République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Ecole Doctorale Nationale Science et Technologie de l'Information et de la Connaissance

Mémoire de Magister

Présenté à l'Université de Guelma
Faculté des sciences et de l'ingénierie

Département de : **Informatique**
Spécialité : **Informatique**
Option : **SIC**

Présenté par : **Mlle. Samira Legrini**

Les Ngrammes dans la Catégorisation Automatique de Textes Arabes

---

JURY

Président	**MC**	**Labiba Souici**	**Université d'Annaba**
Rapporteur	**Pr**	**Hamid Seridi**	**Université de Guelma**
Examinateur	**MC**	**Halima Bahi**	**Université d'Annaba**
Examinateur	**MC**	**Hayet Merouani**	**Université d'Annaba**

2009

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Ecole Doctorale Nationale Science et Technologie de l'Information et de la Connaissance

Mémoire de Magister

Présenté à l'Université de Guelma
Faculté des sciences et de l'ingénierie

Département de : **Informatique**
Spécialité : **Informatique**
Option : **SIC**

Présenté par : **Mlle. Samira Legrini**

Les Ngrammes dans la Catégorisation Automatique de Textes Arabes

JURY

Président	**MC**	**Labiba Souici**	**Université d'Annaba**
Rapporteur	**Pr**	**Hamid Seridi**	**Université de Guelma**
Examinateur	**MC**	**Halima Bahi**	**Université d'Annaba**
Examinateur	**MC**	**Hayet Merouani**	**Université d'Annaba**

2009

Remerciements

En premier lieu, je tiens à exprimer ma profonde gratitude à mon encadreur Monsieur Hamid Seridi de m'avoir proposé ce sujet et diriger constamment de près mon travail. Je le remercie pour sa disponibilité, ses encouragements, son soutien moral, sa sympathie et son amitié sincère dont il a fait preuve le long de ce travail. J'apprécie la confiance qu'il m'a témoignée et les conseils avisés qu'il m'a prodigué. Qu'il trouve ici mes sincères reconnaissances.

Je tiens aussi à remercier Monsieur Djelailia Karim pour son aide, ses inestimables conseils, ses explications claires et ses orientations précises. Qu'il trouve ici mes sincères reconnaissances.

Merci à Madame Suici Labiba de m'avoir fait l'honneur de présider mon jury, je suis très reconnaissante à elle pour l'intérêt qu'elle a porté à mon travail.

Je remercie également tous les membres de l'équipe SERIDI pour leur soutien moral et leur encouragement pour l'accomplissement de ce travail.

Je ne serais sans doute pas arrivé jusqu'ici sans avoir suivi les cours d'enseignants exceptionnels qui ont participé à ma formation, je pense en particulier aux enseignants du département informatique de Guelma et aux enseignants de l'École Doctorale de Guelma.

Enfin, je remercie mes collègues : samia drissi , Hallaci Samir, Farou Brahim, Halimi Khaled, Zedadra Ouarda, toualbia ilyes pour l'amitié et le soutien qu'ils m'ont apportés durant la période d'étude et de préparation de ce travail.

Résumé

La catégorisation automatique de textes est un domaine de recherche en plein essor, en raison de l'explosion de la quantité d'information disponible sous format électronique, et la difficulté d'accéder à l'information pertinente parmi toutes celles qui sont accessibles. Son principal enjeu est de rendre une application informatique capable d'assigner d'une façon autonome une catégorie à un document en se basant sur son contenu. Pour décrire le contenu des documents, la quasi-totalité des systèmes actuels se base sur la représentation sac de mots en raison de sa simplicité. Néanmoins avec une telle représentation le sens de termes dans la majorité des cas reste ambigu, de plus la description de certains concepts nécessite l'utilisation de quelques mots pris simultanément, mais pas séparément. Dans ce cas, l'utilisation des mots simple pour décrire ces concepts va engendrer une ambigüité sémantique

L'objectif de ce mémoire est de proposer une approche qui tente de réduire cette ambigüité et d'améliorer les performances des systèmes de catégorisation de textes arabes en se basant sur des descripteurs plus informatifs et plus discriminants que les mots. L'idée de base de cette approche consiste à bénéficier des avantages liés à utilisation des N-grammes et plus précisément les unigrames et les bi-grammes ayant un apport informationnel élevé pour la représentation des documents, et de tester leur influence sur les performances globales des systèmes de catégorisation de textes arabes.

Afin d'évaluer cette approche, nous utilisons comme classifieur les machine à vecteur support (SVM), et comme base d'apprentissage un corpus en langue arabe. Notons que le choix des SVM est dû essentiellement à leur robustesse ainsi à leur capacité à traiter des espaces de données de grande dimensionnalité.

Mots clés : catégorisation automatique de textes, langue arabe, N-gramme, SVM.

Abstract

Text categorization also known as text classification, is a field of research in full expansion, due to the explosion of the quantity of information available in electronic format, and the difficulty to access to relevant information among all those that are accessible.

At present, text categorization techniques are predominantly keyword-based. With such representation term sense in the majority of the cases remain ambiguous, and more concepts cannot be described by single words, which engender a semantic ambiguity.

The aim of this study is to present our approach which tries to reduce this ambiguity and to improve the performances of Arabic text categorization, by investigating other feature more informative and more discriminates then single words. The basic idea of this approach consists of using ngram and more exactly unigram and bigrams having a high informative contribution as feature in Arabic text categorization.

To evaluate this approach we use SVM as classifier and Arabic corpora as training base. The choice of the SVM is motivated by its high performance in text categorization

***Key word**: text categorization, Arabic language, Ngram, SVM*

ملخص

إن التّصنيف الآلي للنصوص هو من بين مجالات البحث الآخذة في الاتساع بسبب انفجار كمية المعلومات المتاحة في شكل الكتروني, و صعوبة الوصول إلى المعلومات المهمة. الهدف الأساسي من هذا المجال هو تصنيف النصوص بصفة أوتوماتيكية استنادا إلى مضمونها.

لتمثيل النصوص نعتمد في اغلب الأحيان على الكلمات المفردة, و هو ما يسمى بتقنية sac de mot و ذلك لبساطتها و سهولة تطبيقها, إلا أن هذه التقنية تعاني من مشكلة الغموض, فباستعمال الكلمات المفردة لا يمكننا التعبير عن كل المفاهيم.

الهدف من هذه المذكرة هو تقديم طريقة جديدة في التصنيف الآلي للنصوص باللغة العربية, وذلك بالاستناد إلى تقنية Ngram حيث نعتمد على عدة كلمات بالإضافة إلى الكلمات المفردة في تمثيل النصوص. هدفنا هو اختبار مدى تأثير هذه التقنية في مجال تصنيف النصوص باللغة العربية.

من اجل تقويم هذه الطريقة, قمنا بإجراء عدة تجارب على مجموعة من النصوص باللغة العربية و ذلك باستعمال SVM كبرنامج للتصنيف. سبب اختيارنا لهذا البرنامج هو مدى فعاليته في هذا المجال.

كلمات البحث: اللغة العربية, التصنيف الالي للنصوص,SVM,Ngramme

Table des matières

3 La langue arabe

4 Approche proposée

5 Expérimentations et Résultats

Table des figures

Liste des tableaux

Liste des algorithmes

Abréviations

ACM	:	Association for Computing Machinery
AFP	:	Agence France-Presse
ASCII	:	American Standard Code for Information Interchange
CRF	:	Conditional Random Field
ELRA	:	European Language Resources Association
IA	:	Intelligence Artificielle
LDC	:	Language Data Consortium (Pennsylvania)
LSI	:	Latent Semantic Indexing
NLP	:	Natural Language Processing
PLSA	:	Probabilistic Latent Semantic Analysis
RI	:	Recherche d'Information ou IR pout Information Retrieval
RD	:	Recherche Documentaire
SIGIR	:	Special Interest Group on Information Retrieval
SVM	:	Support Vector Machines
TALN	:	Traitement Automatique du Langage Naturel
TC	:	Text Categorization
TF*IDF	:	Term Frequency Inverse Document Frequency
TREC	:	Text REtrieval Conference
XML	:	Extensible Markup Language
YALE	:	Yet Another Learning Environment

Introduction

De nos jours, l'explosion des technologies de l'information a fait de l'Internet un outil incontournable tant au niveau personnel que professionnel, occupant une place prépondérante dans notre société et notre vie. Le web nous offre un monde de l'information prodigieux sans limite. Le nombre de documents disponibles sous format électronique ne cesse pas à croitre d'un jour à l'autre. Le courriel devient un moyen de communication extrêmement populaire, ce qui pourrait apparaitre comme quelque chose de véritablement bénéfique. D'un autre côté, cette ouverture à un monde inexplorable serait sans intérêt si notre capacité à y accéder efficacement n'augmentait pas elle aussi. Pour en profiter au maximum, on a besoin d'outils nous permettant de chercher, classer, mettre à jour et analyser les données accessibles. Il nous faut des outils nous aidant à trouver dans un temps raisonnable l'information désirée, ou, tout au moins, nous facilitant la tâche.

Un des domaines prometteurs qui s'intéressent à cet objectif est la catégorisation automatique de textes. Imaginons-nous en présence d'une banque considérable de textes, qui seraient plus aisément accessibles s'ils étaient répartis dans un ensemble de catégories en fonction de leur sujet. Évidemment, on pourrait demander à un humain de lire tous les textes, et de les classer manuellement. Mais la tâche s'avère colossale s'il fait face à des centaines, voire des milliers de documents. Il apparait alors très intéressant de pouvoir compter sur une application informatique, qui de façon automatique, assigne ces textes à un ensemble prédéfini de catégories. C'est précisément là, le but de la classification automatique de textes. On peut imaginer l'utilité d'un tel processus et ses applications multiples. Par exemple, l'organisation des documents internes d'une compagnie de documents tels les annonces classées qu'un journal a gérées, les courriels reçus par un individu, les demandes de brevet acheminées à un office international. Pensons aussi au filtrage d'articles provenant d'une agence de presse ou à la détection de courriel indésirable (*spam*). La classification automatique de textes pourrait aussi faciliter la recherche de pages Web par leur triage dans une hiérarchie de style Yahoo! Et la liste ne s'arrête pas ici, loin de là.

Les approches actuelles visant à créer des classifieurs automatiques de textes sont directement liées à l'apprentissage automatique. Ce sous-domaine de l'intelligence artificielle qui s'intéresse à conférer aux machines la capacité de s'améliorer à l'accomplissement d'une tâche, en interagissant avec leur environnement. L'apprentissage automatique se divise principalement en deux façons d'apprendre : l'apprentissage supervisé et l'apprentissage non supervisé. C'est dans l'approche dite supervisée que s'inscrit la façon dont on aborde aujourd'hui le problème de la catégorisation automatique de textes. Dans ce paradigme, l'apprentissage s'effectue à partir d'un ensemble d'exemples où chacun d'eux est constitué d'un objet d'entrée et d'une valeur de sortie désirée pour cet objet. En connaissant les sorties prévues, l'algorithme peut généraliser les exemples afin d'identifier les différents attributs des objets qui justifient une sortie particulière, pour devenir en mesure de traiter de nouvelles données. Ce processus correspond exactement à ce qui est souhaité en catégorisation de textes, qui peut être vu comme un processus d'apprentissage sur une banque de documents textuels dont la classe d'appartenance est connue du classifieur. Évidemment, ces documents textuels sont écrits en langage naturel, qui n'est pas interprétable par les classifieurs. Donc une étape de représentation est fortement nécessaire.

L'objectif de cette étape est de décrire le contenu thématique des documents. La quasi-totalité des systèmes de catégorisation automatique (TC) représente les documents par la présence/absence de termes dans le texte. Ces termes sont les unités minimales constitutives d'un texte. Ils peuvent être plus ou moins complexes : mots, racines de mots groupe de mots ou expression. Dans tous les cas, la séquentialité des mots dans le document est irrémédiablement perdue, et l'on ne retient que le nombre d'occurrences du terme. C'est la métaphore du sac de mots. Un document est alors représenté par un vecteur de termes, tel que chaque terme doit correspondre à un trait sémantique.

Dans la majorité des systèmes, les termes correspondent aux mots, car ils sont faciles à segmenter et à détecter à partir de texte. Néanmoins, le sens des mots reste ambigu dans de nombreux cas (polysémie, contexte), et plusieurs mots peuvent correspondre au même concept (synonymie). De plus certains concepts ne peuvent pas être décrits par des mots

simples. Une combinaison de mots prise entièrement possède une signification bien spécifique que la signification des mots pris séparément

Dans la majorité des cas, les bigrammes (deux mots successifs) forment une bonne approximation des concepts, donc ils peuvent construire des descripteurs plus discriminants que les mots. Les bigrammes sont plus informatifs que les mots et permet la désambigüisation des termes d'indexation qui est un critère important pour choisir les descripteurs.

A cet effet, l'objectif de ce mémoire est de proposer une approche à base des n-grammes et plus précisément les unigrames et les bigrammes pour la représentation des documents en catégorisation automatique de textes arabes.
De nombreux travaux ont montré l'efficacité des n-grammes pour les langues latines, et spécialement dans le domaine de l'identification de la parole et la recherche d'information, cependant pour la langue arabe et en particulier en catégorisation automatique de textes, ce constat n'a pas encore été prouvé, ce qui nous a motivé à tester cette technique en catégorisation automatique de textes arabes.

Organisation du mémoire

Ce mémoire est constitué de cinq chapitres principaux ainsi qu'une introduction et une conclusion générale.
En premier lieu, le premier chapitre vise à présenter les différentes facettes de la catégorisation automatique de textes, il traite les principaux modes de représentation des documents utilisés, et les principales techniques appliquées pour réduire la taille du vocabulaire pris en compte.
Puis, le deuxième chapitre expose les principaux algorithmes d'apprentissage ayant fait leurs preuves dans le domaine de la catégorisation de textes. Nous détaillons plus particulièrement les trois algorithmes les plus utilisés Naive bayes, KPPV, SVM. Pour finir, nous indiquons, les différentes méthodes existantes pour mesurer les performances des systèmes de CT.
Pour poursuivre cet état de l'art, le troisième chapitre est consacré entièrement à l'étude de la langue arabe et ses particularités linguistiques ainsi les travaux liés dans ce domaine.

Le quatrième chapitre présente notre approche pour construire un système de catégorisation automatique de textes arabes ainsi les différentes étapes de notre étude.
Le dernier chapitre est consacré pour détailler les différentes expérimentations que nous avons menées pour valider cette approche. Nous présentons en premier lieu le corpus ainsi que les outils, et les moyens utilisés pour effectuer nos expérimentations puis nous passons à la méthodologie et les résultats des expérimentations, suivis par plusieurs discussions.

Enfin, nous concluons ce mémoire en résumant les divers résultats obtenus et en présentant les perspectives de recherche de nos travaux.

Chapitre 1

Catégorisation Automatique de Textes

Sommaire

1.1 Introduction

A L'ère de la communication électronique, il est vital de disposer d'outils automatiques permettant d'accéder facilement à l'information. Rechercher un ouvrage dans un catalogue électronique, naviguer sur le réseau Internet pour trouver une citation, suivre des discussions sur des listes ou par courrier électronique sont des activités de plus en plus courantes. Pour trouver une information souhaitée, il faut être capable de déterminer si un document est, ou non, intéressant. Accéder à l'information sous forme électronique est le rôle que se fixe la recherche documentaire. Elle s'est attachée à proposer des méthodes pour classer, archiver, analyser et diffuser des informations apparaissant dans des documents. Lorsque ces documents sont des textes, recherche documentaire et traitement du langage naturel possèdent un intérêt commun : l'analyse des textes en vue d'extraire les informations pertinentes. Dans ce contexte, la catégorisation de texte (TC), définie comme le processus permettant d'associer une catégorie (ou classe) à un texte libre en fonction des informations qu'il contient, devient un élément important des systèmes de gestion de l'information.

Le TC provient de plusieurs domaines scientifiques qui n'utilisent pas toujours le même vocabulaire, en particulier pour la dénomination des différentes taches. Il est nécessaire de bien clarifier les termes que nous allons employer. La catégorisation correspond à la classification supervisée pour l'apprentissage automatique, et à la discrimination en statistique, alors que la recherche d'information emploie des termes plus proches de l'application concernée : filtrage ou routage. La classification non supervisée de l'apprentissage automatique correspond en statistique à la classification ou *clustering* qui est également le terme utilisé en recherche d'information (RI). Pour éviter toute confusion, nous employons les termes catégorisation et *clustering*.

