnant à chaque étape plus de poids aux exemples incorrectement classifiés à l'itération précédente et en retenant pour les étapes suivantes les classificateurs les plus prometteurs. [4]

Dans notre travail, nous avons utilisé Adaboost comme un algorithme d'un apprentissage pour catégoriser le texte arabe, nous présente en détaille Adaboost et sa fonctionnalités dans la prochain section.

Le tableau suivant résume les méthodes d'apprentissage les plus connus, introduit par F.SEBASTIANI dans [18].

Méthode d'apprentissage	*Type*	*Références*
Word	Non cité	[Yang, 1999]
Drop Bayes	Probabiliste	[Dumais & al, 1998]
Bim		[Li & Yamanishi, 1999]
Nb		[Yang & Liu, 1999]
C4.5	Arbres de décision	[Dumais & al, 1998]
Ind		[Lewis & Ringuette, 1994]
Swap-1	Règles de décision	[Apté & al, 1994]
Ripper		[Cohen & Singer, 1999]
Sleeping Experts		[Cohen & Singer, 1999]
DL-Esc		[Li & Yamanishi, 1999]
Charade		[Moulinier & al, 1996]
LLSF	Régression	[Yang & Liu, 1999]
Balanced Winnow	On line linear	[Dagan & al, 1997]
Widrow-HoFF		[Lam & HO, 1998]
Rocchio	Batch linear	[Cohen & Singer, 1999]
FindSim		[Dumais & al, 1998]
CLASSI	Réseau de neurone	[Ng & al, 1997]
Nnet		[Yang & Liu, 1999]
Gis-W	Example based	[Lam & Ho, 1998]
K-NN		[Yang & Liu, 1999]
SVMLight	SVM	[Yang & Liu, 1999]
ADABoost	Comité	[Schapire & Singer, 2000]
	Bayé-sien naif	[Dumais & al, 1998]

Table 2.1 *Méthodes d'apprentissage de catégorisation de textes[18]*

2.7. Critères d'évaluation des classificateurs

Pour évaluer tout processus de catégorisation, il est nécessaire d'appliquer une méthode d'évaluation. La performance de la catégorisation de textes est souvent mesurée via la précision et le rappel. La précision est défini comme la probabilité conditionnelle et le rappel mesure la largeur de l'apprentissage et correspond à la fraction des documents pertinents, parmi ceux proposés par le classificateur. [1]

Pour mieux illustrer les différentes mesures utilisées, on prend pour point de départ la table de contingence illustrée par le tableau 2.2, distincte pour chaque classe.

	Documents appartenant à la catégorie	Documents n'appartenant pas à la catégorie
Documents assignés à la catégorie par le classificateur	a	b
Documents rejetés de la catégorie par le classificateur	c	d

Table 2.2 Table de contingence à la base de l'évaluation des classificateurs.

On définit à partir des statistiques de cette table les mesures suivantes :

- **précision** («*precision*») :

$$P = a / (a + b) \qquad (2.3)$$

(Soit le nombre d'assignations correctes sur le nombre total d'assignations)

- **rappel** («*recall*») :

$$R = a / (a + c) \qquad (2.4)$$

(Soit le nombre d'assignations correctes sur le nombre d'assignations qui auraient dû être faites)

Aussi, il existe autres mesures comme :

- **exactitude** (*«accuracy»*) :

$$accuracy = (a + d) / (a + b + c + d) \qquad (2.5)$$

- **erreur** (*«error»*) :

$$error = (b + c) / (a + b + c + d) \qquad (2.6)$$

Ces deux mesures, bien que couramment utilisées en apprentissage automatique, sont jugées moins bien adaptées à la tâche de classification de textes. Elles incluent dans leur définition le nombre total de documents. Or, comme un document appartient généralement à un petit nombre de catégories sur l'ensemble, un classificateur qui rejetterait tous les documents présenterait seulement un faible taux d'erreur et une exactitude quand même très élevée. Entraîner un classificateur sur la base de l'optimisation d'un de ces deux critères tendrait à créer un programme qui n'accepte aucun document dans sa catégorie. C'est la raison pour laquelle la précision et le rappel sont les mesures les plus rencontrées dans la littérature.

Aussi, la mesure F1 est beaucoup utilisée. Elle est définie ainsi :

$$\boldsymbol{F - mesure(\beta)} = \frac{\boldsymbol{\beta^2} \times \text{Precision} \times \text{Rappel}}{\boldsymbol{\beta^2} \times \text{Precision} + \text{Rappel}} \qquad (2.7)$$

où

- *R* est le rappel
- *P* est la précision

C'est une fonction qui est maximisée quand la précision et le rappel sont proches. On cherche généralement à l'optimiser lors de l'ajustement du seuil.

Lorsque β = 1, la F-mesure est la moyenne harmonique entre la précision et le rappel (précision et rappel sont pondérés de façon égale).[32]

2.8. Conclusion

L'apprentissage est une étape important dans les applications de classification (catégorisation) automatique de texte .pour ce la Il existe une quantité importante de méthodes pour la classification de documents. Toutes sont issues des recherches sur l'apprentissage («Machine Learning »).

Dans ce chapitre nous présente l'apprentissage automatique de texte qui est existe deux types d'apprentissage supervisé et non supervisé, pour la classification non supervisée, l'étape de représentation des documents est essentielle. On a vu que la plupart des méthodes nécessitent de représenter chaque document sous la forme d'un vecteur (type attribut /valeur). Aussi, appliquées à la classification de documents, les méthodes peuvent se révéler très lentes puisqu'il est courant de traiter plusieurs centaines (voire milliers) de mots pour chaque document.

Contrairement à la classification non supervisée, la classification supervisée peut mesurer l'importance de chaque mot pour classer de nouveaux documents. Par exemple, une mesure («information gain ») calcule la typicité d'un terme. Plus un mot est lié à une catégorie et pas aux autres, et plus il est important : si un nouveau document le contient, ce mot sera très discriminant. De nombreuses mesures semblables ont été mises au point.

Enfin, à l'inverse de la classification non supervisée, il est ici simple d'évaluer les résultats d'une classification. Parmi les N exemples de documents classés, on utilise une partie des documents pour l'entraînement, et le reste pour le test. Pendant la phase de test, on soumet chaque document à l'algorithme de classification et on regarde simplement si la machine trouve la bonne classe. Bien sûr, le résultat de ce test n'est en rien garanti lorsque la machine aura à classer de nouveaux documents !

CHAPITRE 3
ALGORITHME DE BOOSTING
« ADABOOST »

3.1. Introduction

L'idée du boosting vient d'une question posée en 1988 par Kearns[33] : est-il possible de créer un classifieur performant en combinant plusieurs classifieurs faibles? On appelle classifieur faible tout classifieur capable de performances étant, au pire, équivalentes a une prédiction aléatoire. Le boosting est donc un méta-algorithme qui permet d'améliorer les performances de classifieurs faibles en les combinant. La réponse fut obtenue au début des années 1990 par Schapire [34] qui a introduit la première version de boosting. Cette version a été améliorée en suite par Freund [35] un an plus tard en se basant sur le paradigme suivant : « tout algorithme capable, pour toute distribution, d'apprendre avec une confiance faible et une précision inférieur à ½ peut être transformé en un algorithme d'apprentissage avec une confiance aussi grande et une précision aussi bonne que désirée »[1]

l'algorithme d'AdaBoost (Adaptive Boosting) est la dérivée la plus pratiquée de la méthode du Boosting qui vise à stimuler la performance de l'algorithme d'apprentissage. Adaboost consiste à transformer, d'une manière efficace, un classifieur « faible » en un classifieur « fort » en réduisant les taux d'erreur. dans la suite de cette section, nous avons expliqué en détail l'algorithme d'Adaboost et son principe, son utilisation dans le domaine de l'apprentissage automatique et la catégorisation du texte.

3.2. Le boosting d'un algorithme d'apprentissage

Le "Boosting" définit un principe général d'apprentissage permettant d'améliorer la précision d'un algorithme donné d'apprentissage. Le principe général est de combiner linéairement des résultats de classificateurs dits "faibles", c'est-à-dire dont la seule garantie est qu'elles soient un peu meilleures que le hasard, pour construire un classificateur "fort" d'apprentissage à partir de l'ensemble original et une méthode de combinaison de classificateurs construits à partir de chaque nouvel ensemble.

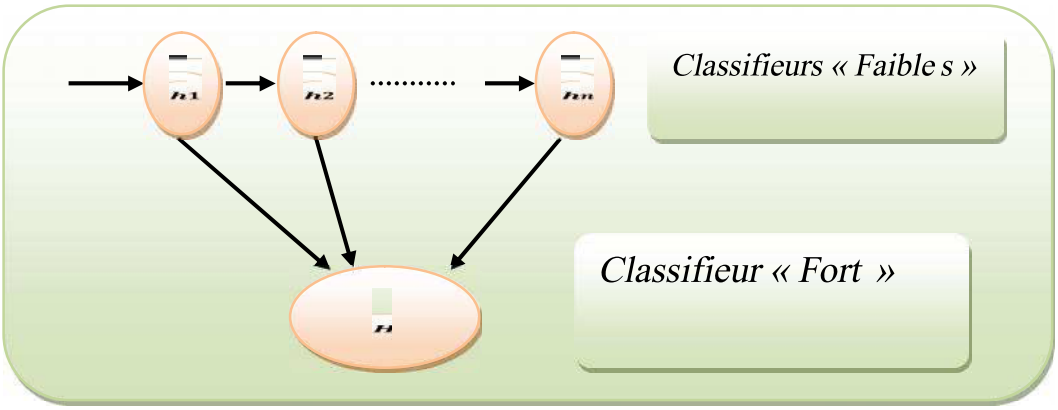

Figure 3.1 Principe de combinaison des classifieurs « faibles ».

Pour définir sa nouvelle technique de "Boosting" , Shapire se base sur l'idée que tout classificateur faible capable d'apprendre avec une certaine confiance et une erreur de classification inferieure à (0,5), peut être transformé en un classificateur plus confiant et avec une erreur de classification aussi petite que désirée. En d'autres termes, un classificateur faible donnant de meilleurs résultats qu'une simple pile ou face (50% de risque) peut être la base pour construire un ensemble de classificateur s.

A chaque itération, l'algorithme cherche à trouver un classificateur faible qui peut corriger au mieux les erreurs des classificateurs obtenus aux itérations précédentes. Dans le principe de "Boosting", cet objectif est réalisé à l'aide d'une pondération des données d'apprentissage.

3.3. L'algorithme d "AdaBoost"

Comme déjà cité précédemment, L'algorithme "AdaBoost" (Adaptive Boosting), développé par Freund et Schapire [37], est un algorithme qui crée impérativement une combinaison linéaire de classificateur de la forme :

$$H(x) = Signe(\sum_{t=1}^{T} \alpha_t \, h_t(x)) \qquad (3.1)$$

Ou :

h_t représente un classificateur faible et α_t son poids.