Dans ce chapitre, nous allons décrire les différents modules qui composent un système de catégorisation automatique. Nous commencerons par une définition formelle de la tâche de catégorisation (section 1.2), puis nous précisons comment les documents textuels sont représentés avant d'être transmis aux algorithmes d'apprentissage. Enfin, nous présenterons les techniques d'évaluation des systèmes de catégorisation.

1.2 Définition de la catégorisation automatique de textes

La catégorisation de textes consiste à chercher une liaison fonctionnelle entre un ensemble de textes et un ensemble de catégories (étiquettes, classes). Cette liaison fonctionnelle, que l'on appelle également modèle de prédiction, est estimée par un apprentissage automatique (*machine learning*) [Sebastiani, 2002]. Pour cela, il est nécessaire de disposer d'un ensemble de textes préalablement étiquetés dite ensemble d'apprentissage, à partir duquel nous estimons les paramètres du modèle de prédiction le plus performant possible, c'est-à-dire le modèle qui produit le moins d'erreurs en prédiction.

Formellement :

La catégorisation de textes consiste à associer une valeur booléenne à chaque paire $(d_j, c_i) \in D * C$ tel que :

D : l'ensemble de textes.

C : l'ensemble de catégories.

La valeur V (vrai) est alors associée au couple (dj, c_i) si le texte dj appartient à la classe c_i, tandis que la valeur F (faux) lui sera associée dans le cas contraire. Le but de la catégorisation de textes est de construire un *modèle* F tel que $F : D \times C \rightarrow \{V, F\}$.
Ce dernier associe une ou plusieurs étiquettes (catégories) à un document d_j, tel que la décision donnée par cette procédure coïncide le plus possible avec la fonction Φ . La vraie fonction qui retourne pour chaque vecteur d_j une valeur c_i. $\Phi : D \times C \rightarrow \{V, F\}$.

1.3 Applications de la catégorisation de textes

Depuis les travaux de [Maron, 1961], la catégorisation de textes est utilisée dans de nombreuses applications. Parmi lesquelles on cite : l'identification de la langue [Cavnar, 1994], la désambigüisation des termes, la reconnaissance d'écrivains [Forsyth, 1999], la catégorisation de documents multimédias [Sable, 2000], et bien d'autres.

Dans cette section, nous allons présenter un cadre général qui résume la manière dont la catégorisation de textes est utilisée. Nous allons voir que la catégorisation peut être vue

comme une fin en soi, par exemple lors de l'étiquetage de documents, comme elle peut être vue comme un support pour différentes applications, où elle représente une étape dans la représentation et le traitement de l'information contenus dans les textes [Mouliner, 1996].

1.3.1 Catégorisation de textes : une fin en soi

L'indexation automatique de textes consiste à associer à chaque texte d'une collection un ou plusieurs termes parmi un ensemble prédéfini. L'objectif est de décrire le contenu de ces textes par des mots ou des phrases clés, qui font partie d'un ensemble de vocabulaire contrôlé. Dans un tel contexte, si nous regardons ce vocabulaire contrôlé comme des catégories, l'indexation de textes peut être alors vue comme une forme de catégorisation de textes [Hayes, 1990][Fuhr, 1991][Tzeras, 1993].

1.3.2 Catégorisation de textes : un support pour différentes applications

La catégorisation de textes peut être un support pour différentes applications parmi lesquelles le filtrage et le routage.

- **Le filtrage** : consiste à déterminer si un document est pertinent ou non (décision binaire) pour une catégorie donnée. Un exemple de filtrage est la détection des courriers indésirables (*spam*), pour ensuite les supprimer [Androutsopulos, 2000].

- **Le routage :** consiste à affecter un document à une ou plusieurs catégories parmi N catégories. Un exemple de routage est la diffusion sélective d'information. Lors de la réception d'un document, l'outil choisit à quelles personnes le faire parvenir en fonction de leurs centres d'intérêt. Ces centres d'intérêt correspondent à des profils individuels [Liddy, 1994].

Notons que si la sélection est faite au niveau producteur de l'information (exemple, l'agence de presse), le système doit diriger l'information au consommateur intéressé (exemple journal) [Liddy, 1994]. Dans ce cas, on parle de routage, tandis que si la sélection est faite au niveau du consommateur de l'information pour un même utilisateur on parle alors de filtrage.

1.4 Processus général de la Catégorisation Automatique

Le processus de la catégorisation intègre la construction du modèle de prédiction, qui reçoit en entrée un texte et en sortie lui associe une ou plusieurs étiquettes (catégorie). Pour identifier la classe ou la catégorie à laquelle un texte est associé, un ensemble d'étapes est habituellement suivi. Ces étapes concernent principalement la manière dont un texte est représenté, le choix de l'algorithme d'apprentissage à utiliser, et comment évaluer les résultats obtenus pour garantir une bonne généralisation du modèle appris.

1.4.1 La Représentation des documents « *document représentation* »

Trois phases principales sont à distinguer dans cette étape: la phase du prétraitement, la phase de réduction du nombre de termes, et enfin la phase de représentation.

1. 4.1.1 Le prétraitement

Le prétraitement consiste en différentes taches. Dans cette section, on va se limiter aux taches les plus importantes, et les plus communément utilisées à savoir : la tokenisation, la radicalisation (*stemming*), la lemmatisation, et la suppression des mots vides « *stop list* ».

A. La tokenisation « *tokenization* »

La tokenisation consiste à effectuer un nettoyage d'un texte [baldi, 2003], en enlevant les expressions inutiles telles que les metadata, les éléments formatés (les balises dans un document XML ou HTML), les marques de ponctuation, les non-lettres...etc. Ensuite, le texte est transformé en une liste de mots appelés *tokens*. Pour la langue arabe, deux tokeniseurs sont les plus connus : *Diab tokenizer* [Diab, 2004], et celui développé par *Buckwalter* [Buckwalter, 2002].

B. Radicalisation ou déssuffixation (*stemming*)

Le *stemming* est une technique morphologique largement utilisée pour la préparation des textes dans une recherche documentaire. Il consiste à rechercher la racine lexicale (radical) pour des mots en langue naturelle, et ceci par l'élimination des affixes (suffixes et préfixes) qui leur sont rattachés. En d'autre terme regrouper sous un même identifiant des

mots dont la racine est commune. Par exemple, les mots « chanteront », « chanteur », « chanter » sont considéré comme des mots de la même racine « chante ». En langue arabe, les mots 'آكلات', 'مأكولات', 'أكلوا' sont des flexions du mot 'اكل'.

Pour cela, des stemmers sont conçus, ils sont généralement destinés pour une langue bien spécifique sur laquelle une certaine expertise doit être élaborée. [Larkey, 2005] considère que l'utilisation d'un dictionnaire de stem (travaux de [Alkharashi, 1994] et l'analyse morphologique (travaux de [Buckwalter, 2002]) sont une autre forme de stemming.

Plusieurs algorithmes de stemming ont été proposés pour la langue anglaise. Le plus connu est Porter's stemmer [Porter, 1980]. Pour la langue arabe, les plus connus sont : AL Stem [Darwish, 2002], et Stemmerlight10 [Larkey, 2005].

C. Lemmatisation

La lemmatisation nécessite une analyse plus poussée que le stemming. Elle se fonde sur un lexique. Un lexique est un ensemble de lemmes que l'on peut assimiler globalement aux entrées d'un dictionnaire [Plantié, 2006].

L'objectif de la lemmatisation est d'associer à chaque mot une entrée dans le lexique. Or, l'analyse morphologique est insuffisante pour extraire les lemmes d'un texte car de nombreux mots de même graphie peuvent provenir de différents lemmes. Par exemple «Offense » peut être considéré comme le lemme définissant la parole ou la faute qui blesse, ou comme le verbe "offenser" conjugué. Cette ambiguïté se résout en analysant la catégorie grammaticale du mot en question dans la phrase. La lemmatisation nécessite donc de réaliser une analyse supplémentaire : l'analyse syntaxique. La lemmatisation recouvre donc deux analyses regroupées sous le terme d'analyse morphosyntaxique. Depuis la fin des années 80, les lemmatiseurs sont capables d'associer à chaque mot d'un texte son lemme, grâce à un étiqueteur morphosyntaxique (nom, verbe, adjectif, etc.) dont les taux de réussite avoisinent les 90% [Schmid, 1994]. La construction d'un lemmatiseur nécessite néanmoins un étiquetage de quelques milliers de mots.

Les lemmatiseurs développés pour la langue arabe sont généralement intégrés dans des environnements d'analyse morphologique conjointement avec des étiqueteurs syntaxiques. On cite à titre d'exemple *Sebawai* [Darwish, 2002], il est utilisé par les participants au

cours de la conférence annuelle TREC, ainsi le lemmatiseur de Tim Buckwalter [Buckwalter, 2002], et celui développé par khoja [Khoja, 1999].
[Larkey, 2006] présente une étude bien détaillée sur le stemming et la lemmatisation pour la langue arabe, ainsi les travaux les plus connus dans ce domaine.

d. Suppression des mots vides « stop list »

L'objectif de cette étape est d'éliminer tous les mots qui ne participent pas activement au sens du document. La « *stop list* » est une liste répertoriant tous les mots outils (pronoms, articles, etc.) et les mots trop fréquents pour être discriminants. D'un point de vue linguistique, les mots outils sont par définition des mots "vides » du sens. Et d'un point de vue statistique, les mots trop fréquents et de distribution uniforme qui ne sont d'aucune aide à un processus de catégorisation puisque non discriminants [Plantié, 2006]. Par sa nature, ce prétraitement s'effectue donc en amont des autres prétraitements linguistiques, et son objectif est d'éliminer toutes les unités linguistiques non discriminantes. L'inconvénient de la « *stop-list* » est qu'elle reste dépendante d'une langue donnée. La « stop-list » pose un autre problème plus fondamental. En effet, qui peut dire à priori que tel type syntaxique ou mot « outil » est vide du sens ? On risque dans certains cas d'omettre certaines unités linguistiques qui auraient permis d'apporter une aide précieuse à la tâche de la classification. En particulier, il est peut-être imprudent d'établir cette liste avant d'avoir fait le choix de la technique de classification qui peut parfois nécessiter toutes les informations disponibles [Plantié, 2006].

1.4.1.2 Réduction du nombre de termes d'indexation « *feature reduction* »

Le problème central de la catégorisation de textes est la grande dimensionnalité de l'espace de représentation. Chaque document est représenté par un ensemble de termes dont la majorité est jugée inutile pour la tâche de la classification. Pour un corpus de taille raisonnable, le nombre de termes peut être de plusieurs dizaines de milliers. Pour beaucoup d'algorithmes d'apprentissage, il faut sélectionner un sous-ensemble de ces descripteurs. Sinon, deux problèmes se posent :

- Le coût du traitement : car le nombre de termes intervient dans l'expression de la complexité de l'algorithme. Plus ce nombre est élevé, plus le volume de calcul est important.
- La faible fréquence de certains termes : on ne peut pas construire des règles fiables à partir de quelques occurrences dans l'ensemble d'apprentissage.

A cet effet, il est nécessaire d'utiliser des méthodes statistiques pour choisir les mots utiles pour discriminer entre documents pertinents et documents non pertinents, ou plus généralement entre les classes de documents.

Les techniques utilisées pour la réduction de dimension sont issues de la théorie de l'information et de l'algèbre linéaire. [Sebastiani, 2002] classe ses techniques de deux façons : selon quelle agisse localement ou globalement, et selon la nature des résultats de la sélection : s'agit il dune sélection de termes, ou d'une extraction de termes.
Pour la sélection des termes (*feature sélection*), un score est associé à chaque terme (attribut) en fonction d'un algorithme chargé de déterminer son degré de pertinence pour un document donné. Les termes ayant les scores les plus faibles sont éliminés. Tandis que pour l'extraction des termes (*feature extraction*), un ensemble de nouveaux attributs extérieurs au document sont générés de manière à représenter ce document dans un espace indépendant, dont le nombre d'attributs est plus restreint.

A. Sélection locale des termes « *local feature selection* »

Il s'agit de proposer pour chaque catégorie C_i un ensemble de termes T_i' dont la cardinalité est nettement inférieure à la cardinalité de l'ensemble initial [Apté, 1998][Lewis, 1996][Schütze, 1995][Ng, 1997][Sable, 2000]. Avec cette technique, chaque catégorie C_i possède son propre ensemble de termes, et chaque document d_j est représenté par un ensemble de vecteurs d_j différents selon la catégorie.

B. Sélection globale des termes « *global feature selection* »

Dans ce cas, le nouvel ensemble de termes T' est choisi en fonction de toutes les catégories. Ainsi, chaque document d_j est représenté par un seul vecteur d_j quelque soit la catégorie [Yang, 1997] [Mladeni'c, 1998] [Caropreso, 2001].

[Bong, 2005] explique en détail la différence entre la sélection locale et globale des termes et quels termes à choisir, il a prouvé que la sélection locale dépasse la sélection globale des termes, et qu'elle conduit à de meilleurs résultats par rapport à cette dernière.

C. Techniques de Sélection des termes

Principe

La méthode idéale consisterait à tester tous les sous-ensembles possibles de descripteurs afin de conserver l'ensemble donnant les meilleurs résultats sur une base de test. Une telle solution n'est évidemment pas possible, car si l'on considère un ensemble de p candidats, le nombre d'ensembles à tester s'élève à 2^p. Donc si l'ensemble initial comporte cent descripteurs, le nombre de combinaisons s'élève à environ 10^{30} ce qui est évidemment beaucoup trop grand pour être réalisable.

Parmi les méthodes testées ici, deux approches différentes sont utilisées ; toutes les deux tiennent compte de la tâche que l'on cherche à accomplir : différencier les textes pertinents des textes non pertinents.

La première approche consiste à calculer un score pour chaque descripteur, indépendamment des autres, en s'appuyant sur les statistiques d'apparition et d'absence du descripteur en fonction de la classe à laquelle appartiennent les textes. Les descripteurs sont ensuite classés selon ce score, les descripteurs en tête de liste étant les plus discriminants pour distinguer les textes pertinents des textes non pertinents. Les méthodes de l'information mutuelle et du chi-2 exposées ci-après reposent sur ce principe.

La deuxième approche est constructive : elle construit itérativement un modèle, en partant d'un ensemble vide et en ajoutant successivement de nouveaux descripteurs en tenant compte des descripteurs déjà sélectionnés. Cette construction est faite en utilisant l'algorithme d'orthogonalisation de Gram-Schmidt.

Pour toutes ces méthodes, le résultat se présente sous la même forme : il s'agit d'une liste de mots ordonnés du plus discriminant au moins discriminant.

➢ **La fréquence documentaire DF (*document frequency*)**

C'est la technique la plus simple, elle consiste à calculer la fréquence de chaque terme dans le corpus, c'est-à-dire le nombre des documents dans lesquels le terme apparaît. Les termes dont la fréquence ne dépasse pas un certain seuil fixé (en général plus de 3) seront éliminés. Cette technique a été utilisée dans [Apté, 1998].

➢ **Le gain d'information IG (*information gain)***

On mesure en quelque sorte le pouvoir de discrimination d'un mot. Le nombre de bits d'information obtenue pour la prédiction de la catégorie en sachant la présence ou l'absence d'un mot. Cette méthode est souvent mise en pratique dans les arbres de décisions, pour choisir l'attribut qui va le mieux diviser l'ensemble des instances en deux groupes homogènes.

Formellement :

$$G(t) = -\sum_{i=1}^{m} p(c_i)\log p(c_i) + p(t)\sum_{i=1}^{m} p(c_i/t)\log p(c_i/t) + p(t')\sum_{i=1}^{m} p(c_i/t')\log p(c_i/t') \quad (1.1)$$

p : signifie la proportion de documents ayant la caractéristique entre parenthèse.

c_i : signifie qu'un document fait partie de la catégorie ci.

t : signifie qu'un document possède le terme t.

t' : signifie qu'un document ne possède pas le terme t.

m : est le nombre de catégorie

➢ **L'information mutuelle MI (*mutual information)***

Cette façon d'évaluer la qualité d'un mot dans la prédiction de la classe d'un document est basée sur le nombre de fois qu'un mot apparaisse dans une certaine catégorie. Plus un mot va apparaitre dans une catégorie, plus l'information mutuelle du mot et de la catégorie va être jugée élevée. Plus un mot va apparaitre en dehors de la catégorie (et plus une catégorie va apparaitre sans le mot), moins l'information mutuelle va être jugée élevée. Il faut ensuite faire une moyenne des scores du mot jumelé à chacune des catégories.

Formellement :

On procède comme suit : on construit 2 tables de contingence pour chaque terme t avec une classe c. A est le nombre de fois où t et c co-occurrent. B Le nombre de fois ou t apparaît sans c. C le nombre de fois où c apparait sans le terme t , et N le nombre de documents.

L'information mutuelle entre **t** et **c** notée **I** est estimée comme suit :

$$I(t,c) = \log\left(\frac{A * N}{(A + C) * (A + B)}\right) \qquad (1.2)$$

➢ **La méthode du *chi* (x^2) :**

Mesure statistique bien connue, elle s'adapte bien à la sélection d'attributs, car elle évalue le manque d'indépendance entre un descripteur t et un thème T. Cette mesure a été utilisée pour la sélection des descripteurs dans [Schütze, 1995][Wiener, 1995]. Le calcul nécessite de construire pour chaque descripteur t du corpus un tableau de contingences comme suit :

	Descripteur t présent	Descripteur t absent	
Thème T présent	a	c	T1 = a+c
Thème T absent	b	d	T0= b+d
	D_1=a+b	D_2 =c+d	N=a+ b+c+d

Tableau 1-1:Tableau de contingence d'un descripteur

On définit :

$$x^2(t,T) = \frac{N(ad - cb)^2}{(a + c)(b + d)(a + b)(c + d)} \qquad (1.3)$$

Si un descripteur t et le thème T sont totalement indépendants, alors t apparait avec la même fréquence dans le sous-ensemble des textes pertinents et dans le sous-ensemble des textes non pertinents, ce qui se traduit par $(ad = cb)$, et dans ce cas la valeur de $x^2(t,T)$ est nulle.

A l'inverse, si le descripteur t apparait systématiquement dans l'ensemble des textes pertinents et jamais dans l'ensemble des textes non pertinents, on aura $c = b = 0$ et $x^2(t,T) = N$ qui est sa valeur maximale. Cette valeur est également atteinte si un descripteur apparait systématiquement dans l'ensemble des textes non pertinents et jamais dans l'ensemble des textes pertinents. Entre ces deux valeurs extrêmes, plus la valeur de $x^2(t,T)$ est grande, plus t et T sont liés. Les descripteurs du corpus sont donc classés par ordre décroissant de la valeur $x^2(t,T)$ .Les termes les plus discriminants figurant en tête de liste.

- **La force du terme TS *(term strength)***

Il s'agit d'une méthode plutôt différente des autres. Elle se propose d'estimer l'importance d'un terme en fonction de sa propension à apparaitre dans des documents semblables. Une première étape consiste à former des paires de documents dont la similarité cosinusoïdale est supérieure à un certain seuil. La force d'un terme est ensuite calculée à l'aide de la probabilité conditionnelle qu'il apparaisse dans le deuxième document d'une paire, sachant qu'il apparait dans le premier.

Formellement :

$$s(t) = p(t \in y / t \in x) \qquad (1.4)$$

1.4.1.3 La phase de représentation

Cette étape consiste généralement en la représentation de chaque document par un vecteur de termes pondérés. Selon ce principe, une collection de textes est représentée par une matrice ou tableau croisé « individu-variables » tel que :

- l'individu est un document textuel d_j étiqueté lors de la phase d'apprentissage ou à classer dans la phase de prédiction.
- Les variables sont les descripteurs ou les termes d'indexation t_k extraits des données textuelles.
- le contenu du tableau (les éléments w_{jk}) représente le poids du terme k dans le document j.

Donc à ce niveau, il est nécessaire de bien préciser à quoi correspond un terme, et comment calculer son poids.

1.4.1.3.1 Choix de termes

Le principal enjeu de la catégorisation de textes par rapport à un processus d'apprentissage classique réside dans la recherche de termes les plus pertinents pour le problème à traiter [Aas, 1999]. Différentes approches sont proposées pour choisir les termes les plus représentatifs de document, c'est-à-dire celles qui permet de bien décrire le contenu de document. Dans cette section nous allons présenter les approches les plus connus ainsi les avantages et limites de chacune d'elles.

A. La représentation sac de mots (*bag of words*)

C'est la représentation de textes la plus simple. Selon cette représentation, le contenu d'un document est décrit à l'aide des mots. Les mots ont l'avantage de posséder un sens explicite. Cependant, plusieurs problèmes se posent. Il faut tout d'abord définir ce qu'est un mot pour pouvoir le traiter automatiquement. On peut le considérer comme une suite de caractères appartenant à un dictionnaire, ou bien, de façon plus pratique, comme étant une séquence de caractères non délimiteurs encadrés par des caractères délimiteurs (caractères de ponctuation) [Gilli, 1998]. Il faut alors gérer les sigles, ainsi que les mots composés. Ceci nécessite un prétraitement linguistique. On peut choisir de conserver les majuscules pour aider, par exemple, à la reconnaissance de noms propres. Les composantes de vecteur sont une fonction de l'occurrence des mots dans le texte. Cette représentation des textes exclut toute analyse grammaticale et toute notion de distance entre les mots, c'est pourquoi

cette représentation est appelée sac de mots. Cette technique de représentation a été utilisée par plusieurs auteurs comme [Apté, 1998][Dumais, 1998][Aas, 1999].

B. Représentation des textes avec des racines lexicales ou des lemmes

Dans le modèle précédent (représentation *sac de mots*), chaque flexion d'un mot est considérée comme un descripteur différent, et donc une dimension en plus. Ainsi, les différentes formes d'un verbe constituent autant de mots ; par exemple les mots خرج, استخراج, خروج sont considérés comme des descripteurs différents alors qu'il s'agit de la même racine خرج. Les techniques de déssufuxation (*stemming*) cherchent à résoudre cette difficulté en remplaçant les flexions par leurs stems ou par leurs lemmes.

C. Représentation des textes par des phrases

Certaines approches utilisent non pas des mots, mais des groupes de mots ou des phrases, comme éléments représentatifs du sens. L'intérêt d'utiliser une phrase ou un groupe de mots est double. Tout d'abord, ce type d'unités linguistiques possède une unité de sens plus complète qu'un simple mot de par la « contextualisation » des mots [Lewis, 1992]. En effet, l'association d'un ensemble de mots, même dans le désordre, offre davantage d'information sur le champ sémantique dans lequel on se trouve. De plus, ce type d'unités linguistiques définit une relation d'ordre entre les mots. Un des avantages de cette représentation est qu'elle ouvre la porte à la gestion des cooccurrences de mots.

En pratique, les expérimentations avec ce type de représentation ne sont pour l'instant pas très concluantes. En effet, si la phrase est une unité linguistique à forte valeur sémantique ajoutée, la fréquence d'apparition de groupes de mots ne permet pas d'offrir des statistiques fiables. Le grand nombre de combinaisons entre les mots engendre des fréquences trop faibles pour être exploitable [Lewis, 1992].

Certains auteurs comme [Caropreso, 2001] ont utilisé la notion de phrase statistique, celle-ci est un ensemble de mots contigus, mais pas nécessairement ordonnés, qui apparaissent ensemble, mais qui ne respectent pas les règles grammaticales. Des affinements de ces méthodes peuvent apparaitre dans la notion de syntagmes nominaux et verbaux

Exemple de syntagme nominal : الإعلام الآلي

Exemple de syntagme verbal : دخل الولد

D. Représentation des textes basée sur la technique des n-grammes

Un n-grammes se définit comme étant une séquence de *n* caractères consécutifs. Cependant, il existe quelques variantes dans la littérature. En effet dans [Caropreso, 2001], l'auteur définit le n-gramme comme étant une séquence de *n* « mots désuffixés » et dans [Cavnar, 1994] l'auteur définit le n-gramme comme étant une séquence de *n* caractères non obligatoirement consécutifs. Nous nous contenterons ici de la notion de *n* caractères consécutifs. Ainsi, le mot : « إسلام » est composé des n-grammes suivants :

- bi-grammes : « اس », « سل », « لا », «ام ».
- tri-grammes : « اسل », « _سلا », « لام », « ام_ ».

Pour un document quelconque, l'ensemble des *n*-grammes qu'on peut générer est la famille de photos qu'on obtient en déplaçant une fenêtre de N cases sur le corps du texte [Jalam, 2003]. Ce déplacement se fait par étapes, à chaque étape une prise de photo se fait. L'ensemble des photos qu'on obtient constitue l'ensemble de tous les n-grammes du document.

La notion des n-grammes a été introduite par Shannon [Shannon, 1948]. Depuis cette date les n-grammes sont utilisés dans plusieurs domaines comme l'identification de la langue, la recherche documentaire.

Les avantages liés à l'utilisation des n-grammes sont multiples : d'abord, elles opèrent indépendamment de la langue, contrairement aux systèmes basés sur les mots dans lesquels il faut utiliser des dictionnaires spécifiques (féminin masculin, singulier pluriel, conjugaisons, etc.) pour chaque langue de plus, avec les n-grammes, on n'a pas besoin d'une segmentation préalable du texte en mots. Ceci est intéressant pour le traitement des langues dans lesquelles les frontières entre mots ne sont pas fortement marquées, comme le Chinois, l'Arabe [Maning, 1999] [Biskri, 2001]. Aussi les n-grammes capturent automatiquement les racines de mots les plus fréquents, donc on n'a pas besoin de l'étape de recherche des racines lexicales ni de lemmatisation. Enfin, les n-grammes sont tolérants aux fautes d'orthographe et aux déformations causées lors de l'utilisation des lecteurs

optiques. Plusieurs recherches montrent que des systèmes de recherches documentaires basés sur les n-grammes ont gardé leurs performances malgré des taux de déformations de 30 %, situation dans laquelle aucun système basé sur les mots ne peut fonctionner correctement [Jalam, 2003].

1.4.1.3.2 Codage des termes

Une fois la liste des termes déterminée, il reste à attribuer une pondération à chacun d'eux. Le choix le plus simple est d'utiliser une pondération binaire : 1 si le terme est présent dans le document, 0 dans le cas contraire. Beaucoup de travaux en RI dans les années 80-90 ont eu pour objectif d'améliorer les pondérations pour que les termes les plus significatifs aient un poids relativement plus élevé. Le processus a été essentiellement empirique à partir de grande compagne d'évaluation où de nombreuses pondérations étaient comparées les unes aux autres (voir par exemple les conférences TREC). Lorsque les algorithmes d'apprentissage automatique ont commencé à être utilisé en catégorisation automatique, les documents étaient souvent représentés avec le modèle binaire (codage binaire des termes) parce qu'on supposait que les classifieurs seraient capable de trier eux-mêmes les termes importants ou par ce que les algorithmes nécessitaient une représentation symbolique binaire [Vinot, 2004]. Mais comme pour la RI, l'utilisation de poids calculés à partir de formules spécifiques au domaine textuel permet généralement d'améliorer les performances.

Il existe essentiellement deux façons pour calculer le poids des termes. La première est issue du modèle vectoriel de Salton développé au sein de logiciel SMART [Salton, 1975] est fondé sur des heuristiques qui se sont raffinées au cours des dernières décennies, et l'autre est issue du modèle probabiliste.

A. Codage des termes dans le modèle vectoriel

Dans ce modèle, les termes sont implicitement considérés comme indépendants (le calcul de poids d'un terme ne dépend pas de celui d'un autre terme, bien que linguistiquement erroné). Cette hypothèse ne semble pas être réellement un obstacle à l'utilisation du modèle. [Domingo, 1996] a montré que les performances du classifieur probabiliste Naïve Bayes (voir le chapitre suivant) ne sont pas affectés par la violation de

l'hypothèse d'indépendance des termes. Nous pensons que ceci est vrai pour beaucoup d'autres algorithmes qui font également cette hypothèse d'indépendance, en particulier tout les classifieurs qui utilisent le modèle vectoriel pour représenter les documents.

Il existe différentes méthodes pour calculer le poids de termes dans ce modèle, en se basant sur les deux observations suivantes :

- Plus le terme est fréquent dans un document plus il est en rapport avec le sujet de ce document.
- Plus le terme est fréquent dans une collection, moins il sera utilisé comme discriminant entre documents.

➢ **Le codage TF*IDF**

TF*IDF a été introduit dans le cadre du modèle vectoriel présenté ci-dessus. Il est basé sur la loi de zipf [Denoyer, 2004] qui stipule que les termes les plus informatifs d'un corpus de document sont ni les mots qui apparaissent le plus dans le corpus, car ceux-ci sont pour la plupart des mots outils (de type article, mots de liaison, etc), ni les mots les moins fréquents du corpus, car il peuvent être par exemple issus de fautes d'orthographe ou de l'utilisation d'un vocabulaire trop spécifique à un unique ou à quelque document du corpus. Par contre, un mot qui apparait beaucoup dans un document possède certainement une information forte sur la sémantique de document.

Donc la pondération TF*IDF se base sur les deux notions :

- La fréquence de terme TF (*terme frequency*) qui prend en compte le nombre d'occurrence de terme dans le document.
- L'inverse de sa fréquence documentaire IDF (*inverse document frequency*), qui prend en compte le nombre d'occurrence de terme dans le corpus.

Ces deux notions sont combinées multiplicativement de façon à attribuer un poids d'autant plus fort que le terme apparait souvent dans le document et rarement dans le corpus.

Formellement :

Le pois assigné à un terme est calculé comme suit :

$$TF * IDF(t,d) = tf(t,d) * idf(d) = tf(t,d) * \log\left(\frac{|D|}{df}\right) \quad (1.5)$$

tf =le nombre d'occurrence du terme dans le document.

$|D|$ =le nombre de documents dans le corpus « la taille de corpus ».

df = le nombre de documents dans le corpus qui contiennent le terme t.

L'exemple suivant illustre le principe de ce type de codage.

Exemple :

Soit à représenter le corpus C constitué de documents d_1, d_2 , d_3 comme suit :

d_1 = قل هو الله احد, الله الصمد

d_2 = هو الله الذي لا اله إلا هو عالم الغيب و الشهادة

d_3 = قل ادعوا الله أو ادعوا الرحمان

En utilisant la représentation sac de mots et le codage TF*IDF, Le corpus C est représenté par la matrice individu-terme comme suit :

	قل	الله	احد	الصمد	اله	عالم	الغيب	الشهادة	الرحمان	الرحيم	ادعوا
d_1	log(3/2)	2*log(1) =0	Log(3)	Log(3)	0	0	0	0	0	0	0
d_2	0	Log(1)=0	0	0	Log(3)	Log(3)	Log(3)	Log(3)	Log(3/2)	Log(3)	0
d_3	Log(3/2)	Log(1)=0	0	0	0	0	0	0	Log(3/2)	0	2* Log(3)

Tableau 1-2:Exemple illustratif du codage TF*IDF

On remarque que les termes qui figurent dans tout le corpus, et les termes rares qui apparaissent dans un seul document ont un poids nul.

➢ **Le codage TFC**

Le codage TF*IDF ne tient pas en compte la longueur de documents ; il favorise les documents les plus longs. Plus un document est long et contient par conséquence beaucoup d'occurrences, plus les poids augmentent. Pour cette raison, le codage TFC est apparu. Il tend à éviter ce problème par une normalisation en cosinus.

Formellement :

Le poids assigné à un terme t est calculé comme suit :

$$TFC(t_k, d_j) = \frac{TF * IDF(t_k, d_j)}{\sqrt{\sum_{s=1}^{k} TF * IDF(t_s d_j)^2}} \qquad (I.6)$$

D'autres codages sont également utilisés, comme par exemple le codage LTC [Bucklay, 1994], ou encore le codage à base d'entropie [Dumais, 1998][Aas, 1999].