"AdaBoost"

Entrées: Une base d'apprentissage $A = \{x_1, x^2, \ldots, x_M\}$

Les étiquettes $y = \{y_1, y_2, \ldots, y_i\}$ ou $y_i \in \{-1, 1\}$

Algorithme de classification C, Nombre d'itérations T

Sorties: Classificateur fort H

/* Initialiser le vecteur de poids des exemples avec $i = 1, 2, \ldots, M$ */ $D_0 = \frac{1}{M}$

Pour $t = 1$ à T **Faire**

/* Trouver un classificateur faible h_t qui minimise l'erreur selon D_t */

h_t = Entrainer Classificateur Faible(C, A, D_t)

ε_t= Erreur (h_t , **A**) /* Taux d'erreur de h_t*/

$\alpha_t = \frac{1}{2} ln(\frac{1-\varepsilon_t}{\varepsilon_t})$ /* Poids du h_t*/

/*Mettre à jour le vecteur de poids */

$$D_{t+1}(i) = \frac{Dt(i) exp(-\alpha_t \; y_i \; h_t(Xi))}{Z_t}$$

Fin Pour

Retourner $H(x) = Signe(\sum_{t=1}^{T} \alpha_t h_t(x))$

Algorithme 3.1 ***"AdaBoost".***

L'algorithme 1.1 montre le pseudo code d'"AdaBoost" .A chaque étapes t de l'algorithme, l'apprenant cherche un bon classificateur $h \in \{-1,1\}$ pour la distribution de probabilité, a priori, sur les exemples d'apprentissage. Cette distribution est représentée grâce à un poids D_t^i pour chaque exemple i qui est mis à jour en fonction de la qualité de la prédiction obtenue à l'étape précédente. L'une des idées principales est de donner aux exemples difficiles à classer un poids D_t^i élève, (et inversement) de manière à amener l'algorithme à se concentrer sur ces derniers.

Chaque classificateur est affecté d'un coefficient proportionnel à l'erreur pondérée. Après T étapes, le classificateur final obtenu est $H(x)$.

3.4. Exemple d'Adaboost

Pour mieux comprendre comment fonctionne l' algorithme Adaboost, considérons l'exemple suivant[2] :

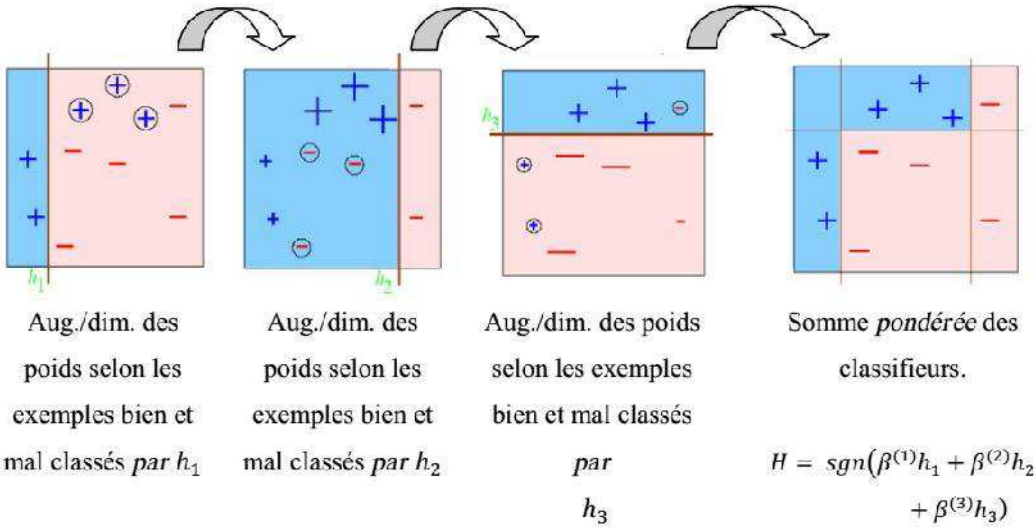

Figure 3.2 *Exemple d'"AdaBoost".*

Dans cette figure, on dispose d'un ensemble de 10 exemples (N=10 , dont 5 sont positifs et 5 sont négatifs repartis aléatoirement dans un espace donné). Le but est de construire en 3 itérations (T=3), un apprenti linéaire capable de séparer les exemples positifs des exemples négatifs pour pouvoir classer correctement un nouvel exemple , il va falloir estimer s'il est positif ou négatif. Pour achever cette tâche, et puisqu' il y a 10 exemples, le dopage commence par assigner le même poids

$$\boldsymbol{D_i} = \frac{\mathbf{1}}{\boldsymbol{M}} = \frac{\mathbf{1}}{\mathbf{10}} = 0.1$$

à tous les exemples i $(i = 1, 2, \dots, M)$ et puis procède par interroger des faibles apprentis en demandant à chacun d'estimer où se trouve le séparateur.

Itération 1 :

le premier apprenti $\boldsymbol{h_1}$ estime que le séparateur existe à gauche de l'espace (comme l'indique la première figure à gauche). Selon ce séparateur tous les exemples à gauche de la ligne sont considérés positifs et ceux qui sont à droite

sont négatifs. Ainsi, $\boldsymbol{h_1}$ a proprement classé deux exemples positifs mais a raté trois autres et donc d'après le pseudo de l'algorithme on obtient :

$$\varepsilon_1 = 0.3$$

$$\alpha 1 = 0.42$$

Dans ce cas là, Boosting diminue le poids de tous les exemples classés correctement par une valeur ≅ 0.07 selon la formule de l'algorithme et augmente les poids des trois autres d'une valeur ≅ 0.17. Il faut noter que l'augmentation et la diminution des poids dans la figure précédente sont indiquées graphiquement par l'agrandissement et la diminution de la taille des symboles + et -. Le fait d'augmenter la taille d'un exemple incite le prochain apprenti faible de se concentrer plus sur sa classification.

Ainsi $\boldsymbol{h_2}$ estime que le séparateur existe où l'indique la figure au milieu. Ce séparateur a proprement classé tous les exemples positifs ainsi que les deux exemples négatifs à droite mais a raté les trois autres et donc :

$$\varepsilon_2 = 0.21$$

$$\alpha 2 = 0.65$$

En conséquent, les poids des exemples correctement classés sont diminués et les poids des exemples mal-classés sont augmentés.

Le boosting continue avec $\boldsymbol{h_3}$ qui estime que cette fois ci le séparateur est horizontal et donc a raté 3 exemples (1 négatif et 2 positifs) et ainsi :

$$\varepsilon_3 = 0.14$$

$$\alpha 3 = 0.92$$

Et par conséquence le classifieur H final est :

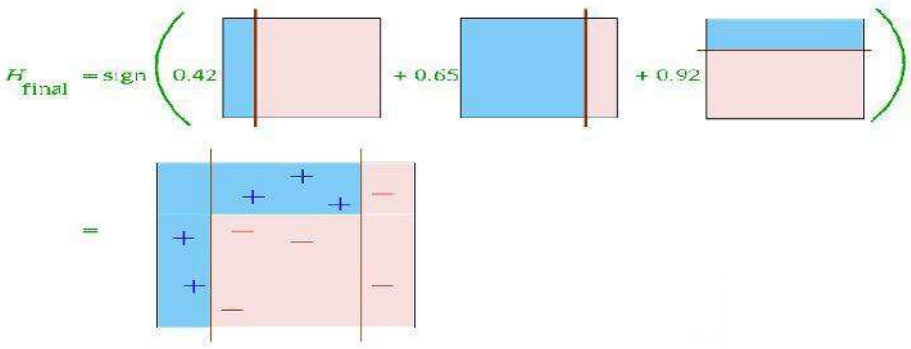

Figure 3.3 *Classifieur H final.*

Pour classer un nouvel exemple, le classifieur final H interroge chacun des trois classifieurs en le pondérant avec le poids qui lui est associé et additionne les trois valeurs obtenus. Si le résultat de cette somme est une

valeur positive c'est que le nouvel exemple est positif et si cette valeur est négative c'est qu'il est négatif. Il est très important de noter que le boosting est une méthode générale qui vise à améliorer la précision et la fidélité de n'importe quel algorithme d'apprentissage(comme les arbres de décision, naïve bayes, les réseaux de neurones) [36].

3.5. Les propriétés d'AdaBoost

3.5.1. L'erreur empirique d'AdaBoost

L'erreur empirique d'AdaBoost est calculée en fonction de :

l'erreur ε_t de h_t :

$$\varepsilon_t = \frac{1}{2} - \gamma_t \qquad (3.2)$$

Ou γ_t mesure l'amélioration apportée par hypothèse h_t par rapport à l'erreur de base $1/2$.

[37], ont montré que l'erreur empirique (l'erreur sur l'´echantillon d'apprentissage S) de l'hypothèse finale H est bornée par :

$$\prod_{t=1}^{T}\left[2\sqrt{\varepsilon_t(1-\varepsilon_t)}\right] = \prod_{t=1}^{T}\sqrt{1-4\gamma_t^2} \leq exp\left(-2\sum_{t}\gamma_t^2\right) \qquad (3.3)$$

Ainsi, si chaque hypothèse faible est légerment meilleure que le hasard, ($\gamma_t \geq \gamma > 0$), alors l'erreur empirique diminue exponentiellement avec t.

3.5.2. L'erreur en généralisation d'AdaBoost

L'erreur en généralisation de l'hypothèse finale H peut être bornée par une expression faisant intervenir l'erreur en apprentissage, le nombre d'exemple d'apprentissage m, la dimension de Vapnik-Chervonenkis d_H de l'espace d'hypothèse et T le nombre d'´ittérations de boosting :

$$R_{Réel}(H) = \varepsilon_T + \mathcal{O}\left(\sqrt{\frac{T \cdot d_{\mathcal{H}})}{m}}\right) \qquad (3.4)$$

Ou $R_{Emp}(H)$ dénote l'erreur empirique mesurée sur l'´echantillon d'apprentissage.

3.5.3. Adaboost et les marges

La marge d'un exemple (x, y) , avec $y = \pm 1$ dans l' AdaBoost désignant la classe, s'exprime par :

$$\boldsymbol{marge(x, y) = \frac{y \sum_{t=1}^{T} \alpha_t \, h_t(x)}{\sum_{t=1}^{T} \alpha_t}} \qquad (3.5)$$

ce nombre est compris dans l'intervalle $[-1, +1]$ et est positif seulement si H classifie correctement l'exemple. La marge peut être interprétée comme une mesure de confiance dans la prédiction. Il a été prouve que l'erreur en généralisation peut être bornée, avec une forte probabilité, pour m assez grand et pour tout $\theta > 0$ par :

$$R_{Réel}(H) \leq \hat{P}r[\text{marge}(x,y) \leq \theta] + \mathcal{O}\left(\sqrt{\frac{d_{\mathcal{H}}}{m\theta^2}}\right) \qquad (3.6)$$

Cette borne est indépendante de T , le nombre d'itérations de boosting.

3.6. Multi-classe et d'Adaboost

3.6.1. Multi-classe d'adaboost

Il existe plusieurs généralisations de AdaBoost au cas multiclasse, par exemple dans [38]. Le principe de multi-classe est :

- Soit Une base d'apprentissage $A = \{x_1, x_2, \ldots, x_M\}$
- On suppose que $y = \{1, \ldots, k\}$,k classe

et le classsifieur associé est :

$$\boldsymbol{H(F(x)) = Arg_k Max(F_k(x))} \qquad (3.7)$$

Où :

F_k représente un classificateur final pour chaque classe.