B. Codage des termes dans le modèle probabiliste

Dans le modèle probabiliste, on considère que les documents sont générés par tirage aléatoire des différents termes qui les composent. Le modèle de génération décrit le processus sous-jacent qui génère les termes puis les documents. Ce processus n'est pas accessible aux observateurs. Seuls les résultats le sont. C'est-à-dire, la liste de documents du corpus qui ont été générés par ce processus l'est. Les paramètres du processus, c'est-à-dire les valeurs de probabilités de chaque tirage sont estimées à partir des occurrences trouvées sur les documents du corpus. Cela revient en général à estimer la probabilité $p(t_k/C_j)$, la probabilité d'apparition de terme t_k sachant que le document appartient à la classe C_j. [Vinot, 2004][Lewis, 1998][Jones, 2000] présentent le modèle probabiliste dans le cadre de la recherche d'information.

1.4.2. Choix du classifieur

Une fois les données textuelles transformées en données statistiques, il reste à choisir le type de classifieur à utiliser pour résoudre le problème de la catégorisation. Dans cette

optique, plusieurs classifieurs mis au point pour des problèmes quelconques en apprentissage automatique ont été adaptés et appliqués dans notre domaine de recherche avec plus ou moins de succès.

Parmi les classifieurs les plus souvent utilisées figurent la régression logistique [Hull, 1994], les réseaux de neurones [Schütze, 1995], les k plus proches voisins [Yang, 1994], les arbres de décision [Quinlan, 1996] [Apté, 1998], les réseaux bayesiens [Lewis, 1992][McCallum, 1998], les modèles de Markov Cachés [Zaragoza, 1999], les machines à vecteurs supports [Dumais, 1998][Joachims, 1998] et plus récemment les méthodes basées sur la méthode dite de *boosting* [Schapire, 1998][Iyer, 2000].

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

Les travaux semblent se concentrer aujourd'hui principalement autour de trois algorithmes: les k plus proche voisin, les machines à vecteur support, et le boosting sur lesquels nous allons donner plus de détails dans le chapitre suivant.

1.4.3 Evaluation des performances des classifieurs textuels

Dans le domaine de la RI et de l'apprentissage automatique, la validation des méthodes est généralement empirique. Il existe plusieurs grandes compagnes d'évaluation internationales, chacune se focalise sur une tache particulière dont le but est de comparer objectivement différents systèmes sur les mêmes données et avec les mêmes mesures de performance. Malgré ces compagnes, il reste très difficile d'affirmer la supériorité d'un classifieur sur un autre, car les résultats sont très dépendants des corpus utilisés, de la tâche à évaluer, des mesures de performance et même des implémentations [Salzberg, 1997].

Dans le contexte des mesures de performance, il faut noter qu'il y a deux critères importants à signaler :

- **L'efficacité :** est le critère technique qui mesure les temps de calcul et la taille mémoire nécessaire.
- **La justesse des prédictions :** mesure si la catégorisation effectuée est correcte.

Pour mesurer l'efficacité, il est possible d'utiliser la théorie de la complexité [Vinot, 2004]. Cette dernière permet de partager les différents algorithmes en plusieurs grandes classes (polynomiaux, exponentiels), mais les analyses théoriques des algorithmes sont souvent difficilement comparables. Il est aussi possible d'utiliser des mesures empiriques pour comparer les temps de calcul. Malheureusement, ces mesures dépendent fortement des conditions d'expérimentations et sont difficilement réutilisables.

En revanche, de très nombreux travaux cherchent à comparer les performances de différents classifieurs en termes de justesse des prédictions. Ces comparaisons sont essentiellement empiriques à partir de mesures effectuées sur des corpus de test. Il faut donc découper un corpus en deux parties, l'un servant à l'apprentissage et l'autre aux tests. Suivant le découpage effectué, les résultats peuvent être significativement différents.
Ce problème a été largement étudié en statistique à travers les théories de l'échantillonnage. Plusieurs méthodes ont été proposées pour le résoudre [Dietterich, 1995] la plus connue est la technique de validation croisée qu'on va détailler au niveau de chapitre suivant. Il est, de plus nécessaire de faire les tests sur de nombreux corpus différents afin que les résultats soient généralisables sur d'autres corpus non testés. Il existe de nombreuses mesures pour rendre compte des résultats. Dans le chapitre suivant, nous allons donner une définition formelle de ces mesures.

1.5 Difficultés particulières de la catégorisation

L'utilisation des méthodes d'apprentissage automatique pour traiter les données textuelles est plus difficile que le traitement de données numériques. Le langage naturel par opposition aux langages informatiques n'est pas univoque.
"Un langage univoque est un langage dans lequel chaque mot ou expression a un seul sens, une seule interprétation possible et il n'existe qu'une seule manière d'exprimer un concept donné" [Lefèvre, 2000].
Le langage naturel est équivoque, il y a plusieurs façons d'exprimer la même idée soit en utilisant des termes différents, et dans ce cas on parle de synonymie, soit en utilisant les mêmes termes pour dire des choses différentes. Dans ce cas on parle du problème de

polysomie. En plus, ce qui est exprimé possède souvent plusieurs interprétations. C'est le problème de l'ambiguïté.

Ajoutons à ces particularités, le grand nombre de descripteurs, ou la grande dimensionnalité de l'espace de représentation, qui peut dégrader l'efficacité des algorithmes d'apprentissage, car le nombre de descripteurs est un paramètre de la complexité des algorithmes.

Enfin, nous notons la subjectivité de la décision prise par les experts qui déterminent la catégorie dans laquelle classer un document (Un autre expert peut prendre une décision différente). Une classification opérée pour un corpus, peut devenir caduque lorsque le nombre de documents augmente. On doit revoir les classes et les critères de classification.

1.6 Recherche d'information et catégorisation

La RI et la CT partagent un grand nombre de points de vue et de méthodes. Dans les deux cas, il s'agit de sélectionner les documents qui comportent une information pertinente pour l'utilisateur. Il est même possible de représenter le problème de RI sous la forme d'une tâche de CT. A chaque requête correspond un problème de CT constitué de deux classes : la catégorie des documents pertinents et celle des documents non pertinents. Il s'agit alors de classer l'ensemble des documents dans une de ces deux classes.

La différence principale entre les deux approches réside dans le type d'informations recherché par l'utilisateur. La RI est utilisée pour des demandes ponctuelles alors que la CT modélise un intérêt à long terme. Cette différence implique des variations sur le type et la quantité d'information que l'utilisateur peut fournir au système dans les deux modes de recherche. En RI, l'utilisateur n'a pas le temps de décrire très précisément sa requête, il se contente de fournir quelque mots clés. En CT, le besoin de catégoriser peut perdurer plusieurs mois, le système a donc la possibilité d'utiliser toute l'information contenue dans l'ensemble des documents déjà catégorisés. La quantité d'information est bien plus importante. Cette différence s'atténue lorsque le système de RI interagit avec l'utilisateur pour préciser les requêtes (feedback) dans ce cas l'information utilisée comprend la requête initiale ainsi que les documents pour lesquels l'utilisateur a fourni un jugement. Le feedback utilise donc également la notion de documents catégorisés, ce qui rend son fonctionnement très proche de celui de la CT. Les algorithmes développés pour l'exploiter

(Rocchio et Naive Bayes) ont d'ailleurs été appliqués avec succès en CT. Néanmoins, les deux approches ne sont pas toujours comparables en raison de la différence du nombre de documents dans le corpus d'apprentissage : lors d'une recherche avec feedback, l'utilisateur fournit un jugement pour quelques documents (de un à quelques dizaines), alors qu'en CT le corpus peut en contenir plusieurs milliers. C'est pourquoi les algorithmes de CT peuvent utiliser une représentation plus complexe sans risquer le sur apprentissage des paramètres, ce que ne peuvent pas faire les algorithmes de feedback [Vinot, 2004].

1.7 Conclusion

Ce chapitre a permis de brosser un portrait global de la catégorisation automatique de textes. Tout d'abord, une description du problème est présentée, ainsi que les applications possibles. Par la suite, nous avons présenté le processus général de la catégorisation automatique, ou nous avons fait un aperçu sur les techniques de sélection d'attributs. Celles-ci visent à réduire la taille du vocabulaire à traiter pour que les algorithmes évoluent dans un espace vectoriel de dimension raisonnable. Puis les différentes stratégies de représentation des documents sont exposées. Le choix judicieux d'un mode de représentation des données est nécessaire, comme pour toute application de l'apprentissage automatique. Enfin, nous avons exposé quelques difficultés de la catégorisation automatique de textes.

Chapitre 2

Classifieurs Utilisés dans la CT

Sommaire

2.1 Introduction

Comme il a été précisé, c'est principalement par apprentissage automatique que l'on tente de résoudre le problème de la catégorisation automatique de textes. Dans cette optique, plusieurs algorithmes mis au point pour des problèmes quelconques en apprentissage automatique ont été adaptés et appliqués dans ce domaine de recherche. L'objectif des algorithmes d'apprentissage et plus précisément les algorithmes d'apprentissage supervisés est d'ajuster un modèle qui explique le lien entre des documents d'entrée et les classes de sortie. En catégorisation automatique de textes, on fournit à la machine des exemples sous la forme (document, classe). Cette méthode de raisonnement est appelée inductive, car on induit de la connaissance (le modèle) à partir des données d'entrée et de sorties. Grâce à ce modèle, on peut prédire les classes de nouveaux documents.

Le nombre très important des conférences et de publications relatives à la catégorisation automatique de textes rend impossible une présentation exhaustive des algorithmes d'apprentissage. Dans ce chapitre, nous nous contentons de présenter les algorithmes les plus utilisés dans la littérature, d'abord d'une façon générale puis en détaillant trois d'entre eux. Dans un deuxième temps, on abordera les critères d'évaluation des classifieurs, qui sont essentiels pour mesurer leurs performances, et surtout pour les comparer entre eux.

2.2 Les Classifieurs

Différents types de classifieurs ont été mis au point, toujours dans le but d'atteindre un degré maximal de précision et d'efficacité, chacun ayant ses avantages et ses inconvénients. Ils partagent toutefois, des caractéristiques communes. La construction de la majorité d'entre eux comporte deux principales étapes : d'abord, la définition d'une fonction qui associe à un document une valeur entre 0 et 1 représentant son degré d'appartenance à la catégorie. Cette fonction est appelée CSV pour *«Categorization Status Value»* et prend différentes formes selon le type de classifieur. La deuxième étape, mais non la moindre, est de déterminer un seuil qui va servir lors de la prise de décision finale, à

savoir si oui ou non un document va être accepté ou rejeté de la catégorie. Si la fonction CSV retourne une valeur supérieure au seuil pour un document, alors on décide de l'associer à la catégorie. Plusieurs méthodes de détermination du seuil sont possibles et le choix de l'une d'entre elles peut influencer significativement les performances d'un classifieur.

Parmi la panoplie de classifieurs existants, on peut faire des regroupements et distinguer de grandes familles. Par exemple, on peut discerner les classifieurs probabilistes qui utilisent l'ensemble d'entrainement, c'est-à-dire les textes déjà classés, pour estimer les paramètres de la distribution de probabilité des mots par rapport aux catégories. C'est dans cette famille qu'on retrouve entre autres le classifieur Naïve Bayes. On trouve aussi des classifieurs se basant sur un profil, les classifieurs linéaires. Dans ce contexte, le profil est un vecteur de termes pondérés construit pour chaque catégorie, dans le but de les représenter d'une façon générale. Ce vecteur est bien sûr construit à l'aide des données d'entrainement. Quand un nouveau texte doit être classé, il est alors comparé à ce vecteur *«type»*. Un avantage de cette approche est qu'elle produit un classifieur compréhensible par un humain, dans le sens où le profil de la catégorie peut être interprété assez facilement. Par contre, l'inconvénient principal de tous les classifieurs linéaires est que l'espace est divisé seulement en deux portions, ce qui peut être restrictif, car tous les problèmes ne sont pas nécessairement linéairement séparables. Parmi les nombreux membres de cette famille, nous retrouvons Rocchio, Widrow-Hoff [Lewis, 1996]. Les machines à vecteurs supports s'apparentent aux classifieurs linéaires, dans le sens où elles tentent de séparer l'espace en deux, mais certaines manipulations mathématiques les rendent adaptables à des problèmes non linéaires. Il y a aussi une famille de classifieurs qui se basent sur l'exemple. On parle alors d'apprentissage à base d'instances. Les nouveaux textes à classer sont comparés directement aux documents de l'ensemble d'entrainement. L'algorithme des k- plus proches voisins est sans doute le plus connu de cette famille.

Dans les sections suivantes, trois classifieurs seront exposés plus en détail. D'abord, le classifieur Naive Bayes qui est souvent utilisé comme point de référence à cause de sa simplicité et du fait qu'il est connu depuis assez longtemps. Puis, les k-plus proches voisins

et les machines à vecteurs support, qui représentent vraisemblablement à l'heure actuelle les deux meilleurs choix en catégorisation automatique de textes .

2.2.1 Naïve Bayes

Naïve Bayes est le représentant le plus populaire des classifieurs probabilistes. Le principe qui régit et donne son nom à l'algorithme est très simple. Il indique simplement que les différents attributs (dans le cas de texte, les différents termes présents dans le document) sont considéré comme indépendants. C'est la même hypothèse que pour le modèle vectoriel, mais cette fois exprimée explicitement dans le cadre de la théorie probabiliste. Cet algorithme dont le modèle d'apprentissage est très général est utilisé dans de nombreux autres domaines que le texte.

La classification d'un document s'obtient par estimation de $p(c_j/d_i)$ la probabilité connaissant le document d_i que celui-ci fasse partie de la classe c_j. Le choix optimal est de mettre l'exemple dans la classe qui a la plus forte probabilité a posteriori. Comme cette probabilité n'est pas connue, il faut l'estimer à partir des données contenues dans le corpus d'apprentissage.

La formule de Bayes permet d'inverser la probabilité conditionnelle :

$$p\left(\frac{c_j}{d_i}\right) = \frac{p(c_j) * p(d_i \backslash c_j)}{p(d_i)} \qquad (2.1)$$

Comme le but est de discriminer les différentes classes (il suffit d'ordonner les $p(c_j/d_i)$ pour toutes les classes, il est inutile d'obtenir la valeur exacte), on peut alors supprimer le terme $p(d_i)$ qui est le même pour toutes les classes. $p(c_j)$ est la probabilité a priori qui est le plus couramment estimée par le pourcentage d'exemples appartenant à la classe c_j dans le corpus d'apprentissage.

$$p(c_j) = \frac{N(c_j)}{N} \qquad (2.2)$$

Le calcul de $p(d_i/c_j)$ dépend du modèle de génération des exemples.

Dans le cas du texte les plus populaires sont le modèle multivarié de Bernoulli et le modèle multinomial [Lewis, 1998] [McCallum, 1998].

➢ **Le modèle multivarié de Bernoulli**

Dans ce modèle, un document est un vecteur binaire de la taille du vocabulaire. Seule la présence (resp. absence) des termes est utilisée. Leur nombre d'occurrence dans le document n'a pas d'incidence. Le document se présente comme le résultat du tirage de N variables aléatoires indépendantes : t_1, $t2$.... t_N.

$p(t/c_j)$ se simplifie alors en un produit de probabilités d'apparitions de chaque terme.

$$p(d_i/c_j) = \prod p\,(t/c_j) \qquad (2.3)$$

Il est possible d'estimer $p(t/c_j)$ à partir des exemples d'apprentissage. On utilise généralement l'estimateur du maximum de vraisemblance avec le lissage de Laplace pour éviter les probabilités nulles. On parvient alors dans ce cas à :

$$p(t_k/c_j) = \frac{1 + \sum_{d_i} l_{ik}}{2 + |c_j|} \qquad (2.4)$$

Avec $l_{ik} = 1$ si t_k est présent dans le document d_i, 0 sinon.

➢ **Le modèle multinomial**

Dans le modèle précédent, il n'est pas possible d'utiliser les fréquences des termes dans les documents. Pour prendre en compte cette information supplémentaire, un autre modèle plus complexe a été proposé. Un document est une séquence de mots, chacun étant tiré aléatoirement parmi l'ensemble des mots du vocabulaire. Un document est donc généré par une distribution multinomiale des mots avec autant de tirage que des mots dans le document. L'hypothèse d'indépendance des termes reste nécessaire.