"AdaBoost Multi-classe"

Entrées: Une base d'apprentissage $A = \{x_1, x_2, \ldots, x_M\}$

Les étiquettes $y = \{y_1, y_2, \ldots, y_i\}$ ou $y_i \in \{1, 2, \ldots, k\}$

Algorithme de classification C, Nombre d'itérations T

Sorties: Classificateur fort H

/* Initialiser le vecteur de poids des exemples avec $i = 1, 2, \ldots, M$ */ $D_0 = \frac{1}{M}$

Pour $t = 1$ à T **Faire**

/* Trouver un classificateur faible h_t qui minimise l'erreur selon D_t */

h_t = Entrainer Classificateur Faible(C, A, D_t)

/* Taux d'erreur de h_t */

ε_t= Erreur $(h_t, A) = \sum_{i:h(x_i)\neq y_i} D_t(i)$

/* Poids du h_t */

$$\alpha_t = ln\left((k-1)\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

/*Mettre à jour le vecteur de poids */

Pour $i = 1$ à n **Faire**

Si $h(x_i) \neq y_i$

$$\widetilde{D}_{t+1}(i) = D_t(i)(K-1)(k-1)\frac{1-\varepsilon_t}{\varepsilon_t}$$

Sinon

$$\widetilde{D}_{t+1}(i) = D_t(i)$$

Fin Pour

$$D_{t+1}(i) = \frac{\widetilde{D}_{t+1}(i)}{Z_t}$$

Fin Pour

/*calcul de F_t pour chaque classe */

Pour tout : $y \in Y$

$$F_t(x, y) = \sum_{t=1}^{T} \alpha_t h_t(x))$$

Retourner

$$H(x) = Arg_{y\in Y} Max(F_T(x, y))$$

Algorithme 3.2 ***"AdaBoost Multi-classe".***

Où :

Z constante de normalisation :

$$Z_t = \sum_i \widetilde{D}_{t+1}(i) \qquad (3.8)$$

Dans l'algorithme d'Adaboost de multi-classe, mêmes étapes s'implique comme l'adaboost (3.3) mais la diffrence est pour chaque itération, Les poids sont calculés en termes de classe k. aussi On calcule le vote majore pour chaque classe F_k qui permet de calculer le vote final H_{final} .

3.6.2. Le boosting et les problèmes multi-classes

L'utilisation d'une méthode de boosting telle que AdaBoost est délicate dans le cas d'un problème multi-classe. En effet, les résultats théoriques sur l'erreur en apprentissage sont conditionnés par les performances de la méthode d'apprentissage utilisée dans le cas binaire : les hypothèses produites doivent avoir un taux d'erreur en apprentissage inferieur à $1/2$, c'est 'a dire un peu meilleur que l'aléatoire. Il n'en va pas de même lorsque le problème comporte $k > 2$ classes. Il peut d'avérer alors plus difficile d'obtenir un tel taux de succès, dans la mesure où une méthode légerment plus performante qu'un étiquetage aléatoire ne garantira qu'une performance supérieure à $1/k$. S'il est toutefois possible d'obtenir des hypothèses suffisamment performantes pour garantir la convergence du taux d'erreur, la construction d'hypothèses simples est avantageuse en termes de cout computationnel.

3.7. Variantes d'AdaBoost

"AdaBoost" a montré une bonne performance ainsi qu'une certaine robustesse au problème de sur-apprentissage sauf s'il y a présence de données bruitées [39]. Son inconvénient est qu'il augmente exponentiellement le poids des exemples bruités et de trouver des hypothèses faibles qui corrigent les erreurs et qui se concentrent sur ces exemples bruités. La conséquence est que la combinaison des hypothèses faibles tend à sur-apprendre le bruit. Pour éviter ce problème, les techniques proposées apportent des modifications au niveau de la mise à jour des poids des exemples ainsi que sur les poids des classificateurs faibles.

Dans la suite, nous présentons quelque variantes d'"AdaBoost" proposées dans ce cadre.

"MAdaBoost"[40] propose une modification simple sur le calcul de poids des exemples. Cette modification a pour but de limiter la croissance incontrôlée des poids. Les auteurs proposent de borner le poids de chaque exemple par son poids initial, ce qui lui évitera de devenir arbitrairement trop grand. L'inconvénient de cette méthode est qu'elle propose une modification qui réduit l'effet positif initial de la croissance exponentielle (rapidite de la convergence).

"IAdaBoost" [41] propose une modification de la mise à jour des poids des exemples dans "AdaBoost" en prenant en compte non seulement le problème de sur-apprentissage, mais aussi la vitesse de convergence. Cette mise a jour est calculée en exploitant une mesure d'entropie locale adaptative, issue d'un graphe de voisinage construit sur les exemples.

Modest"AdaBoost" [42] Sa différence principale par rapport aux autres méthodes de "boosting" est la façon de calculer les poids des hypothèses faibles (classificateurs faibles). A chaque construction d'une nouvelle hypothèse faible le poids est calculé en prenant en compte la corrélation entre cette nouvelle hypothèse et le classificateur fort construit avant. Plus la corrélation n'est pas élevée, plus cette hypothèse sera pénalisée en diminuant son poids.

3.8. Adaboost et la catégorisation du texte

L'utilisation de l'algorithme d'apprentissage Adaboost Pour la catégorisation de texte est simple, par exemple la vérification de la présence ou de l'absence d'un terme dans le texte donné. Tous les mots et les paires de mots adjacents sont des termes potentiels. Basé uniquement sur les résultats de ce test, l'hypothèse faible émet des prédictions et des confidences que chaque étiquette est associée avec le texte.

Par exemple, une catégorie possible peut être politique, et le prédicateur correspondant est: Si le terme " la loi " apparaît dans le document puis de prédire que le document appartient à politique avec une grande certitude, au " ministère des Finances " avec peu de confiance, et qu'il n'appartient pas à des sports avec une grande certitude. Si, d'autre part, le terme n'apparaît pas dans le document, puis prédire qu'il n'appartient pas à l'une des classes ayant un faible confiance.

Les étapes de l'algorithme « Adaboost » qui s'implique pour la catégorisation du texte comme suivant :

Soit x représentent le domaine des documents de texte possibles pour la catégorisation et soit $y = \{y_1, y_2, \ldots, y_s\}$ un ensemble fini d'étiquettes ou classes.

Soit $D = \{Doc_1, Doc_2, \ldots, Doc_m\}$ désignent l'ensemble de la formation des documents de texte, ($x \subset D$).

La formation des documents de texte peut être une représentation de texte comme la représentions en mot, représentation de N-gramme comme le cas de notre travail, pondération de termes de texte...Etc. Aussi, le même pour les classes, chaque classe a un ensemble des attribués ou formations associées.

Les mêmes étapes d'adaboost pour la catégorisation du texte. On peut utiliser algorithme 3.1 dans la catégorisation binaire de textes, ou l'algorithme 3.2 pour la catégorisation multi calasse de textes.

3.9. Conclusion

Adaboost est une nouvelle méthode de Boosting principalement utilisée pour stimuler les performances des algorithmes d'apprentissage. Depuis son apparition jusqu'à aujourd'hui, il y a eu la naissance de plusieurs générations. Dans ce chapitre nous avons présenté le principe de bossting d'une manière générale et le premier algorithme de boosting « Adaboost », ainsi nous avons indiqué le principe et le fonctionnement de Adaboost dans le cas d'une catégorisation binaire, l'application de adaboost sur une catégorisation multi-classe, et enfin l'utilisation de adaboost dans la catégorisation de textes.

CHAPITRE 4

REALISATION

ET

EXPERIMENTATION

4.1. Introduction

Dans ce projet, nous nous intéressons à l'utilisation de l'apprentissage automatique pour catégoriser (ou classer) les textes arabes d'où la nécessité de les extraire d'abord avant de les traiter. Pour atteindre cet objectif, nous avons développé une application (analyseur, classifieur) à l'aide de l'environnement de programmation Delphi.

4.2. Technologies et outils de développement

Nos expérimentations ont été développées sur une machine possédant les caractéristiques suivantes : un processeur Intel® Core™ i3-3217U CPU,1.80 GHz .Et une mémoire de 4Go. L'ensemble est piloté par le système d'exploitation Windows 7.

4.2.1. Environnement de développement

Notre choix pour l'environnement de développement s'est porté sur le l'environnement Delphi, Delphi est un environnement de programmation visuel orienté objets permettant de développer des applications sous Windows. Il représente la suite logique de la famille turbo Pascal avec ses nombreuses versions (précisément le pascal objet). Delphi est un outil moderne, puissant, faisant appel à une conception visuelle des applications, à la programmation orientée objet, à une bibliothèque de composants très riche (la VCL: Visual Components Library), des fichiers DLL (Dynamic Link Library) et API (Application Programming Interface) de Windows. Delphi se classe comme l'un des meilleurs environnements de développement rapide des applications (RAD) dans le monde informatique.

Delphi, c'est aussi un produit qui évolue avec le temps. Chaque nouvelle version possède son lot de nouveautés.

4.3. Présentation du classifieur réalisé « *CTAAdaBoost* »

4.3.1. Corpus utilisé

Le corpus d'apprentissage utilisé lors de cette étude contient 170 documents de différentes tailles et contenus répartis sur 5 catégories comme le démontre le tableau ci-dessous.

Catégorie	Nombre de Documents
Politique	35
Economie	35
Sport	35
arts	35
santé	35

Table 4. 1 *La répartition des documents dans les catégories du corpus.*

Le corpus en entier contient 205 textes nous avons reparti le corpus d'apprentissage comme montre le Tableau 4. 1, ou nous avons utilisé 170 pour l'apprentissage et le reste pour le test.

	Textes d'apprentissage	Le reste (textes de test)
Corpus	170	35

Table 4. 2 *Les textes d'apprentissage et* textes de test *pour chaque catégorie d u corpus.*

4.3.2. Mots outils arabes (Arabic stopwords)

Comme d'autres langues, l'arabe contient des mots fonctionnels (ou mots outils) qui ne véhiculent pas un sens particulier dans le texte. Donc, il est nécessaire d'éliminer ces mots avant la phase d'apprentissage. Ces mot sont appelés également les mots vides « Stop Words ».

La liste des mots vides la plus répandue, et largement reprise par les travaux dans ce domaine, est celle de Khodja qui contient 168 entités, en 2008 kadri a ajouté à la liste de Khodja d'autres entités, ce qui a donné un nombre de 431 entités[33].

Le tableau ci-dessous présente quelques mots outils arabes.

Mots outils (stop words) arabe
الكلمة بيد سوى غير لاسيما بكم بما بماذا بمن كم كيف ما ماذا متى مما ممن من أنّى أيّ ّأيّان أين أينما حيثما كيفما ما متى من مهما أولئك أولئكم أولاء أولالك تان تانِك تلك تلكم تلكما تِه تِي تَيْن تينك ثمّ ثمّة ذا ذاك ذان ذانك ذلك ذلكم ذلكما ذلكن ذِه ذوا ذواتا ذواتي ذِي ذَيْن ذينك كذلك هَؤلاء هَاتان هَاتِه هَاتِي هَاتَيْن هاهنا هَذا هَذان هَذِه هَذِي هَذَيْن هكذا هنا....

Table 4. 3 *quelques mots outils arabes.*

La liste des mots vides que nous avons utilisés pour le traitement des textes arabes dans notre application sont donnés dans l'Annexe 1.