Formellement :

$$p(d_i/c_j) = p(|d_i|)|d_i|!\prod_{a_i d_i}\frac{p(a_k/c_j)^{Occ(i,k)}}{Occ(i,k)!} \qquad (2.5)$$

Les probabilités sont une nouvelle fois estimées à partir des occurrences des exemples du corpus avec l'estimateur du maximum de vraisemblance et du lissage de Laplace :

$$p'(t_k/c_j)\frac{1+\sum Occ(k,i)}{|V|+\sum_{K=1}^{V}\sum_{d_i} Occ(k,i)} \qquad (2.6)$$

Quant à la classification, l'estimation $p'(d_i/c_j)$ est calculée à partir des formules (2.3) et (2.5) suivant le modèle utilisé, en remplaçant les probabilités $p(t_k/c_j)$ par leur estimateur $p'(t_k/c_j)$. La classe c_j ayant la probabilité $p'(c_j/d_i)$ la plus élevée est choisie.

2.2.2 K plus proches voisins « K-PPV »

L'algorithme des k plus proches voisins est l'un des plus vieux algorithmes de la reconnaissance des formes [Cover, 1967]. Il a été employé avec succès dans nombreux domaines et a engendré toute une famille de classifieurs connus sous le nom de classifieurs paresseux (*lazy learners*)[Aha, 1997]. Dans ces systèmes, le seul traitement effectué au cours de la phase d'apprentissage est le stockage des exemples sous une forme optimale, de façon à pouvoir les extraire ensuite rapidement. Tous les calculs sont reportés à la phase de classification, d'où le terme « paresseux ».

Lorsqu'un nouveau document à classer arrive, il est comparé aux documents d'entrainement à l'aide d'une mesure de similarité. Ses k plus proches voisins sont alors considérés. On observe leurs catégories et celle qui revient le plus parmi les voisins est assignée au document à classer. C'est là une version de base de l'algorithme que l'on peut raffiner. Souvent, on pondère les voisins par la distance qui les sépare du nouveau texte. On accorde plus de poids, lors de la prise de décision aux documents plus similaires.

La valeur de k est un des paramètres à déterminer lors de l'utilisation de ce type de classifieur. Il est à noter qu'il n'existe pas de solution pour choisir une bonne valeur pour

ce paramètre. Ce choix relève d'un compromis. Si « k » est trop petit, le nombre d'exemples qui prennent part à la décision est faible. et les exemples bruités peuvent alors jouer un rôle important. Si k est trop grand, l'hypothèse de localité n'est plus respectée, car des exemples très éloignés du document sont sélectionnés pour participer au vote. Dans le domaine textuel, la valeur optimale pour k dépend du corpus et de l'application. D'après les travaux réalisés jusqu'à présent, la meilleure classification est obtenue avec une valeur de k comprise entre 10 et 50 [Hmeidi, 2008] [Bawaneh, 2008].

Une des caractéristiques fondamentales de ce type du classifieur est l'utilisation d'une mesure de similarité entre les documents. Les textes étant représentés sous forme vectorielle, donc comme des points dans un espace à n dimensions. On peut au premier abord penser à déterminer les voisins les plus proches en calculant la distance euclidienne entre ces points comme suit :

$$d(a,b) = \sqrt{\sum (p_t(a) - p_t(b))^2} \qquad (2.7)$$

Où

- t : l'ensemble des attributs.
- $p_t(a)$: le poids du terme t dans le document a.
- $p_t(b)$: le poids du terme t dans le document b.

D'autres façons de calculer la similarité des documents sont utilisées, comme la similarité cosinusoïdale. Elle est préférable en classification de textes pour plusieurs raisons. Premièrement, la similarité cosinusoïdale permet de comparer des textes de longueurs différentes en normalisant leurs vecteurs. De plus, elle met l'accent plutôt sur la présence de mots que sur l'absence de mots. Justement, la présence de mots est probablement plus représentative de la catégorie du texte que l'absence de mots.

La similarité cosinusoïdale entre deux documents a et b est mesuré comme suit :

$$d(a,b) = \sum_t \frac{p_t(a) - p_t(b)}{\sqrt{\sum_t p_t(a)^2 - \sum_t p_t(b)^2}} \qquad (2.8)$$

K-PPV a été utilisé en catégorisation automatique de textes pour la première fois par [Masand, 1992]. [Yang ,1994] a montré que l'algorithme est parmi les meilleurs. Il est très souvent utilisé comme élément de comparaison [Joachim, 1998][Cooley, 1999]. Quelques auteurs ont cherché à l'améliorer [Lam, 1998] [Han, 2001] [Baoli, 2003].

2.2.3 Machine à Vecteur Support (SVM)

La notion de machines à vecteurs supports a été proposée par Vapnik [Vapink, 1995]. L'algorithme repose sur une interprétation géométrique simple [Burges, 1998] : il s'agit de trouver l'hyperplan séparant les deux classes qui maximise la marge. Cette dernière se définit comme la plus petite distance entre les exemples de chaque classe et la surface séparatrice s :

$$marge(s) = \sum_{c_j \in c} \min x_i \in c_j \left(d(x_i, s)\right) \qquad (2.9)$$

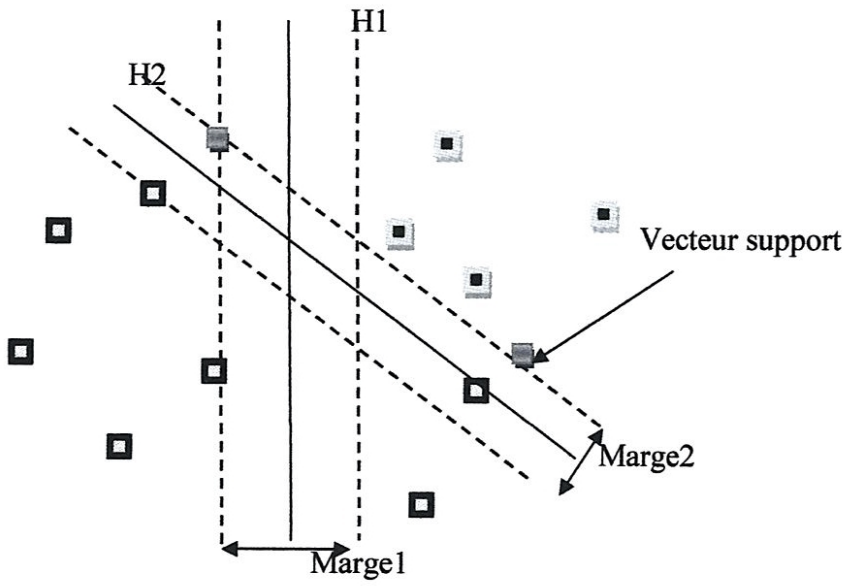

Figure 2-1 : exemple d'hyperplans séparateurs en dimension deux.

Dans l'exemple de la figure 2-1, les exemples des deux classes peuvent être séparés par un hyperplan, le problème est dit *linéairement séparable.* Les deux hyperplans H1 et H2 sont tous les deux des séparateurs acceptables, mais l'hyperplan H1 a une plus grande marge et sera donc préféré. Pour calculer l'hyperplan optimal et donc la marge, seul les exemples les plus proches de la zone frontière sont mis à contribution. L'apprentissage consiste à déterminer ces exemples appelés vecteurs de support. Tous les autres peuvent être écartés et n'interviennent plus dans les calculs.

Le problème se traduit mathématiquement en un problème d'optimisation quadratique. Trouver l'hyperplan (w, b) (b *est la distance à l'origine de l'hyperplan*) qui minimise la norme de *w* sous les contraintes :

$$\forall\, d_i, c_i\ (w.d_i - b) \leq 1 \qquad (2.10)$$

Avec di le $i^{ème}$ document, de la classe Ci (+1 ou -1).

Pour calculer cette marge, nous procédons comme suit :

Un hyperplan a pour équation :

$$y = < \vec{w}, \vec{d} > + b \qquad (2.11)$$

$< \vec{w}, \vec{d} >$ Dénote le produit scalaire entre les vecteur $\vec{w}$ et $\vec{d}$. Pour un individu $\vec{d}$ de classe y.

On cherche *w* tel que :

$$\begin{cases} < \vec{w}, \vec{d} > + b > +1 & si \quad y = +1 \\ < \vec{w}, \vec{d} > + b \leq -1 & si \quad y = -1 \end{cases} \qquad (2.12)$$

Donc on a :

$$y(< \vec{w}, \vec{d} > + b - 1) > 0 \qquad (2.13)$$

$\vec{d}$ est un vecteur qui représente un document quelconque dans un espace à n dimensions et y est la classe de document (+1 ou -1).

Pour résoudre ce problème, on utilise généralement un opérateur appelé le lagrangien et noté L_p comme somme de fonctions à optimiser (fonctions objectives), et l'opposé de chaque contrainte γ_i multipliée par une constante $\alpha_i \in \Re^+$ qui constitue un multiplicateur de Lagrange. Nous avons donc :

$$L_P = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i \gamma_i (< \vec{w}, \vec{d} > + b) + \sum_{i=1}^{n} \alpha_i \qquad (2.14)$$

L_P doit être minimisé par rapport à $\vec{w}$ et b, et il faut que les dérivées par rapport à $\vec{w}$ soient nulles.

Le gradient de L_p doit être nul par rapport à $\vec{w}$ et b on écrit :

$$\begin{cases} \frac{\partial L_p}{\partial \vec{w}} = 0 \\ \frac{\partial L_p}{\partial b} = 0 \end{cases} \quad d'ou \quad \begin{cases} \vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{d}_i \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases} \qquad (2.15)$$

De la formulation de L_p et de ces deux équations, on tire la formulation duale du lagrangien en éliminant $\vec{w}$

$$L_d = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j < \vec{d_i}, \vec{d_j} > \qquad (2.16)$$

Cette dernière doit être maximisée. Le maximum de L_d et le minimum de L_p sont obtenus pour les mêmes valeurs de $\vec{w}, b, \alpha_i$.

2.2.3.1 Classification d'une nouvelle donnée

La classe d'une donnée d *est* ± 1. Elle est fournie par le signe de $< \vec{w}, \vec{d} > +b$.

Si $< \vec{w}, \vec{d} > +b > 1$ cela signifie que x est au-dessus de H+.

Sinon si $< \vec{w}, \vec{d} > +b \leq -1$ cela signifie que x est en dessous de H-.

En note par $sgn(< \vec{w}, \vec{d} > +b)$ le signe de la quantité $< \vec{w}, \vec{d} > +b$.

Puisque $\vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{d}_i$ et sachant que seuls les vecteurs supports ont un multiplicateur de Lagrange non nul, on a :

$$sgn(< \vec{w}, \vec{d} > +b) = sgn(\sum_{i=1}^{n} \alpha_i y_i (< \vec{d_i}, \vec{d} > +b)) = sgn(\sum_{j=1}^{m} \alpha_j y_j (< \vec{s_j}, \vec{d} > +b)) \quad (2.17)$$

Où d est l'instance à classer et d_i sont les exemples d'apprentissage. Les s_j sont les vecteurs supports. m est le nombre de ces vecteurs.

2.2.3.2 Cas des classes non linéairement séparables

Si les données ou les exemples d'apprentissage ne sont pas linéairement séparables, on peut les plonger conceptuellement dans un espace de dimension plus grande (la dimension peut même être infinie) par une fonction de transformation appelée noyau (kernel). Dans cet espace, les exemples seront plus facilement séparables. Une propriété de l'algorithme est qu'il ne requiert pas les coordonnées de chaque exemple, mais seulement les produits scalaires de chaque couple d'exemples, qui restent calculables une fois les exemples plongés dans un nouvel espace même de dimension infini. Le théorème de Cover en 1965 indique qu'un ensemble de données non linéaires transformé dans un espace de plus grande dimension a plus de chance d'être linéairement séparable que dans son espace d'origine.

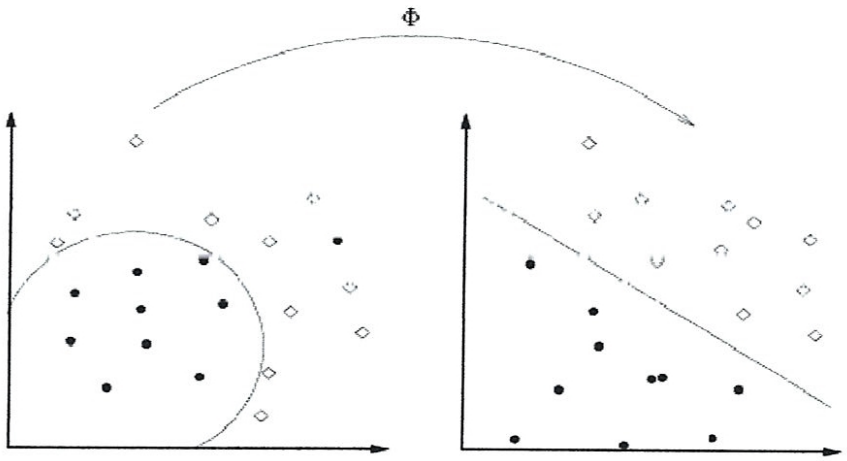

Figure 2-2: Transformation des données vers un espace de plus grande dimension

Cet algorithme est particulièrement bien adapté à la CT, car il est capable de gérer des vecteurs de grande dimension, et de sélectionner les attributs pertinents sans sur apprentissage [Vinot, 2004]. Dans la pratique en CT, les catégories sont quasiment toujours linéairement séparables. Il n'est donc pas nécessaire d'employer les méthodes avec des noyaux sophistiqués qui alourdissent inutilement les calculs. SVM a été introduit en CT pour la première fois par Joachims [Joachims, 1998] qui a notamment travaillé à rendre SVM compatible avec les données typique de la CT (grande dimension).
Depuis cette date, l'algorithme est très souvent réutilisé; par exemple pour la catégorisation de dépêches [Cooley, 1999], ou la détection de courriers électroniques non sollicités [Drucker, 1999].

2.2.4 La combinaison de plusieurs classifieurs

Chaque algorithme possède ses forces et ses faiblesses pour classifier de nouveaux documents. En outre, certaines de ces forces sont complémentaires et peuvent améliorer les résultats lorsqu'elles sont combinées. Cette combinaison est surtout utile lorsque les classifieurs se trompent sur des documents différents. Si les algorithmes font les mêmes erreurs, cette technique perd tout son lustre et ne change presque rien aux résultats. Il y a de nombreuses façons de conjuguer les efforts de plusieurs classifieurs en un comité.

Un comité est un ensemble de classifieurs dont on combine les prédictions d'une façon quelconque [Angeline, 1993].La manière la plus simple est sans contredit d'entraîner les classifieurs le plus indépendamment possible, puis de les faire voter. La prédiction du comité est la classe qui revient le plus souvent. Il a été démontré que l'efficacité moyenne d'un comité est supérieure à la moyenne d'efficacité des classifieurs de ce comité [Bodo, 1999].

Une méthode un peu plus complexe, mais aussi plus susceptible d'apporter une amélioration à l'efficacité se nomme « stacking » [Cooper, 1993]. Il s'agit d'entrainer séquentiellement plusieurs algorithmes, où le $n^{ème}$ classifieur, dans son apprentissage, tient compte de l'erreur de généralisation de ceux qui le précède. Par exemple, le comité peut contenir huit classifieurs, dont sept sont entrainés indépendamment sur un même échantillon. L'apprentissage du huitième consiste à associer le meilleur des sept classifieurs à utiliser selon les caractéristiques du document à classer.

Une autre alternative est le « boosting » [Cohen, 2002] dont le principe est d'agréger les résultats de plusieurs classifieurs pour obtenir un résultat plus robuste. Les différents classifieurs peuvent correspondre à différents algorithmes [Littlestone, 1994] ou au même algorithme utilisé avec différents sous échantillons du corpus d'apprentissage [Dagan, 1995]. Dans le boosting, les différents classifieurs sont appris séquentiellement. A chaque étape, le corpus d'origine est échantillonné suivant une distribution qui favorise le tirage des exemples mal classés par le classifieur construit à l'étape précédente. Les classifieurs se focalisent ainsi de plus en plus sur les éléments difficiles à catégoriser. Lors de la phase de test, tous les classifieurs sont utilisés et un vote pondéré par les taux de performance de chaque système est effectué. Le boosting est souvent utilisé avec un algorithme peu performant, mais très rapide (*surnommé Weak Learner*) avec lequel on construit plusieurs centaines de classifieurs. L'algorithme garantit que le taux d'erreur sur le corpus d'apprentissage peut être rendu aussi petit que l'on veut en augmentant simplement le nombre de classifieurs construits. Les expériences menées montrent qu'en pratique les performances en généralisation sont également très bonnes (il n'y a pas de sur-apprentissage) [Schapire, 1997][Drucker, 1996]. Et comme les SVM le boosting a été adapté à la CT avec succès [Schapire, 1998][Iyer, 2000][Kim, 2000][Schapire, 2000]

2.3 La Validation

La validation est une phase indispensable à tout processus d'apprentissage. Elle consiste à vérifier que le modèle construit sur la base d'apprentissage est un classifieur performant. Dans ce cas, *performant*, signifie qu'il permet de classer tout individu avec le minimum d'erreurs possible.

Les méthodes de validation vont dépendre de la nature de la tâche et du problème considéré. Nous distinguerons deux modes de validation : statistique et par expertise.
Pour certains domaines d'application (le diagnostic médical, par exemple), il est essentiel que le modèle produit soit compréhensible. Il y a donc une première validation du modèle produit par l'expert, celle-ci peut être complétée par une validation statistique sur des bases de cas existantes.

Pour la validation statistique, la première tâche à réaliser consiste à utiliser des méthodes élémentaires de statistique. L'objectif est d'obtenir des informations qui permettront de juger le résultat obtenu, ou d'estimer la qualité ou les biais des données d'apprentissage.
Pour la classification supervisée, la deuxième tâche consiste à décomposer les données en plusieurs ensembles disjoints. L'objectif est de garder des données pour estimer les erreurs des modèles, ou de les comparer. Il est souvent recommandé de constituer :

- Un ensemble d'apprentissage.
- Un ensemble de test.
- Un ensemble de validation.

L'ensemble d'apprentissage permet de générer le modèle. L'ensemble de test permet d'évaluer l'erreur réelle du modèle sur un ensemble indépendant évitant ainsi un biais d'apprentissage. Lorsqu'il s'agit de tester plusieurs modèles et de les comparer, on peut sélectionner le meilleur modèle selon ses performances sur l'ensemble de validation et ensuite évaluer son erreur réelle sur l'ensemble de test (Figure 2-2).

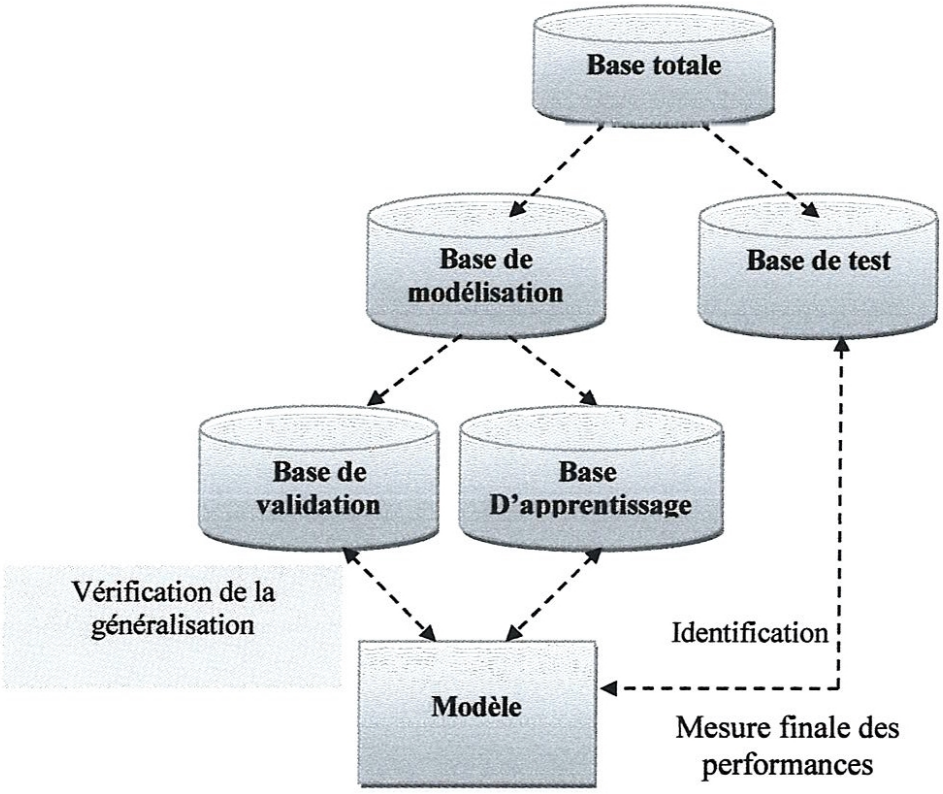

Figure 2-3: processus de validation par le test

Lorsqu'on ne dispose pas de suffisamment d'exemples, on peut se permettre d'apprendre et d'estimer les erreurs avec un même ensemble par la technique de validation croisée.

2.3.1 Validation croisée

Le corpus est découpé en **x** parties de même taille (souvent 10). Le test se fait en phases. A chaque étape, (x-1) ensemble sont regroupés pour former le corpus d'apprentissage, et le classifieur est testé sur les exemples du dernier ensemble. Le taux de performance est calculé comme la moyenne des taux sur chaque essai (Figure 2-3).

La validation croisée sert à estimer l'erreur réelle d'un modèle selon l'algorithme suivant :

Algorithme1 Validation croisée

1. S : un ensemble, x : un entier
2. Découper S en x parties égales S_1, ..., S_X
3. Pour i de 1 à x {
4. Construire un modèle M avec l'ensemble S - S_i
5. Evaluer une mesure d'erreur de M avec S_i }
6. Retourner l'espérance mathématique des mesures des erreurs

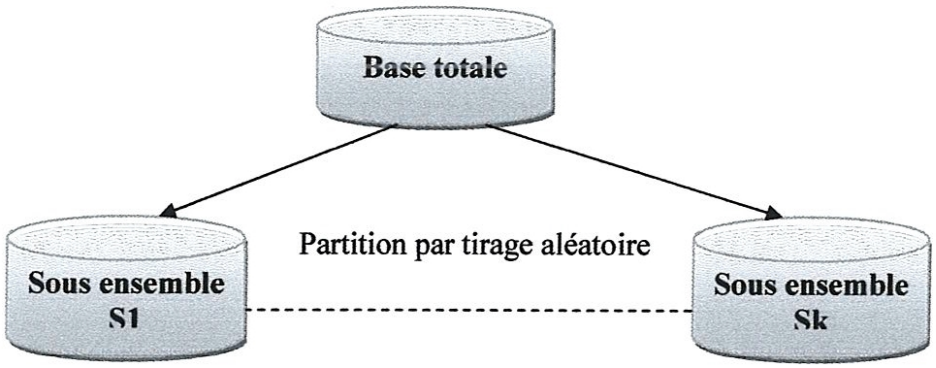

Figure 2-3: processus de validation croisée

Si la taille des *Si* est de un individu, on parle alors de validation par « *leave one out* ». L'apprentissage se fait donc sur tout le corpus sauf un exemple; celui qui est testé.

2.3.2 Mesures de performances

Afin de valider correctement la procédure de classification, nous utilisons des mesures de performances sur les résultats de la classification.

L'efficacité peut se définir selon plusieurs critères. Les deux critères généralement utilisés pour évaluer un processus de catégorisation sont : la précision et le rappel.
Avant de donner la définition formelle de ces deux mesures, il est nécessaire de définir les quatre notions suivantes pour une classe *i* :

- *VP* est l'ensemble des textes de la classe *i* bien classés.
- *FP* est l'ensemble des textes assignés par erreur à la classe *i.*
- *FN* est l'ensemble des textes de la classe *i* non classés *i* par le classifieur.
- *VN* est l'ensemble des textes n'appartenant pas à la classe *i* et identifiés comme tels.

On peut visualiser ces notions sur le tableau suivant :

Classe i		Classement de l'expert	
		VRAI	FAUX
Classement du système	POSITIF	VP	FP
	NEGATIF	FN	VN

Tableau 2-1: les quatre possibilités d'un classifieur

a. Précision – Rappel - F-Mesure

La précision et le rappel sont deux quantités qui sont définies lorsque les classifieurs prennent des décisions binaires (filtre) : soit un document est sélectionné par le filtre, soit il ne l'est pas. La précision pour une classe est le nombre de documents que le système a attribué à cette classe et l'expert a confirmé cette appartenance. Autrement dit, cette mesure indique la capacité du classifieur à classer correctement les documents, ou tout simplement, elle permet de savoir en particulier si le classifieur, quand il classifie des documents, n'affecte pas trop de documents à une classe par erreur. Formellement, la précision s'exprime de la façon suivante :

$$p_i = \frac{VP}{VP + FP} \qquad (2.18)$$

Le rappel est le rapport entre le nombre de documents attribués à la classe par le système et le nombre de documents que l'expert a attribué à la classe. Autrement dit, cette mesure indique la capacité du classifieur à classer correctement l'intégralité des documents. Formellement, le rappel s'exprime de la façon suivante :

$$R = \frac{VP}{VP + FN} \qquad (2.19)$$

Le rappel et la précision sont souvent utilisés, car ils reflètent le point de vue de l'utilisateur; si la précision est faible, l'utilisateur sera insatisfait, car il devra perdre du temps à lire des informations qui ne l'intéressent pas. Si le rappel est faible, l'utilisateur n'aura pas accès à une information qu'il souhaitait avoir.

Un filtre parfait doit avoir une précision et un rappel égal à un (1), mais ces deux exigences sont souvent contradictoires et une très forte précision ne peut être obtenue qu'au prix d'un rappel faible et vice-versa.

Généralement on ne peut tenir compte de la précision ou du rappel séparément lors de l'évaluation des performances d'un classifieur. Pour cette raison, le *« break-even point »*, le point où la précision et le rappel sont égaux est calculé. Souvent, le *break-even point* n'existe pas, pour cela une autre mesure appelée le F-mesure ou F-score est généralement calculée.

$$F_B = \frac{(B^2 + 1)PR}{B^2P + R} \qquad (2.20)$$

La valeur du paramètre β permet d'accorder plus ou moins de poids à la précision d'un système. Habituellement, la valeur de β est fixée à 1 et la mesure est ainsi appelée ***F-score***.

b. Taux d'erreur et de performance

Lorsque le problème est multi classe, les mesures de précision rappel ne sont pas adaptées. En effet, lorsque les classes sont prises en compte simultanément, le rappel et la précision deviennent liés : une erreur qui fait baisser le rappel dans une classe fait aussi

baisser la précision dans une autre classe. On utilise alors généralement soit le taux d'erreur, soit son opposé, le taux de bonne classification (*accuracy en anglais*) :

Taux d'erreur :

$$E = \frac{FP + FN}{VP + FN + FP + VN} \qquad (2.21)$$

Taux de performance :

$$PER = \frac{VP + VN}{VP + FN + FP + VN} \qquad (2.22)$$

c. Micro et macro moyenne

Toutes ces mesures fondées sur l'utilisation de la table de contingence sont définis pour une catégorie. Il existe deux méthodes pour agréger les résultats de chaque classe. Soit, on construit une table de contingence globale où $VP = \sum VP_I$, et de même pour VN et FP *et* FN), et l'on calcule les valeurs P, R, E, PER à partir de cette matrice . C'est la micro moyenne.

Soit on calcule P, R, E, PER pour chaque classe, et l'on fait la moyenne de chaque valeur. C'est la macro moyenne.

$$p = \frac{1}{|C|} * \sum p_j \qquad (2.23)$$

Si la majorité des études utilise la macro moyenne, le choix de privilégier l'une ou l'autre de ces mesures dépend intrinsèquement de l'application dans laquelle sera utilisé le système.

2.4 Conclusion

Suite à la présentation de la catégorisation automatique de textes au premier chapitre, il a été question dans ce chapitre de donner un aperçu sur les différents classifieurs mis en œuvre pour traiter ce problème. On a pu constater la variété de techniques d'apprentissage pouvant amener une application informatique à classer des textes avec autonomie. De façon

détaillée, on comprend bien maintenant le fonctionnement du classifieur Naive Bayes, de l'algorithme des k plus proches voisins et des machines à vecteurs support qui semblent actuellement le plus performant des classifieurs. La deuxième partie du chapitre a fait un aperçu sur le processus d'évaluation des classifieurs, mais il est à noter qu'il est difficile de fournir des valeurs chiffrées sur les performances qu'un système de catégorisation peut actuellement atteindre. La tâche est souvent subjective : lorsque deux experts humains doivent déterminer les catégories d'un ensemble de documents, il y a souvent désaccord sur plus de 5 % des textes. Mettre une référence sur ce constat, il est donc illusoire de rechercher une classification automatique parfaite. Actuellement, les systèmes de CT avec apprentissage automatique ont rattrapé les performances des systèmes manuels qui nécessitent de longs mois de développement à des experts humains pour fournir les règles de catégorisation.

Les meilleurs algorithmes obtiennent sur les corpus standards (Reuters, Newsgroups) souvent plus de 90 % de bonne classification.

Chapitre 3

La Langue Arabe

Sommaire

3.1 Introduction

La langue Arabe est la langue parlée à l'origine par le peuple arabe. C'est la langue officielle d'au moins 22 pays, et est parlée par plus de 250 millions de personnes. Elle est utilisée communément dans beaucoup de pays islamiques, parce que c'est la langue spirituelle de l'islam. L'expansion et le développement de la langue arabe ont été intimement liés à la naissance et la diffusion de l'islam. L'arabe s'est imposé, depuis l'époque arabo musulmane, comme langue religieuse, mais plus encore comme langue d'administration, langue de la culture et de la pensée, d'ouvrages historiques, de dictionnaires, de traités des sciences et des techniques. Ce développement s'est accompagné d'une rapide et profonde évolution en particulier dans la syntaxe et l'enrichissement lexical.

3.2 Particularité de la langue arabe

L'alphabet de la langue arabe compte 28 lettres (Tableau 3-1). L'arabe s'écrit et se lit de droite à gauche. Les lettres changent de forme de présentation selon leur position (au début, au milieu ou à la fin du mot) [Boulaknadel, 2005], ce qui les rendent extensibles à 90 lettres avec l'ajout des voyelles [Aljlayl, 2002]. Le Tableau 3-2 montre les variations de la lettre ع (Ayn). Toutes les lettres se lient entre elles sauf (ا, و, ر, ز, د, ذ) qui ne se joignent pas à gauche.