4.4. Présentation du processus de catégorisation dans « *CTA_AdaBoost*»

Le schéma suivant explique le processus de catégorisation éffectué par notre application CTA AdaBoost :

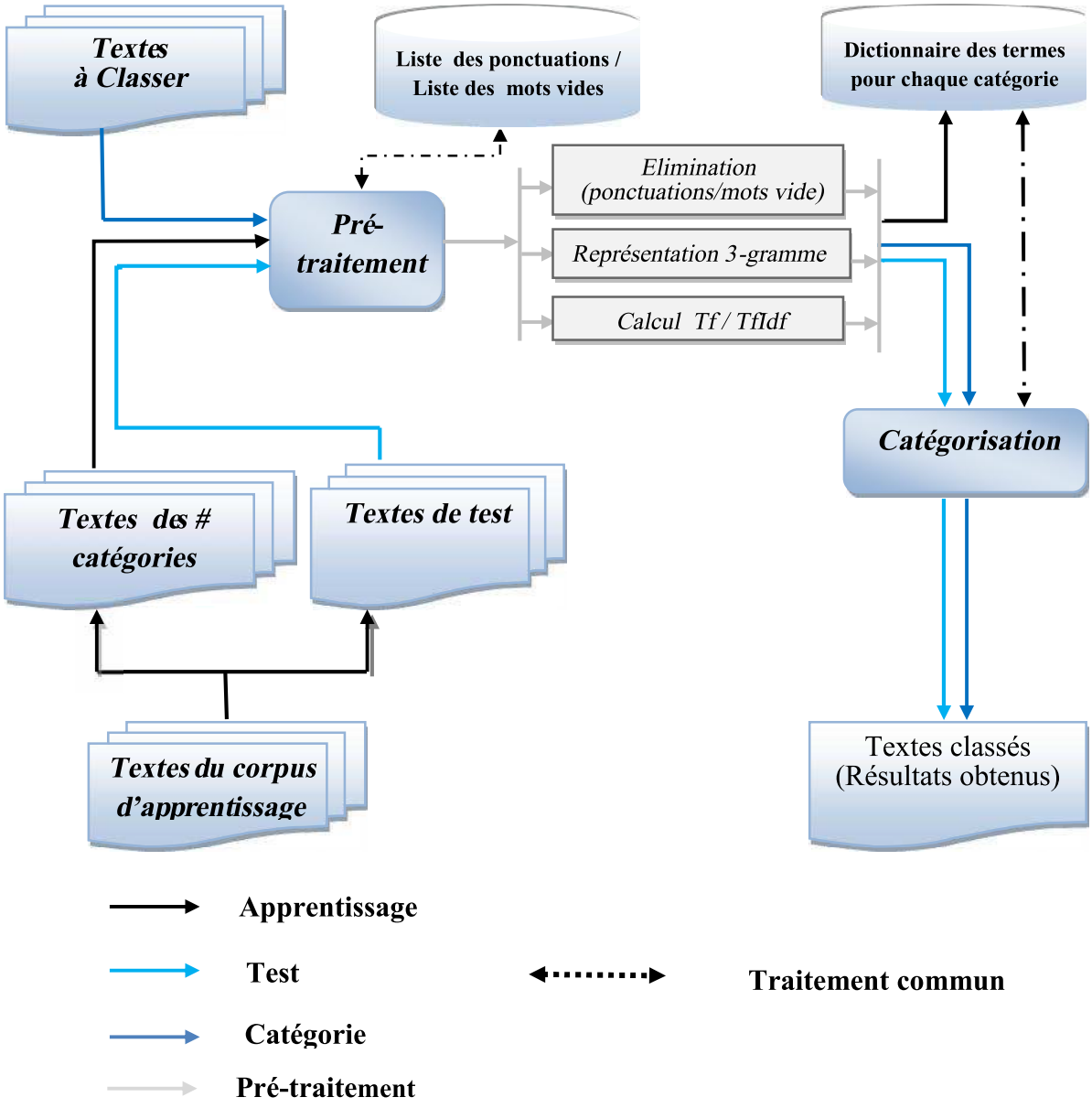

Figure 4.1 *Structure et fonctionnement de l'application « CTA_AdaBoost».*

Comme illustré dans la figure 1.4, dans la phase de prétraitement des textes du corpus d'apprentissage et de test, nous utilisons l'approche "3_grammes" comme méthode de représentation. Les documents seront représentés par des vecteurs de termes (3_grammes) sans utilisation de ressource sémantique. Une fois les documents représentés, ils seront pondérés afin de donner un poids pour chaque terme dans chaque document. La pondération utilisée est la pondération TF, plus un poids pour chaque terme dans le corpus de documents pour calculer à la fin une nouvelle pondération plus efficace qui est la pondération TFidf (Term Frequency x Inverse Document Frequency).

Par la suite, nous utilisons la méthode Adaboost pour catégoriser les nouveaux documents traités selon les étapes suivantes :

- Préparation des textes : effectuer des prétraitements sur les textes du corpus d'apprentissage et de test ou même ceux à catégoriser pour construire à la fin les vecteurs (3-grammes) associés à chaque texte.
- Pour procéder à la phase de catégorisation proprement dite en utilisant l'algorithme adaboost, on prend en entrée de l'algorithme : les vecteurs des termes (les 3-grammes) qui représentent les différentes catégories. Chaque terme représente une hypothèse (un calssifieur , une regèle de classification) pour l'algorithme.
- Utilisation d'un vecteur de poids initiaux D_0 dont chaque composante est associée à un terme (en entrée de l'algorithme). Notons que D_0 sert à calculer l'erreur de classification de chaque hypothèse (ex : soit le vecteur du texte $V_d = \{$اقت, مال, سلط, جزا, صاد, ... $\}$), les poids initiaux associés sont identiques et égale a 1/m (m : le nombre des 3-grammes du texte).
- Fixation d'un nombre T d'itérations pour l'algorithme, à chaque itération t nous cherchons un ensemble de classifieurs faibles parmi l'ensemble des hypothèses (les termes ou les 3-grammes) du vecteur du dictionnaire de la catégorie (ex : pour le vecteur de la catégorie « économie » $V_h = \{$اقت, مال , صاد, مؤس , ... $\}$). nous calculons l'erreur d'entraînement ε_t pour chaque hypothèse h_t de l'ensemble des hypothèses.
- A la fin de chaque itération t, nous recalculons le vecteur de poids D_t associé aux termes du texte en fonction des valeurs de pondération α_t associées aux hypothèses h_t avec $\alpha_t = \frac{1}{2} * \ln\left(\frac{1-\varepsilon_t}{\alpha\varepsilon_t}\right)$
- Apres T itérations, nous obtenons comme sortie un ensemble des regèles de classification. Chaque règle de classification (classifieur, hypothèse) est pondérée par une valeur α_t .
- La combinaison de ces regèles de classification h_t permet de trouver une hypothèse finale H $(H(x) = Signe(\sum_{t=1}^{T} \alpha_t\, h_t(x)))$ qui détermine si le texte $peut\ être\ classé$ dans la catégorie C_i (si la valeur de H(score de classe C) >= 0) ou non (si la valeur de H (score de classe C) < 0).
- dans le cas de la catégorisation multi-classes, et pour déterminer la catégorie du texte on prend la valeur maximale de H parmi les valeurs de H des différentes catégories.

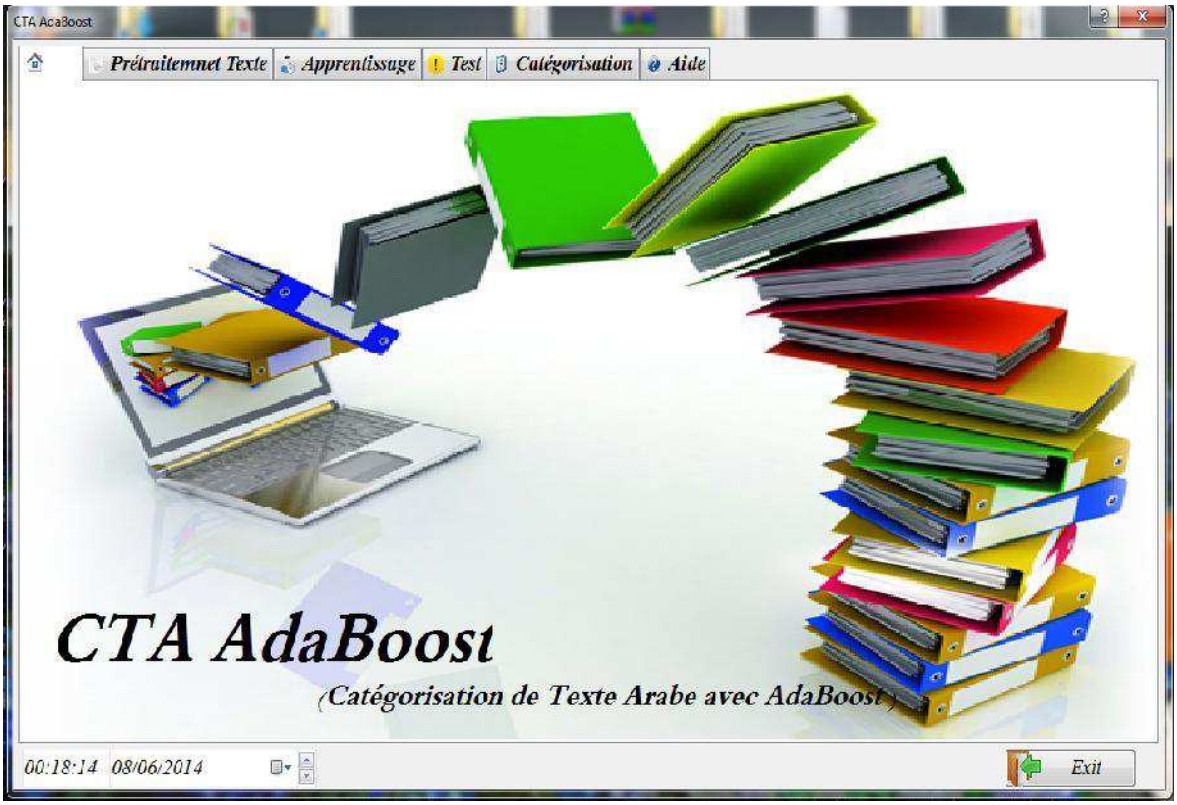

Figure 4.2 Fenêtre *« Interface du classifieur CTA_AdaBoost».*

4.4.1. Prétraitement du texte

Comme nous avons défini la représentation n-gramme dans le chapitre 1. Les textes sont transformés en vecteurs dont chaque composante représente un terme de 3 grammes.

Le prétraitement consiste à :

- Eliminer les ponctuations : des signes de ponctuations, des chiffres,….
- Eliminer les mots vides de la langue arabe.

Nous donnons dans la Figure 4.3, la représentation d'un texte en vecteur (3-grammes).

احتلت السعودية المركز الـ16 عالميا كأكبر قوة اقتصادية متفوقة بذلك على كثير من الدول الاقتصادية الكبرى، في حين احتلت قطر المرتبة الأولى عالمياً وعربياً وخليجياً من حيث نصيب كل فرد من الناتج المحلي، وبلغ 146.521 ألف دولار، بحسب تقرير برنامج المقارنات الدولية الصادر عن البنك الدولي.

ات	لي	يا	ى ع	ن ا	ل ا	ة م	ميا	احت
ت د	ي و	ا ن	عا	اح	اق	مت	يا	حتل
دو	وب	نص	عال	احت	اقت	متف	ا ك	تلت
دول	وبل	نصي	الم	حتل	قتص	تفو	كأ	لت
ولي	بلغ	صيب	لمي	تلت	تصا	فوق	كأك	ت س
لية	لغ	يب	ميا	لت	صاد	وقة	أكب	سع
ية	غ أ	ب ف	يا	ت ق	ادي	قة	كبر	سعو
ة ص	أل	فر	ا و	قط	دية	ة ب	بر	عود
صا	ألف	فرد	وع	قطر	ية	بذ	ر ق	ودي
صاد	لف	رد	وعر	طر	ة ك	بذل	قو	دية
ادر	ف د	د ن	عرب	ر م	كب	ذلك	قوة	ية
در	دو	نا	ربي	مر	كبر	لك	وة	ة م
ر ب	مج	نات	بيا	مرت	برى	ك ك	ة ا	مر
بن	ج م	اتج	يا	رتب	رى	كث	اق	مرك
بنك	نات	تج	ا و	تبة	ى ف	كثي	اقت	ركز
نك	مق	ج م	وخ	بة	في	ثير	قتص	كز
ك د	مقا	مح	وخل	ة أ	في	ير	تصا	ز ع
دو	قار	محل	خلي	أو	ي ح	ر د	صاد	عا
دول	ارن	حلي	ليج	أول	حي	دو	ادي	عال
ولي	رنا		يجي	ولى	حين	دول	دية	الم
			جيا	لى	ين	ول	ية	لمي

Figure 4.3 *Exemple d'un texte arabe et son vecteur associé(3_grm).*

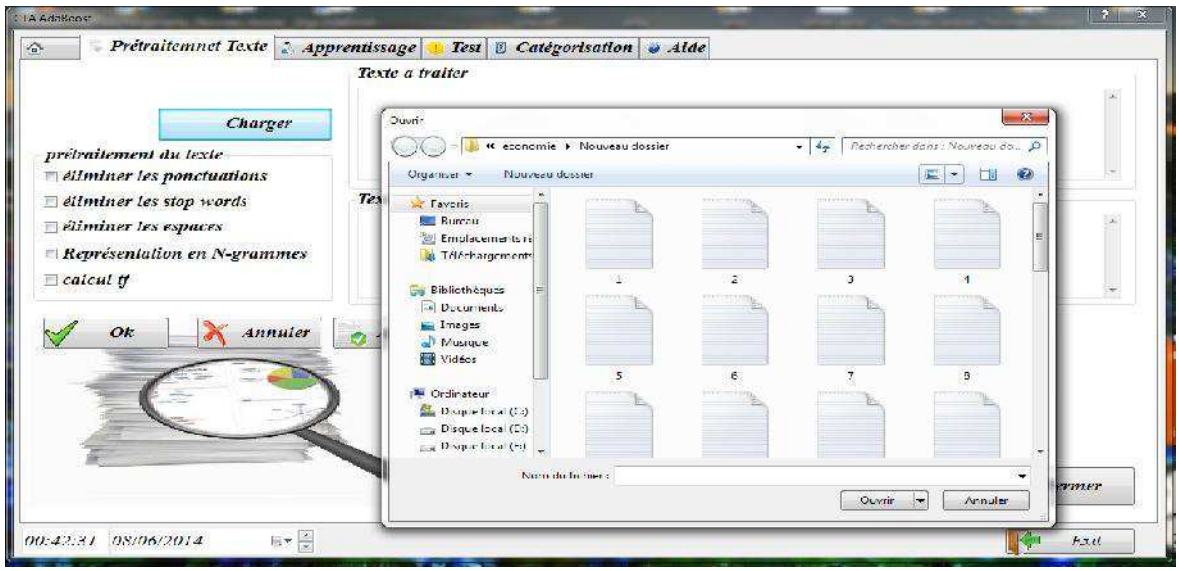

Figure 4.4 *Fenêtre « Chargement d'un texte ».*

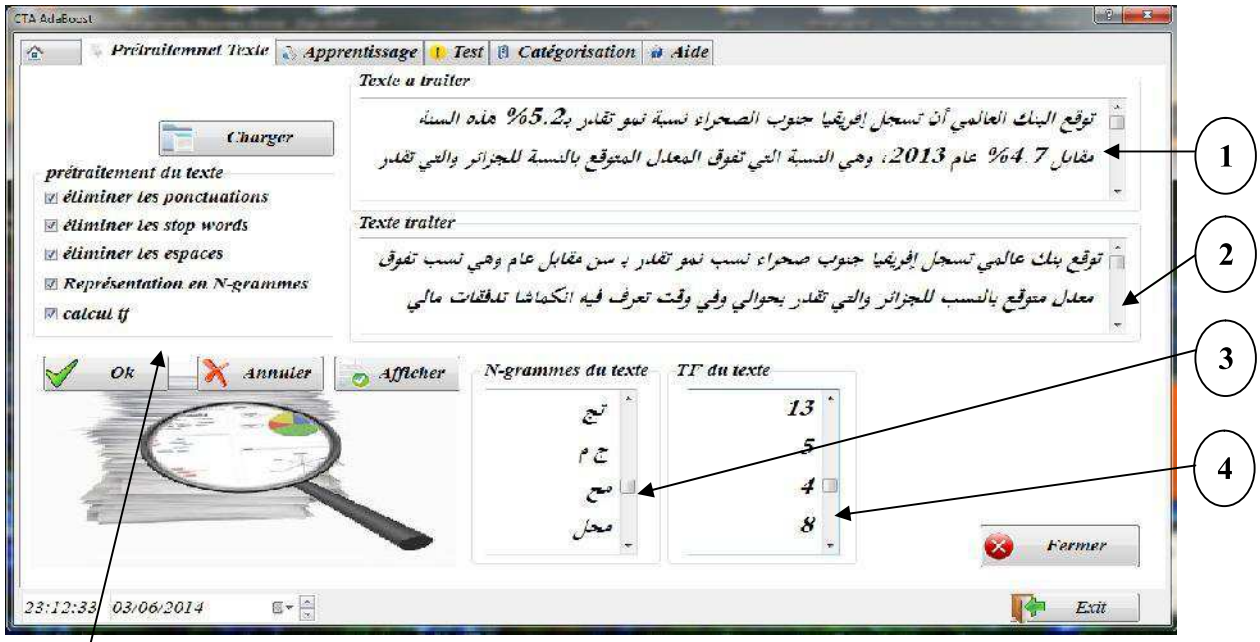

Figure 4.5 *Fenêtre «prétraitement d 'un texte ».*

1: Le texte avant prétraitement

2: Le texte âpres prétraitement

3: Le vecteur 3-grammes associé au texte

4: Le vecteur de fréquence TF du texte

5: Les prétraitements appliqués sur le texte

4.4.2. Dictionnaire des termes des différentes catégories

Nous avons construit un dictionnaire pour chaque catégorie à partir d'un corpus d'apprentissage, les dictionnaires construits, servent à aider le classifieur CTA AdaBoost lors de la phase de test et de catégorisation.

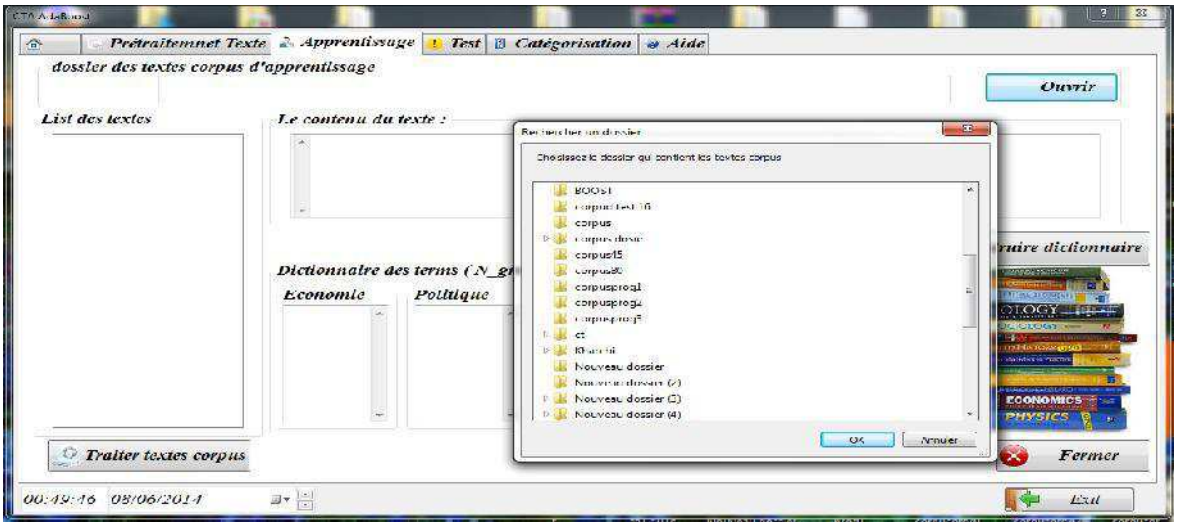

Figure 4.6 *Fenêtre « Chargement d'un corpus d'apprentissage ».*

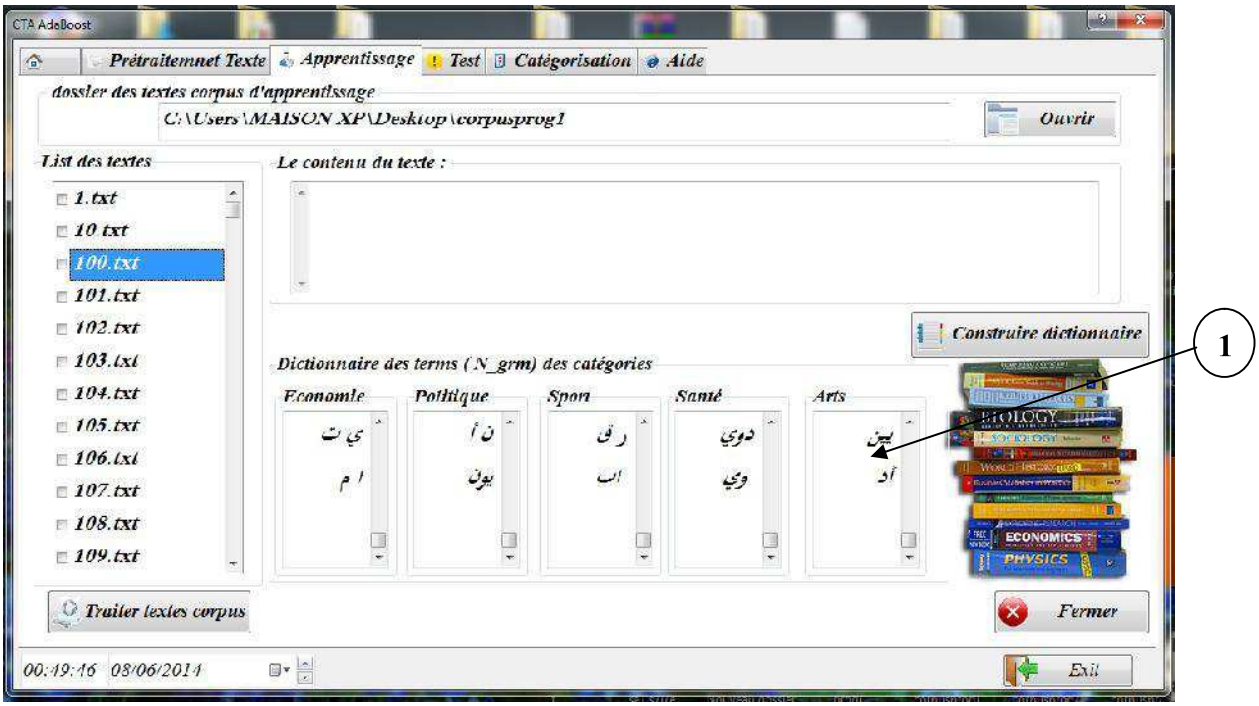

Figure 4.7 *Fenêtre « La phase d'apprentissage ».*

1 : Les dictionnaires des différentes catégories.