N°	Caractère arabe	Valeur Unicode	Forme translittéré	Prononciation
1	ا	U+0627	A	Alef
2	ب	U+0628	B	Ba
3	ت	U+062A	T	Ta
4	ث	U+062B	Th	Tha
5	ج	U+062C	J	Jim
6	ح	U+062D	H	Ha
7	خ	U+062E	Kh	Kha

8	د	U+062F	D	Del
9	ذ	U+0630	D	Dhel
10	ر	U+0631	R	Ra
11	ز	U+0632	Z	Zi
12	س	U+0633	S	Sin
13	ش	U+0634	Ch	Chin
14	ص	U+0635	S	Sad
15	ض	U+0636	D	dhad
16	ط	U+0637	T	Ta
17	ظ	U+0638	Z	Dha
18	ع	U+0639	‘’	Ayn
19	غ	U+063A	Gh	Ghayn
20	ف	U+0641	F	Fa
21	ق	U+0642	Q	Qaf
22	ك	U+0643	K	Kaf
23	ل	U+0644	L	Lam
24	م	U+0645	M	Mim
25	ن	U+0646	N	nun
26	ه	U+0647	H	Ha
27	و	U+0648	W	wew
28	ي	U+064A	Y	Ya

Tableau 3-1: les lettres arabes

À la fin d'une lettre non joignable	**À la fin**	**Au milieu**	**Au début**
ع	ـع	ـعـ	عـ

Tableau 3-2 : Exemple de variation de la lettre ع

Un mot arabe s'écrit avec des consonnes et des voyelles. Les voyelles quant à elles sont de deux types : les voyelles courtes ou brèves qui ont la forme d'une marque

diacritique (ـَ ، ـِ ، ـُ), ajoutées au-dessus ou au-dessous des lettres, tandis que les voyelles longues (ي ، ا ، و) collent aux consonnes, et sont toujours écrites même dans les formes non vocalisées. *Le tanwin* (ـً ، ـٍ ، ـٌ) est un autre genre de voyelles [El kassas, 2005]. Il est réalisé par un signe diacritique fusionné au signe de la voyelle courte (voir tableau 3-3).

voyelle	Valeur Unicode	Définition
Fatha ـَ	U+064E	surmonte la consonne et se prononce comme un « a » en français
damma ـُ	U+064F	surmonte la consonne et se prononce comme un « u » en français
kassra ـِ	U+0650	se note au-dessous de la consonne et se prononce comme un « i »
Sukun ـْ	U+0652	indique qu'une consonne n'est pas suivie (ou mue) par une voyelle. Il est noté toujours au-dessus de la consonne.
Fathatan ـً	U+064B	se positionnent au-dessus de la consonne et on la prononce ann
Kasratan ـٍ	U+064D	se positionnent au-dessous de la consonne, on la prononce inn
Dammatan ـٌ	U+064C	se positionnent au-dessus de la consonne, on la prononce onn
chadda	U+0651	Comme dans le français "immédiatement", l'arabe peut renforcer une consonne quelconque. Ce renforcement est indiqué à l'aide d'un signe nommé *chadda* (intensité).
wasla	U+0671	quand la voyelle d'un *alif* au commencement d'un mot doit être absorbée par la dernière voyelle du mot qui précède, on en indique l'élision par le signe wasla placé au-dessus de l'*alif*.
madda	U+0653	Prolongation se place sur l'*alif* pour indiquer que cette lettre tient lieu de deux *alifs* consécutifs ou qu'elle ne doit pas porter le *hamza*. Ce signe de contraction a la forme d'un *alif* horizontal.

Tableau 3-3: les voyelles arabes

Les voyelles sont nécessaires à la lecture et à la compréhension correcte d'un texte, et permettent de différencier des mots ayant la même représentation. Cependant ils ne sont utilisés que pour le coran, ou les textes sacrés et didactiques [Larkey, 2005].

Le tableau suivant illustre les différentes interprétations possibles des mots كتب et مدرسة dans les cas où les voyelles sont omises.

Mot sans voyelles	1ère interprétation		2 ème interprétation		3 ème interprétation	
كتب	كَتَبَ	Il a écrit	كُتِبَ	Il a été écrit	كُتُب	Des livres
مدرسة	مَدرَسَة	École	مُدَرسَة	enseignante	مُدَرَسَة	enseignée

Tableau 3-4 : Ambiguïté causée par l'absence des voyelles

Il faut noter que l'ambiguïté causée par l'absence des voyelles courte est atténuée par l'association de sens, de contexte, etc.

3.2.1 Morphologie Arabe

Le lexique arabe comprend trois catégories de mots : verbes, noms et particules. Les verbes et les noms sont le plus souvent dérivés d'une racine à trois consonnes radicales [Baloul, 2002], et avec un degré moindre à quatre consonnes, et rarement à cinq [Darwish, 2002][Tuerlinckx, 2004]. Une famille de mots peut ainsi être générée d'un même concept sémantique à partir d'une seule racine, à l'aide de différents schèmes. Ce phénomène est caractéristique à la morphologie arabe. On dit donc que l'arabe est une langue à racines réelles à partir desquelles on déduit le lexique arabe selon des schèmes qui sont des adjonctions et des manipulations de la racine.

Le tableau 3-5 présente quelques exemples de schèmes appliqués aux mots كتب « *Écrire* » et حمل « *porter* ». On peut ainsi dériver un grand nombre de noms, de formes et de temps verbaux.

Schèmes	KTB	كتب	Notion d'écrire	HML	حمل	Notion de portée
$R_1àR_2iB$	KàTiB	كَاتِب	Écrivain	HàMiL	حَامِل	Porteur
$R_1aR_2aR_3a$	KaTaBa	كَتَبَ	A écrit	HaMaLa	حَمَلَ	A porté
$MaR_1R_2aR_3$	maKTaB	مَكتَب	Bureau	maHMaL	مَحمَل	Brancard
$R_1uR_2iR_3a$	KuTiBa	كُتِبَ	A été écrit	HuMiLa	حُمِلَ	A été porté

Tableau 3-5: Exemple de schèmes pour les mots كتب écrire et حمل porter

Les lettres en majuscule (**Ri**) désignent les consonnes de base qui composent la racine. (**â, a, i,..**) désignent les voyelles, et les consonnes en minuscule (**m,..**) sont des consonnes de dérivation utilisées dans les schèmes.

La majorité des verbes arabes ont une racine composée de trois consonnes. L'arabe comprend environ (150) schèmes ou patrons, dont certains plus complexes, tel le redoublement d'une consonne, ou l'allongement d'une voyelle de la racine, l'adjonction d'un ou de plusieurs éléments, ou la combinaison des deux. Une autre caractéristique est le caractère flexionnel des mots : les terminaisons permettent de distinguer le mode des verbes et la fonction des noms [Baloul, 2002].

3.2.2 Structure d'un mot arabe

En arabe, un mot peut signifier toute une phrase grâce à sa structure composée, qui est une agglutination d'éléments de la grammaire. La représentation suivante schématise une structure possible d'un mot. Notons que la lecture et l'écriture d'un mot se font de droite vers la gauche.

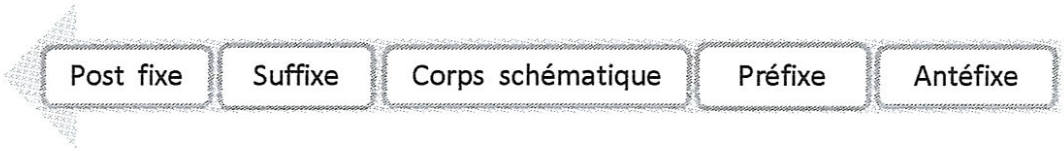

Figure 3-1: Structure d'un mot arabe

- **Antéfixes** sont des prépositions ou des conjonctions.
- **Préfixes et suffixes (les affixes) :** expriment les traits grammaticaux et indiquent les fonctions : cas du nom, mode du verbe et les modalités (nombre, genre,…).
- **Corps schématique :** racine de mot (stem).
- **Postfixes** sont des pronoms personnels.

Exemple

أتتذكروننا ؟ = "*Est ce que vous vous souvenez de nous* ?"

La segmentation de ce mot donne les constituants suivants :

Post fixe	suffixe	stem	préfixe	antéfixe
نا	ون	تذكر	ت	أ
Pronom suffixe complément de nom	Suffixe verbal exprime le pluriel	dérivé de la racine selon le schème $taR_1aR_2aR_3a$	prefixe verbal du temps de l'inaccomplie	conjonction d'interrogation

Tableau 3-6: segmentation de mot 'أتتذكرونا'

3.2.3 Catégories de mots arabes

L'arabe considère 3 catégories de mots [Khoja, 2001][Maamouri, 2004] : Nom, verbe et particule.

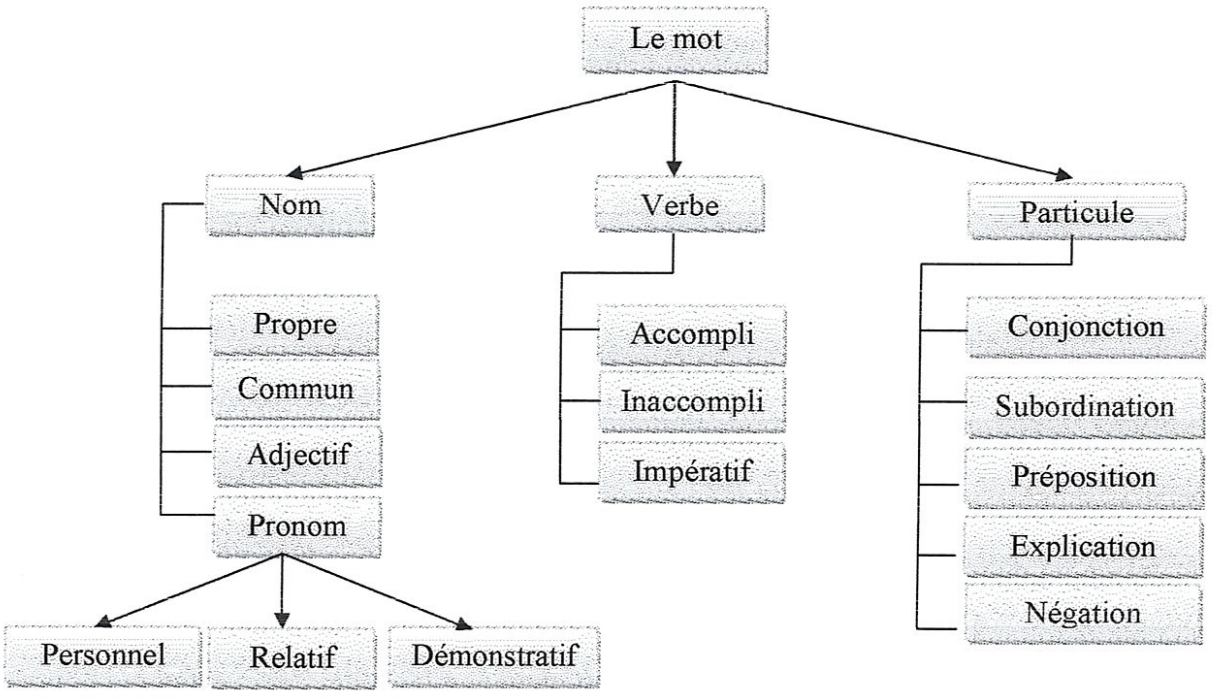

Figure 3-2: classification hiérarchique de mot proposée par khoja

3.2.3.1 Le verbe

Le verbe est une entité exprimant un sens dépendant du temps. C'est un élément fondamental, auquel se rattachent directement ou indirectement les divers mots qui constituent l'ensemble.

La plupart des mots en arabe dérivent d'un verbe de trois lettres. Chaque verbe est donc la racine d'une famille de mots. Comme en français, la conjugaison des verbes se fait en ajoutant des préfixes et des suffixes. La langue arabe dispose de trois temps :

A. **L'accompli**

Correspond au passé et se distingue par des suffixes.

Exemple :

Pour le pluriel féminin, on dit : كتبن 'KaTaBna', *elles ont écrit.*

Et pour le pluriel masculin, on dit : كتبوا 'KaTaBuu', *ils ont écrit.*

B. **L'inaccompli**

Présente l'action en cours d'accomplissement, ses éléments sont préfixés.

Exemple :

Pour le masculin singulier, on dit : 'يكتب ' yaKTuBu *il écrit.*

Pour le féminin singulier, on dit : ' تكتب ' taKTuBu, *elle écrit.*

C. **L'impératif**

Se distingue par des suffixes ainsi le préfixe « ا ».

Exemple :

Pour le dual masculin ' اكتبا ' oKToBa écrivez.

Pour le pluriel féminin ' اكتبن ' oKToBna écrivez.

En résumé, la conjugaison des verbes dépend de plusieurs facteurs :

- **Le temps**
  - ✓ Accompli.
  - ✓ Inaccompli.
  - ✓ Impératif.

- **Le nombre du sujet**
  - ✓ Singulier
  - ✓ Duel.
  - ✓ pluriel.
- **Le genre du sujet**
  - ✓ Masculin.
  - ✓ Féminin.
- **Le mode**
  - ✓ Actif.
  - ✓ Passif.

3.2.3.2 Le nom

Le nom est l'élément désignant un être ou un objet qui exprime un sens indépendant du temps. La catégorie des noms regroupe toutes les unités lexicales référant à un sens qui n'est pas lié au temps. Cette catégorie comprend le substantif et l'adjectif (الصفة و الموصوف) Les substantifs arabes sont de deux catégories, ceux qui sont dérivés de la racine verbale, et ceux qui ne le sont pas comme les noms propres et les noms communs [Elkassas, 2005]. Dans le premier cas, le fait que le nom soit dérivé d'un verbe, il exprime donc une certaine sémantique qui pourrait avoir une influence dans la sélection des phrases saillantes d'un texte pour le résumé [Douzidia, 2004].

La déclinaison des noms se fait selon les règles suivantes :

- **Le féminin singulier** : on ajoute le ة à la fin de nom.

  Exemple :

  صغير *'petit'* devient صغيرة 'petite'.

- **Le féminin pluriel** : de la même manière, on rajoute pour le pluriel les lettres ' ات '.

  Exemple :

  صغير *'petit'* devient صغيرات *'petites'*.

- **Le masculin pluriel** : pour le pluriel masculin, on rajoute les deux lettres ' ين ' ou ' ون ' dépendamment de la position du mot dans la phrase (sujet ou complément d'objet).

  Exemple :

  الراجع 'revenant' devient الراجعين **ou** الراجعون '*revenants*'.

A. **Le Pluriel irrégulier** : Il suit une diversité de règles complexes et dépend du nom.

Exemple :

طفل 'un *enfant*' devient أطفال '*des enfants*'.

Le phénomène du pluriel irrégulier dans l'arabe pose un défi à la morphologie, non seulement à cause de sa nature non concaténative, mais aussi parce que son analyse dépend fortement de la structure [Kiraz, 1996] comme pour les verbes irréguliers. Certains dérivés nominaux associent une fonction au nom :

- ✓ Agent (celui qui fait l'action).
- ✓ Objet (celui qui a subi l'action).
- ✓ Instrument (désignant l'instrument de l'action).
- ✓ Lieu.

Pour les pronoms personnels, le sujet est inclus dans le verbe conjugué. Il n'est donc pas nécessaire (comme c'est le cas en français) de précéder le verbe conjugué par son pronom. On distinguera entre singulier, duel (deux) et pluriel (plus de deux) ainsi qu'entre le masculin et féminin.

3.2.3.3 Les particules

Ce sont principalement les mots outils comme les conjonctions de coordination et de subordination. Les particules sont classées selon leur sémantique et leur fonction dans la phrase. On en distingue plusieurs types (introduction, explication, conséquence, ...). Elles jouent un rôle important dans l'interprétation de la phrase. Elles servent à situer des faits ou des objets par rapport au temps ou au lieu, elles jouent également un rôle clé dans la cohérence et l'enchaînement d'un texte.

Comme exemple de particules qui désignent un temps بعد 'après ' قبل 'avant ' , un lieu حيث *'où'*, ou de référence الذين *'ceux'*...

3.3 Propriétés linguistiques de la langue arabe

La langue arabe présente plusieurs propriétés linguistiques, parmi lesquelles on cite :

A. C'est une langue flexionnelle

En typologie des langues, on appelle langue flexionnelle une langue dans laquelle les lemmes ('mots') changent de forme selon les apports grammaticaux qu'ils entretiennent avec les autres lemmes. Certains mots modifient donc leur forme (sonore et/ou visuelle). On dit d'eux qu'ils subissent le jeu de la flexion. L'ensemble des formes différentes d'un même mot fléchi forme son paradigme [Elkassas, 2005].

Selon cette définition, l'arabe se classe comme une langue à morphologie extrêmement riche, parce que le système dérivationnel se présente comme un jeu de construction basé sur le rôle sémantique de l'élément dérivé. L'association d'affixes à une racine donnée permet d'engendrer des mots avec des significations différentes, mais qui peuvent être classés comme mots de la famille du dérivant.

Par exemple, les mots dérivés du verbe كسر signifiant 'casser' sont dérivés par doublement de la consonne « s », le verbe *kassara* 'casser en mille morceaux', et par ajout du préfixe « in », le verbe انكسر *inkasara* 'se casser'.