4.4.3. Test du classifieur *CTA_AdaBoost*

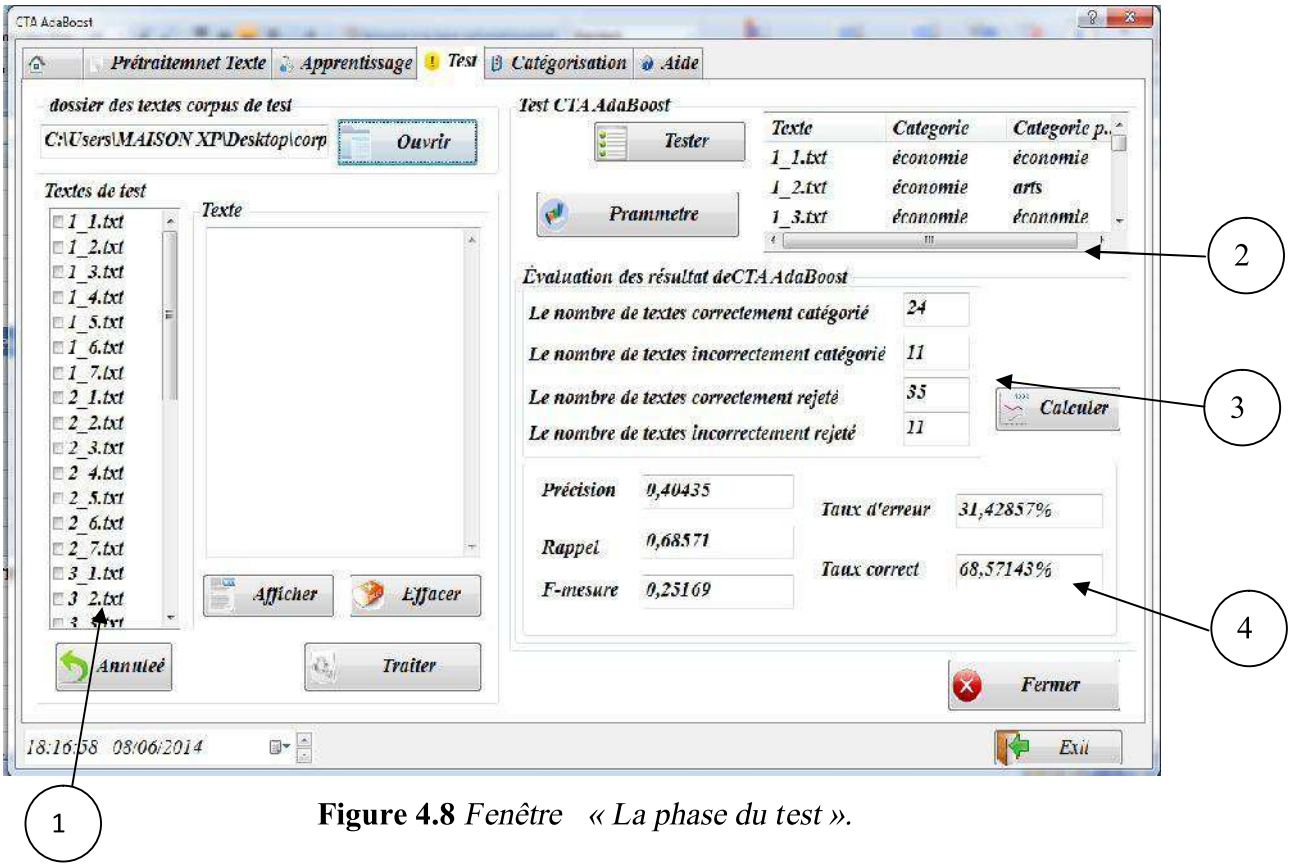

Figure 4.8 *Fenêtre « La phase du test ».*

1: Les textes de corpus de test

2: Les résultats de test

3: Les nombres des textes (correctement / incorrectement) catégorisé et rejeté par « CAT_AdaBoost »

4: Les résultats de évaluation de test « CAT_AdaBoost » (Rappel ,Précision ,F_mesure, Taux d'erreur, taux correct)

4.4.4. Catégorisation des textes arabes par notre classifieur *CTA_AdaBoost*

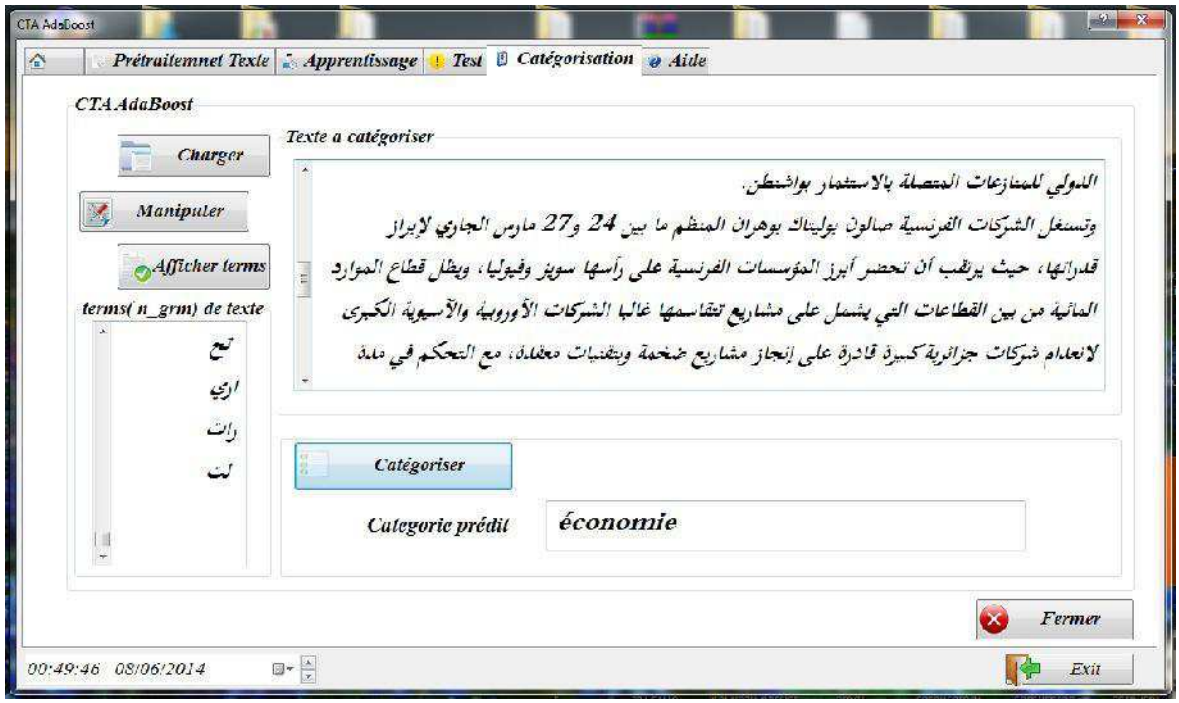

Figure 4.9 *Fenêtre « La phase de catégorisation ».*

4.5. Expérimentation et évaluation

Afin d'évaluer et valider la contribution présentée dans ce mémoire, une phase d'expérimentation s'avère indispensable. Cette phase a pour objectif d'étudier les performances de notre approche implémentée. En outre, ceci nous permettra aussi d'identifier les contraintes et les insuffisances de notre approche. La suite sera consacrée à l'évaluation des résultats obtenus.

4.5.1. Evaluation

Corpus de test(nombre de textes)	35
Le nombre de textes bien catégorisés	24
Le nombre de textes mal catégorisés	11

Table 4. 4 *Nombre de textes (bien /mal) catégorisés du corpus de test.*

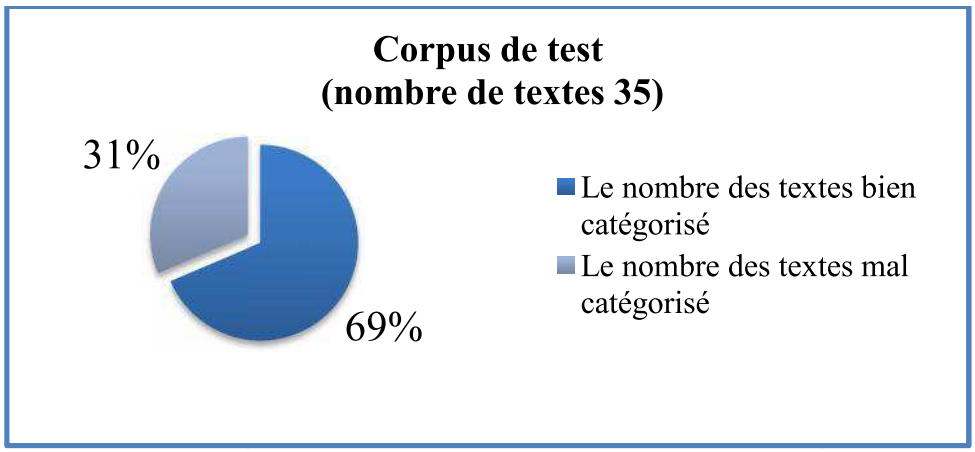

Figure 4.10 *Nombre de textes (bien /mal) catégorisés du corpus de test.*

Catégories	Le nombre de textes	Le nombre de textes bien catégorisés pour chaque catégorie	Le nombre des textes mal catégorisés pour chaque catégorie
Economie	7	6	1
Politique	7	6	1
Sports	7	5	2
Santé	7	3	4
Arts	7	4	3

Table 4. 5 *Nombre de textes (bien /mal) catégorisés pour chaque catégorie.*

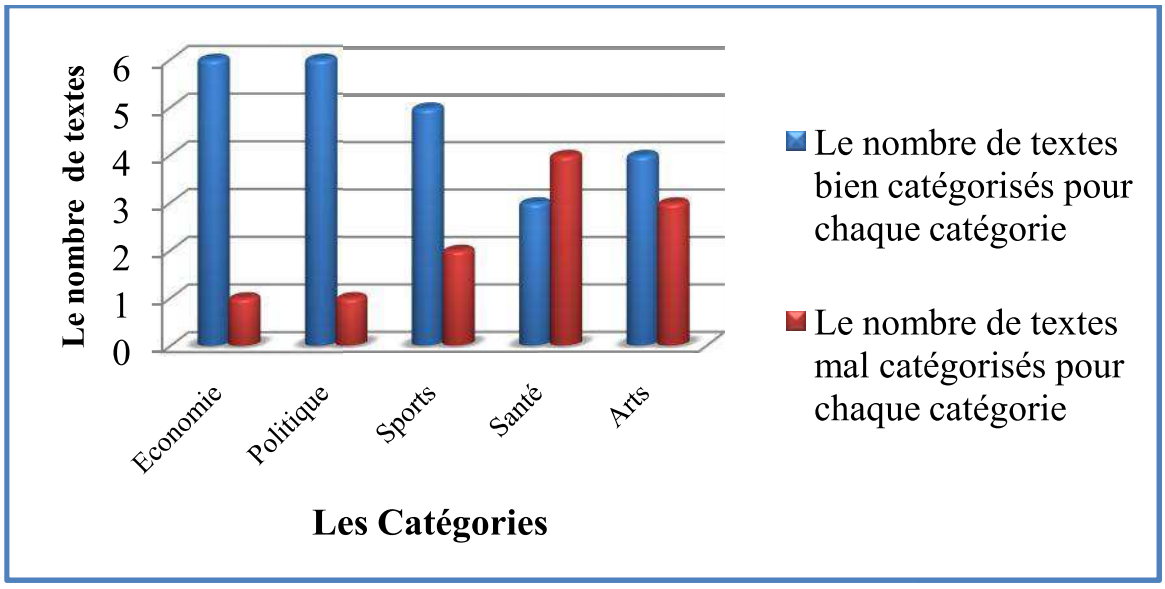

Figure 4.11 *Résultats du test par catégorie issus du clasifieur « CTA_AdaBoost».*

Catégories	Rappel	Précision	F_mesure	Taux _erreur	Taux _correct
Economie	0,85	0,40	0,27	0,14	0,85
Politique	0,85	0,42	0,28	0,14	0,85
Sports	0,71	0,45	0,27	0,28	0,71
Santé	0,42	0,37	0,22	0,57	0,42
Arts	0,57	0,36	0,22	0,42	0,57

Table 4.6 *Résultats de l'évaluation pour chaque catégorie.*

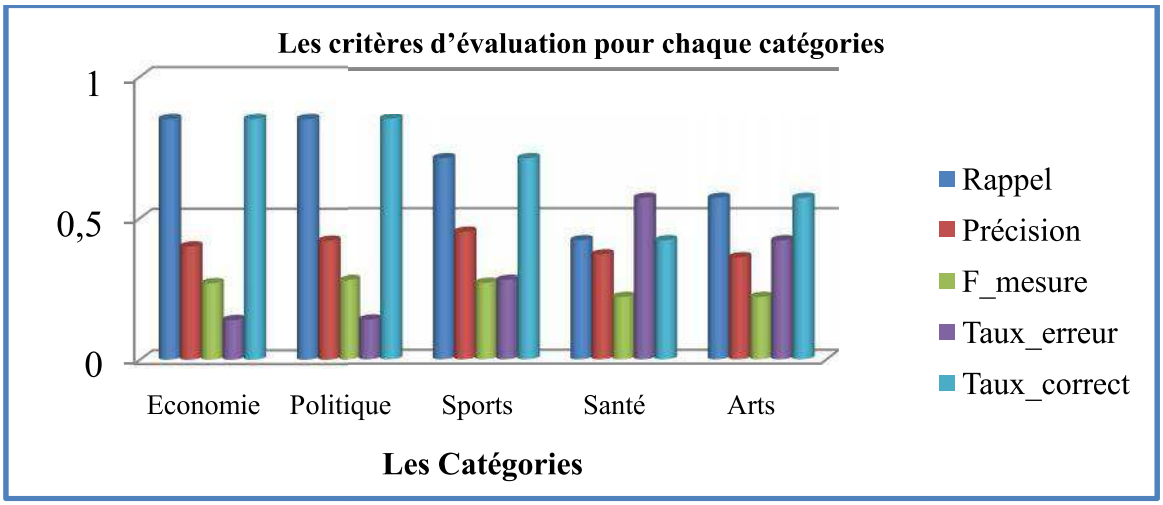

Figure 4.12 *Résultats d' évaluation pour chaque catégorie.*

Rappel	Précision	F_mesure	Taux_erreur	Taux_correct
0,68	0,40	0,25	0,31428	0,68

Table 4. 7 *Résultats de l'évaluation du classifieur « CTA_AdaBoost».*

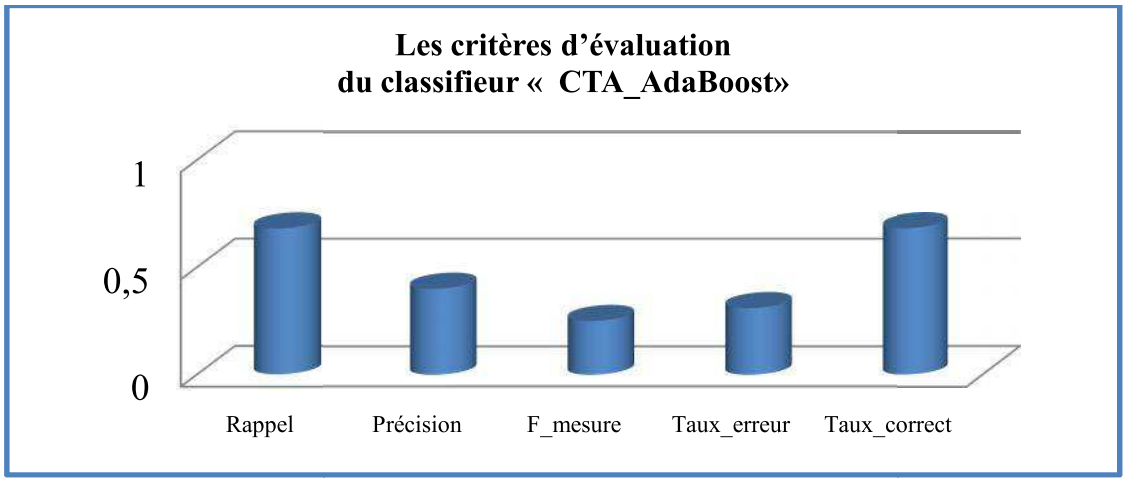

Figure 4.13 *Résultats de l'évaluation du classifieur « CTA_AdaBoost».*

4.5.2. Discussion

L'analyse et l'observation de ces expérimentations montre que les résultats Obtenus sont très satisfaisants pour certaines catégories telles que : la politique (taux_correct = 85,71%) , l'économie (taux_correct = 85,71%), sport(taux_correct = 71 ,42%) . Et des résultats acceptables dans les catégories restantes : santé (taux_correct = 42,85%), arts(taux_ correct = 57,14%) . On souligne ici que plusieurs facteurs peuvent influencer sur les résultats obtenus, tels que la représentation des textes en (3_gramme) qui peut être efficace pour certaines catégories et non pas pour certaines autres, et la même chose pour seuil choisi dans la phase de réduction vocabulaire. De même pour la sélection du corpus d'apprentissage. Sans oublier que l'utilisation du langage arabe naturel pose un grand problème dans la phase de la catégorisation. En effet, les problèmes rencontrés à ce niveau sont dans la plupart du temps, incontournables même chez l'être humain à cause du phénomène de l'ambigüité à tous les niveaux de l'analyse du texte.

Mais finalement, nous pouvons dire que les techniques utilisées peuvent être considérées comme satisfaisantes et relativement acceptables.

Conclusion

Dans le présent chapitre, nous avons illustré quelques résultats pratiques sur la catégorisation thématique des textes arabes en utilisant l'algorithme de Adaboost. Les résultats obtenus sont très significatifs et peuvent être améliorés dans des travaux futurs en modifiant certains paramètres tels que : les tailles des corpus utilisés, l'approche de représentation des textes adoptée, le seuil de réduction du vocabulaire choisi, et même la variante de l'algorithme Adaboost implémentée, … etc. d'où nous pouvons dire que les résultats sont très satisfaisants et par conséquence, l'objectif fixé au départ est atteint.

CONCLUSION GENERALE

Conclusion générale

La catégorisation automatique de textes est une phase très importante dans le domaine de la recherche de l'information. L'importance de la catégorisation de textes vient en premier lieu de la demande croissante des humains en information pertinente fouillée dans des centaines de milliers de documents numériques disponibles sur internet et la difficulté d'accéder à ces informations d'une façon manuelle..

La catégorisation de textes a connu une progression très considérable pendant les dix dernières années grâce à l'introduction de nouvelles techniques inspirées de l'apprentissage automatique ce qui a amélioré très significativement la qualité de la classification obtenue (le taux de succès). Cependant, il est toujours difficile de fournir des valeurs chiffrées sur les performances qu'un système de classification peut atteindre. Surtout pour la catégorisation des textes arabes, car la langue arabe est une langue très riche et sensible au contexte d'où elle représente de nombreux défis dans des applications divers telles que : la recherche de l'informations, le traitement automatique du langage naturel, etc...

la Le traitement des textes arabes rencontre des difficultés énormes qui restent toujours objet de recherche, notamment : le problème de l'ambiguïté issue de l'absence des voyelles, le problème de la reconnaissance des formes fléchies (la langue arabe étant fortement flexionnelle) et le problème d'absence de travaux publiés sur l'extraction de l'information à partir d'un texte arabe en utilisant des modèles statistiques du langage, s'ajoute à tout cela, la diversité des techniques et des méthodes relatives au processus de catégorisation qui pose un problème de choix, tout cela pose un énorme défi difficile à surmonter.

Malgré tous ces problème, nous avions le courage de prendre l'initiative et de travailler sur la catégorisation thématique des textes arabes ce qui nous a permis d'Apprendre beaucoup de choses tout au cours de la réalisation de ce projet telles que la programmation orientée objets avec le langage Delphi, des notions de base dans le domaine de la catégorisation automatique de textes ATC et l'apprentissage

automatique ML (les algorithmes d'apprentissage et en particulier l'algorithme Adaboost), le traitement automatique du langage naturel TALN (surtout pour la langue arabe).

Malheureusement, le temps qui nous a été attribué était très court, d'où il était nécessaire de concentrer sur certains concepts et paramètres et de négliger d'autres. Par exemple, il était intéressant pour nous d'observer le comportement de la méthode Adaboost sur la catégorisation multi-classe multi-labelle sur plus d'un corpus de textes et en mettant l'accent sur la variabilité de la langue (le multilinguisme). Notre perspective dans un premier temps, est de consolider l'algorithme implémenté en l'évaluant sur d'autres collections de textes, ensuite d'élargir notre étude vers d'autres langues (ajouter d'autres langues pour rendre notre classifieur multi-langues) et pourquoi pas sur des corpus de textes plus volumineux .

BIBLIOGRAPHIE

ET

WEBOGRAPHIE

BIBLIOGRAPHIE ET WEBOGRAPHIE

[1] Jalam, Radwan . "Apprentissage automatique et catégorisation de textes multilingues". Thèse de doctorat, Université Lumière Lyon 2, (2003) .

[2] Fabrizio Sebastiani. Guest editors' introduction to the special issue on the 25th European Conference on Information Retrieval Research. *Information Retrieval*,2004, pp 235-237.

[3] Turenne Nicolas, " Etat de l'art de la classification automatique pour l'acquisition de connaissances à partir de textes". UMR INRA-INAPG – Biométrie et Intelligence Artificielle (BIA). INRIA, Technical report. 2001.

[4] Réhel, Simon, " Catégorisation automatique de textes et cooccurrence de mots provenant de documents non étiquetés " . ,Maître ès sciences (M.Sc.)Université Laval FACULTÉ DES SCIENCES ET DE GÉNIE Maîtrise en informatique , 2005 .

[5] NAKACHE Didier, " Extraction automatique des diagnostics à partir des comptes rendus médicaux textuels" . thèse doctorat, laboratoire CEDRIC, équipe ISID, CNAM de Paris , soutenu le 26 septembre 2007 .

[6] Caropreso Maria Fernanda, Stan Matwin , Fabrizio Sebastiani, " Statistical Phrases in Automated Text Categorization " ,Department of Computer Science of the University of University of Ottawa supported by Grant FOMEC376 from the Universidad de Buenos Aires, 2000.

[7] Furnkranz Johannes, Tom Mitchell, Ellen Rilo, « A Case Study in Using Linguistic Phrases for Text Categorization on the WWW » School of Computer Science Carnegie Mellon University, 1998.

[8] M. F. Porter. "An algorithm for sufix stripping". Program, Morgan Kaufmann Publishers Inc, 1980, pp 130–137.

[9] Camelia Ignat, " Représentation de textes a l'aide d'étiquettes sémantiques dans le cadre de la classification automatique " , European Commission, IPSC, Strasbourg, France, 2007.

[10] Fletcher Pratt, "*Secret and Urgent: The Story of Codes and Ciphers*", 1939.

[11] C. E. Shannon, " A mathematical theory of communication ". Bell system technical journal, 1948.

[12] Nicolas BÉCHET, « Extraction et regroupement de descripteurs morphosyntaxiques pour des processus de Fouille de Textes » , thèse de doctorat , Université des Sciences et Techniques du Languedoc, Soutenue le 8 décembre 2008 .

[13] SOUISSI .E, "Étiquetage grammatical de l'arabe voyellé ou non ",Thèse de doctorat, Université de Paris III, Soutenue le 8 décembre 2008.

[14] Lefèvre, Philippe. *La recherche d'informations : du texte intégral au thésaurus.* Paris :Hermès Science Publications, 2000. p 253.

[15] Uren, V. S. "evaluation of text categorization errors " . In *Proc. Workshop on the Evaluation of Information Management Systems*, London, 15 September 2000 , pp 79–87.

[16] Fabrizio Sebastiani." A tutorial on automated text categorization". In Analia Amandi and Alejandro Zunino (eds.), *Proceedings of the 1st Argentinian Symposium on Artificial Intelligence (ASAI 1999),*Buenos Aires, AR, 1999, pp. 7-35.

[17] Cleverdon," Optimizing convenient online access to bibliographic databases". *Information Services and Use* , 1984, p 39.

[18] Fabrizio Sebastiani, Machine learning in automated text categorization. ACM Computing Surveys, 2002.

[19] I. Androutsopoulos, J. Koutsias, K.V. Chandrinos and C.D. Spyropoulos, "An Experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Encrypted Personal E-mail Messages". In Belkin, N.J., Ingwersen, P. and Leong, M.-K. (Eds.), Proceedings of the *23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR 2000), Athens, Greece, pp. 160-167, 2000.

[20] KESSLER Rémy, Juan Manuel TORRES-MORENO et Marc EL-BEZE ," Classification thématique de courriels avec apprentissage supervisé, semi supervisé et non supervisé" , Laboratoire d'Informatique d'Avignon Universite d'Avignon, 2004.

[21] Fabrizio Sebastiani, "Text categorization ", In Laura C.Rivero, Jorge H. Doorn and Viviana E. Ferraggine (eds.), The Encyclopedia of Database Technologies and Applications,Idea Group Publishing, Hershey, US, 2005, pp. 683-687

[22] Tzeras. K et Hartmann. S, " Automatic indexing based on Bayesian inference networks ". In Korfhage, R.,Rasmussen E., and Willett, P., editors, Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval, pp 22– 34, Pittsburgh, US. ACM Press, New York, US. 1993.

[23] Sadik Bessou, " Analyse de Données Textuelles pour la Classification Automatique par les Techniques de Text Mining, application à la Langue Arabe ", Mémoire de Magister En Informatique , Université de Sétif,2007.

[24] F.Yvon ,« Des apprentis pour le traitement automatique des langues », Mémoire pour l'obtention de l' Habilitation à Diriger des Recherches ,Université Pierre et Marie Curie - Paris VI,2006.

[25] Guillaume Cleuziou , " une méthode de classification non supérvisé pour l'apprentissage des règles et la recherche d'information " ,thèse de Doctorat de Université d'Orléans , 2004 .

[26] Kiri Wagsta ,Claire Cardie ," Constrained K-means Clustering with Background Knowledge" , the Eighteenth International Conference on Machine Learning, 2001, p. 577-584.

[27] Romain VINOT ,2004 , « classification automatique de textes dans la catégorie non thématique » , thèse pour obtenir le grade de docteur d'école nationale supérieur des télécommunication ,Présentée et soutenue le 09 février 2007.

[28] Guillaume OBOZINSKI, « Introduction aux modèles graphiques », cours, Décembre 2010-2011.

[29] Zighed, D. A. et Rakotomalala, R , " Graphes d'induction. Apprentissage et Data Mining " , Hermes Science Publication, Paris,2000.

[30] Romain Vinot, Natalia Grabar, Mathieu Valette , " Application d'algorithmes de classification automatique pourla détection des contenus racistes sur l'Internet " ,Centre de Recherche en Ingénierie Multilingue –Université Paris 6 UMR 7114 CNRS/Paris X (MoDyCo) , (2003) .

[31] Yasmine Hanane et Zeggane Mokhtar, " Algorithmes d'apprentissage pour la classification de documents ", Mémoire de licence en ligne, Université de Mostaganem Algérie, 2009.

[32] Cyril Grouin, Martine Hurault-Plantet, Patrick Paroubek, Jean-Baptiste Berthelin, " DEFT'07 : une campagne d'évaluation en fouille d'opinion " , LIMSI–CNRS, Orsay, France, 2009.

[33] Kadri Youssef, "Recherche d'Information Translinguistique sur les Documents en Arabe " , Thèse présentée à la Faculté des études supérieures en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.) en informatique ,septembre 2008.

WEBOGRAPHIE

[1] François Denis et Rémi Gilleron. Techniques de boosting en apprentissage automatique. Report équipe Grappa, http://www.grappa.univlille3.fr/ftp/reports/boosting.ps.gz , consulté le : 03 /06/2014.

[2] Cet exemple est pris de « A Boosting Tutorial » par Robert Schapire http://videolectures.net/mlss05us_schapire_b/ , consulté le : 03 /06/2014.

Annexe

Annexe 1

Nous présentons dans cette annexe ,la liste des mots vides, proposée notons que la notre contient 450 mots .

الكلمة بيد سوى غير لاسيما بكم بما بماذا بمن كم كيف ما ماذا متى مما ممن من أنّى أيّ ّأيّان أين أينما حيثما كيفما ما متى من مهما أولئك أولئكم أولاء أولالك تان تانِك تلك تلكم تلكما تِه تِي تَيْن تينك ثمّ ثمّة ذا ذاك ذان ذانك ذلك ذلكم ذلكما ذلكن ذِه ذوا ذواتا ذواتي ذِي ذَيْن ذينك كذلك هَؤلاء هَاتان هَاتِه هَاتِي هَاتَيْن هاهنا هَذا هَذان هَذِه هَذِي هَذَيْن هكذا هنا هناك هنالك أي إذ إذا بعض تجاه تلقاء جميع حسب حيث سبحان سوى شبه كل لعمر لما مثل مع معاذ نحو أقل أكثر إذاً آهاً بسّ حاي صهْ صهٍ طاق طَق عَدَسْ كِخ نَخْ هَجْ وا واهاً وَيْ آمينَ آه أُفٍّ أُفٍّ أمامك أمامكَ أوّهْ إلَيْكَ إلَيْكَ إليكَ إليكم إليكم إليكما إليكنّ إيهٍ بخٍ بَسْ بطآن بَلْهَ حَذارِ حيَّ حيَّ دونك رويدك سرعان شَتَّانَ عليك مكانَك مكانَك مكانَك مكانكم مكانكما مكانكنّ مه ها هاؤم هاكَ هلمَّ هيّا هيت هَيْهَاتَ وا وراءَك وُشْكَانَ ويكأنّ الألاء الألى التي الذي الذين اللائي اللاتي اللتان اللتيا اللتين اللذان اللذين اللواتي أَيُّ ذا ذات ما مَنْ أب أخ حم ذو فو لن لو لولا لوما نعم بِئْسَ حَبَّذَا سَاءَ سَاءَمَا نِعْمَ نِعِمّا إن لات ما لا أنَّ إنَّ علّ كَأَنَّ لَعَلَّ لَكِنَّ لَيْتَ آي كي أجمع جميع عامة عين كل كلا كلاهما كلتا كليكما كليهما نفس ء ؤ ئ آ أ أ ب ت ة ث ج ح خ د ذ ر ز س ش ص ض ط ظ ع غ ف ق ك ل م ن ه و ى ي إلّا حاشا خلا عدا لكن فيم فيما هل سوف هلّا قد إمّا كأنّما كما لكي لكيلا إلى رُبَّ على عن في مِن عَمَّا حَتَّى منذ مذ لم لمّا أجل إذن إي بلى جلل جير كلَّا إذما لئن أمّا ألا أما أم أو بل ثُمَّ أيا هيا أن بك بكم بكما بكن بنا به بها بي لك لكم لكما لكن لنا له لها لي أنا أنت أنتِ أنتم أنتما أنتن نحن هم هما هن هو هي إياك إياك إياكم إياكما إياكن إيانا إياه إياها إياهم إياهما إياهن إياي دون ريث عند عوض قبل قطْ كلّما لدن لدى لمّا الآن آناء آنفا أبدا أصلا أمد أمس أول أيّان إذ بعد تارة حين صباح ضحوة غدًا غداة مرّة مساء يومئذ خلال أمام أين إزاء بين تحت ثمّ خلف شمال ضمن فوق يمين حوالى حول طالما

قلما ابتدأ اخلولق انبرى أخذ أقبل أنشأ أوشك جعل حرى شرع طفق عسى علق قام كاد كرب هبّ إنّما لكنَّما ارتدّ استحال انقلب آض أصبح أضحى أمسى بات تبدّل تحوّل حار راح رجع صار ظلّ عاد غدا كان ماانفك مابرح مادام مازال مافتئ لَيْسَ لَسْتُ لَسْنَا لَسْتَ لَسْتِ لَسْتُمَا لَسْتُم لَسْتُنَّ لَيْسَتْ لَيْسَا لَيْسَتَا لَيْسُوا لَسْنَ بضع ذيت فلان كأيّ كأين كأيّن كذا كم كيت

ملخص

في إطار مذكرة التخرج هذه , قمنا بإنجاز برنامج يسمح بالتصنيف الآلي للنصوص العربية بالاعتماد على المحتوى أوالمضمون,,وهذا بناءا على أنواع معروفة مسبقا من الأصناف وهي (الاقتصاد, السياسة، ، الفن والرياضة) ،و لقد استخدمنا في ذلك خوارزما جديدا يسمى Adaboost .

الكلمات المفتاحية : التصنيف الآلي للنصوص،النصوص العربية، خوارزم التصنيف ,Adaboost.

Résume

Dans le cadre de ce mémoire, nous avons réalisé un système automatique pour la classification de documents arabes (un classifieur) en basant sur le contenu ou le contexte. Les classes ou les catégories auxquelles on affecte les textes sont prédéfinies au préalable (Economie, Politique, arts, Sport). Pour l'apprentissage, nous avons exploité l'un des algorithmes les plus efficace dans le domaine qui est l'algorithme Adaboost.

Mot clé : *Catégorisation automatique, Algorithme d'apprentissage, texte arabe, Adaboost.*

Abstract

In this thesis, we have realized an automatic system for the classification of Arabic documents (a classifier) bas ing on content or context. Classes or categories which we assign to the texts are predefined in advance (Economy, Politics, Arts, Sports). For learning, we used one of the most efficient algorithm in the field which is the Adaboost algorithm.

Key word:*Automatic Categorization, learning Algorithms, Arabic text, Adaboost.*