B. C'est une langue pro-drop

L'arabe standard moderne omet systématiquement la réalisation morphologique du pronom sujet. Le verbe s'accorde néanmoins en personne, genre et nombre avec le pronom omis, comme le montre l'exemple suivant. Le pronom correspondant est mis entre accolades :

اكلوا {هم}		اكلن {هن}
ont mangé {ils}	vs	ont mangé {elles}
'Ils ont mangé'(أكلوا)		'Elles ont mangé' (اكلن)

3.4 Problèmes du traitement automatique de l'arabe

A. Problèmes d'absence des voyelles

Un des aspects complexes de la langue arabe est l'absence des voyelles dans le texte, qui risque de générer une certaine ambiguïté à deux niveaux : identifier le sens du mot ainsi sa fonction dans la phrase (différencier entre le sujet et le complément...).

En catégorisation de texte, ceci peut influencer les fréquences de mots. Etant donné qu'elles sont calculées après la détection de la racine ou la lemmatisation des mots qui est basée sur la suppression de préfixes et suffixes. Lors du calcul des scores, il peut arriver que des mots soient considérés comme dérivants d'un même concept alors qu'ils ne le sont pas. Cette ambigüité pourrait, dans certains cas, être levée soit par une analyse plus profonde de la phrase ou des statistiques (par exemple, il est plus probable d'avoir العلم الوطني *le drapeau national* que la *science nationale*).

Considérons par exemple cet extrait réduit de texte électronique arabe :

ولد هذا العالم و الباحث في مصر

Ce texte peut être interprété et traduit selon les deux possibilités suivantes et qui sont toutes syntaxiquement correctes :

وُلِدَ هَذا العَلِمُ و الباحِثُ في مِصرَ

Ce savant et chercheur est né en Egypte

وَلَدُ هَذا العالِمُ و الباحِثُ في مِصرَ

Le fils de ce savant et chercheur est en Égypte

Le mot (« ولد » admet deux caractéristiques morphologiques différentes. Le mot (العالم) admet aussi deux caractéristiques {العالِم،العالم}.

De plus, la capitalisation n'est pas employée dans l'arabe ce qui rend l'identification des noms propres, des acronymes, et des abréviations encore plus difficiles [Hammou, 2002].

B. Détection de racine

Pour détecter la racine d'un mot, il faut connaître le schème par lequel il a été dérivé et supprimer les éléments flexionnels (antéfixes, préfixes, suffixes, post fixes) qui ont été ajoutés.

Le tableau 3-7 présente la liste des préfixes et des suffixes proposés par [Darwish, 2002]. Plusieurs d'entre eux ont été utilisés par [Cohen, 2002] pour la lemmatisation de mots arabes, ils ont été déterminés par un calcul de fréquence sur une collection d'articles arabes de l'Agence France Press (AFP).

préfixes							
فا	وبـ	ال	ومـ	نتـ	متـ	بتـ	والـ
لا	فيـ	للـ	كمـ	بمـ	وتـ	يتـ	فالـ
با	وا	ليـ	فمـ	لمـ	ستـ	لتـ	بالـ
suffixes							
ا	ـة	ين	ية	هم	ته	وه	ات
	ـه	يه	تك	هن	تم	ـان	وا
	ي	ية	نا	ها	كم	تي	ون

Tableau 3-7: liste des préfixes et suffixes utilisés dans (Al-stem)

L'analyse morphologique devra donc séparer et identifier des morphèmes semblables aux mots préfixés comme les conjonctions و 'wa' et ف 'fa', et des prépositions préfixées comme بـ 'bi' et لـ 'li', l'article défini 'الـ', des suffixes de pronoms possessifs.

La phase d'analyse morphologique détermine un schème possible. Les préfixes et les suffixes sont trouvés en enlevant progressivement des préfixes et des suffixes et en essayant de faire correspondre toutes les racines produites par un schème afin de retrouver la racine [Darwish, 2002].

Lorsqu'un mot peut être dérivé de plusieurs racines différentes, la détection de la racine est encore plus difficile, en particulier en absence de voyelles .Par exemple, pour le mot arabe أيمان 'Ayman' les préfixes possibles sont : Φ,'ا ' et ' أبـ ' et les suffixes possibles sont: Φ et ' ان ' (Tableau 3-8).

Stem	Stem translittéré	Préfixe	Schème	Suffixe	Racine	signification
أيمان	AymAn	Φ	$R_1yR_2aR_3$	Φ	امن	croyance
يمان	ymAn	ا	$R_1R_2aR_3$	Φ	يمن	Convenant
ايم	Aym	Φ	$R_1R_2R_3$	ان	ايم	Deux veuves

Tableau 3-8: Les stems possibles pour le mot أيمان

Certains verbes sont considérés comme irréguliers. Ce sont ceux qui portent des consonnes particulières dites faibles (ي, ا, و), ou des voyelles longues. Ils sont appelés ainsi parce que, lors de leur déclinaison, chacune de ces lettres est soit conservée, soit remplacée ou éliminés. Le tableau 3-9 donne un exemple de dérivation du mot 'قال' *dire*.

Caractère ا est remplacé par	قال	Dire
ا	قال	Il a dis
و	يقول	Il dit
ـي	قيل	Il a été dis
Φ	قال	dis

Tableau 3-9 : Exemple de déclinaison du verbe irrégulier 'قال'

C. L'agglutination

Une difficulté en traitement automatique de l'arabe est l'agglutination, par laquelle les composantes du mot sont liées les unes aux autres. Ce qui complique la tâche de l'analyse morphosyntaxique pour identifier les vrais composants du mot.

Exemple :

Prenons le mot أَلَمُهُمْ 'ALaMuhum', *leur douleur*

Dans sa forme voyellée il n'accepte qu'une seule segmentation : أَلَمُ + هُمْ « ALaMu+hum ». Dans sa forme non voyellée المهم 'ALMHM', le même mot accepte au moins les trois segmentations présentées dans le Tableau 3-10.

Segmentation possible		Traduction en français
ا+لم+هم	A+LM+hm	Les a-t-il ramassés
الم+هم	ALM+hm	Leur douleur
الم+هم	ALM+hm	Il les a fait souffrir
ال+مهم	Al+MHM	L'essentiel

Tableau 3-10: Exemple de segmentation de mot المهم

L'amplification de l'ambigüité de segmentation s'opère selon deux façons [Débili, 2002]. D'abord, il y a plus d'unités ambigües dans un texte non voyellé que dans son correspondant voyellé, mais aussi, les unités ambiguës acceptent plus de segmentations dans le texte non voyellé.

De plus, le fait de précéder la lemmatisation par la troncature des préfixes avant les suffixes (et réciproquement) peut influencer les résultats. En considérant l'exemple dans le Tableau 3-10, sur un texte où la notion de douleur est importante, le fait d'avancer la suppression des préfixes avant les suffixes les mots comme المهم *leur douleur (pour le pluriel)*, المهما *leur douleur* (pour le duel) exprimera une autre notion.

D. Problèmes de proclitique

Contrairement aux langues latines, en Arabe, les articles, les prépositions, les pronoms collent aux adjectifs, noms, verbes et particules auxquels ils se rapportent. Comparé au Français, un mot arabe peut parfois correspondre à une phrase en français. Exemple : le mot arabe ' تتذكرونا ' correspond en français à la phrase 'vous vous souvenez de nous'

Cette caractéristique engendre une ambigüité morphologique au cours de l'analyse. En effet, il est parfois difficile de distinguer entre une proclitique ou enclitique et un caractère original du mot. Par exemple, le caractère ' و ' dans le mot (وصل) est un caractère original alors que dans le mot (وفتح), il s'agit d'un proclitique (' و '+ ' فتح').

3.5 Travaux sur la catégorisation de textes Arabes

Les études sur la catégorisation des textes en langue Arabe ne sont qu'à leur début. Il n'y a pas encore des travaux de recherche avancés qui exploitent ou adaptent les techniques de la catégorisation actuelles pour la langue arabe. Dans cette section, nous allons exposer les travaux les plus connus, et les plus récents avec les résultats obtenus dans ce domaine pour cette langue.

- [MESLEH, 2007] implémente un système de catégorisation de documents arabes en utilisant les machines à vecteur supports avec la méthode de CHI square pour la sélection des termes, les résultats obtenus montrent que le système est très efficace. Ses performances en terme de F-mesure sont de l'ordre de 88.11%.

- [El-Halees, 2007] utilise la méthode de l'entropie maximale pour construire son système de classification de documents arabes, les performances de son système sont très encourageantes, il a obtenu des scores de F-mesure de l'ordre de 80.41 %.
- [El-Halles, 2006] décrit une méthode basée sur les règles d'association pour le classement des documents avec une précision de 74,41 %.
- [El Kourdi, 2004] utilise l'algorithme Naive Bayes pour la classification des documents. La précision qu'il a obtenu est loin des scores obtenus pour l'Anglais et les langues européennes. Elle n'est que de 68,78 %.
- Sakhr a monté Siraj, un système de classification automatique de documents Arabes, disponible à l'adresse suivante : *www.siraj.sakhr.com*. Malheureusement aucun support technique n'est disponible pour ce système. Ses performances ne sont même pas mentionnées.
- Un autre système proposé par [Sawaf, 2001] qui utilise les méthodes de classification statistique telle que le calcul d'entropie maximale pour classer les articles, il a obtenu des scores de F-mesure de l'ordre de 62.7 %.

D'autres travaux moins consistants basés sur des corpus montés localement tels que :

- [Hmeidi, 2008] a essayé de comparer les performances de KNN et SVM pour la catégorisation automatique de documents arabes. Il a prouvé que les deux classifieurs sont très performants en classification des documents Arabes et que les SVM dépassent clairement les KNN en terme de justesse de prédiction et en temps du calcul.
- Les travaux réalisés par [Khreisat, 2006] qui a procédé au calcul de l'effet de la méthode de représentation en n-grammes et le calcul de la dissimilarité entre documents. Elle obtient des résultats très variables d'une classe à une autre allant de 60 % à 93 %.
- Les travaux de [Djelailia, 2008] qui a visé à confirmer que les résultats d'une catégorisation de textes obtenus pour d'autres langues avec les SVM et basées sur la représentation sac de mots sont ou ne sont pas liées à la nature de la langue. Les résultats qu'il a obtenus sont très encourageants et sont de l'ordre de 90 % de précision.

3.6 Conclusion

Ce chapitre a pour objectif de présenter un petit aperçu sur la langue arabe ainsi ses caractéristiques et ses particularités linguistiques tout en faisant la lumière sur quelques points critiques qui rendent cette langue difficile à maitriser non seulement dans le domaine de la catégorisation automatique, mais aussi dans le domaine du traitement automatique de langage naturel.

Nous avons commencé notre chapitre par la description des particularités de la langue arabe, suivi de sa morphologie ainsi que ses propriétés linguistiques à savoir le caractère flexionnel et dérivationnel qui permet de rendre l'arabe une langue à morphologie extrêmement très riche.

Dans la deuxième partie de ce chapitre, on a pu soulevé quelque points critiques qui posent des problèmes en traitement automatique de la langue arabe, a savoir les problèmes d'ambigüité due à l'absence de voyelles, amplifiée par l'agglutination des mots qui complique la tâche de l'analyse morphosyntaxique ainsi la détection de la racine (*stemming*). Cette tâche bien qu'elle soit difficile pour les langues avec des morphologies complexes comme l'arabe, elle est particulièrement importante et utile en particulier dans

Chapitre 4

Approche Proposée

Sommaire

4.1 Introduction

Plusieurs thèmes gravitant autour de la catégorisation automatique de textes ont été abordés jusqu'à présent dans ce mémoire. Il a été question au premier chapitre des particularités du problème en tant que tel. On a vu en quoi consistait la catégorisation automatique de textes, quels traitements elle impliquait, et de quelle façon on réussissait jusqu'à présent à la résoudre. Au deuxième chapitre, on a jeté un regard sur la façon de construire un classifieur. On a pu constater la variété des techniques d'apprentissage pouvant amener une application informatique à classer des textes avec autonomie. Parallèlement, le chapitre trois a permis de faire un aperçu sur la langue arabe et ses particularités linguistiques, ainsi qu'une panoplie des travaux récents sur la catégorisation automatique de textes arabes.

À travers cet état de l'art, nous avons pu constater que la majorité des systèmes proposés dans ce domaine sont des systèmes basés sur la représentation *bag of words*. Quoique cette représentation semble efficace, mais cela n'empêche pas de dire qu'elle a des limites. Avec une telle représentation, on néglige complètement la sémantique du document et l'on suppose que les termes sont indépendants, évidemment cela est faux.
Par exemple dans le bigramme الذكاء الاصطناعي, le mot الاصطناعي dépend fortement du mot الذكاء pour construire ce concept, qui ne peut être décrit que par le bigramme (les deux mots successifs), et non pas par ses constituants pris séparément. Dans la majorité des cas, les mots simples ne peuvent pas bien décrire les concepts ce qui engendre une ambigüité sémantique. Ainsi, le concept réseau de neurones a une signification bien précise qui ne peut être décrite par les mots réseau ou neurone pris séparément. Donc il sera utile de penser à d'autres techniques de représentation des documents, et de trouver d'autres descripteurs plus informatifs que les mots dans le but d'améliorer les performances des systèmes de catégorisation.

À cet effet, l'objectif du ce chapitre est de présenter notre approche pour la représentation des documents en catégorisation automatique de textes arabes. Cette approche s'appuie sur l'utilisation des ngrammes et plus précisément les unigrames et les bigrammes.

Dans les sections suivantes de ce chapitre, nous présentons une description détaillée de cette approche, ainsi que les différentes étapes de notre étude.

4.2 Description de l'approche proposée

La modification du processus de catégorisation que nous proposons intervient au niveau de la représentation de documents, et plus précisément dans le choix de termes d'indexation.

Notre objectif est de chercher des termes plus informatifs que les mots, et plus discriminants. Évidemment, les bigrammes sont plus informatifs et moins ambigus que les mots simples (les unigrames). Avec deux mots successifs on peut bien décrire les concepts ce qui peut améliorer les performances des systèmes de catégorisation.

À cet effet, il sera utile d'utiliser les bigrammes comme des descripteurs dans la représentation des documents. Rappelons nous qu'un bigramme dans notre approche signifie une séquence de deux mots successifs, ou tout simplement une phrase composé de deux mots.

L'idée de base de notre approche est d'utiliser les bigrammes en addition des unigrames (mots simples), mais pas à la place des unigrames. C'est en quelque sorte une approche qui combine la représentation sac de mot avec la représentation par les phrases. Mais il est à noter que seuls les bigrammes les plus informatifs ou les bigrammes ayant un apport informationnel élevé sont utilisés. Pour les sélectionner, nous avons employé la méthode d'*information gain et DF (la fréquence documentaire)* comme techniques de sélection des termes. Cela signifie que les bigrammes que nous avons utilisés vont être fort probable des bons discriminateurs et moins probable pour être bruyantes.

4.3 Méthodologie de l'étude menée

La démarche générale que nous avons suivie s'articule sur trois phases principales : une phase de préparation des données, suivie d'une phase d'apprentissage puis une évaluation des performances. Notre contribution intervient dans la première phase qui peut être devisé dans notre cas en trois phases essentielles : prétraitement de corpus, extraction des bigrammes, et représentation des documents. Donc en résumé, cinq phases sont à distinguer dans notre étude:

- La phase de prétraitement du corpus.
- La phase d'extraction des bigrammes.
- La phase de représentation des documents.
- La phase de construction de classifieur.
- La phase d'évaluation des performances.

La première phase consiste à nettoyer le corpus textuel. Nous avons réalisé un prétraitement où des processus puissants de filtrage sont déclenchés. Ensuite un algorithme d'extraction des bigrammes est appliqué sur la totalité du corpus. Au cours de cette phase, nous effectuons différents traitements que nous détaillerons par la suite. Nous obtenons donc à la fin de cette étape un vocabulaire composé des bigrammes et des unigrames, qui sera utilisé par la suite pour indexer les documents. Le modèle de représentation choisi est le modèle vectoriel. Selon ce modèle un corpus textuel est représenté par une matrice, cette dernière sera transmise au classifieur afin de construire le modèle de prédiction sur la base d'un apprentissage supervisé. Il faut noter que nous avons utilisé le classifieur SVM qui est qualifié comme un des meilleurs classifieurs dans le domaine de la catégorisation de textes.
La figue 4-1 résume l'ensemble des traitements effectués pour implémenter notre approche.

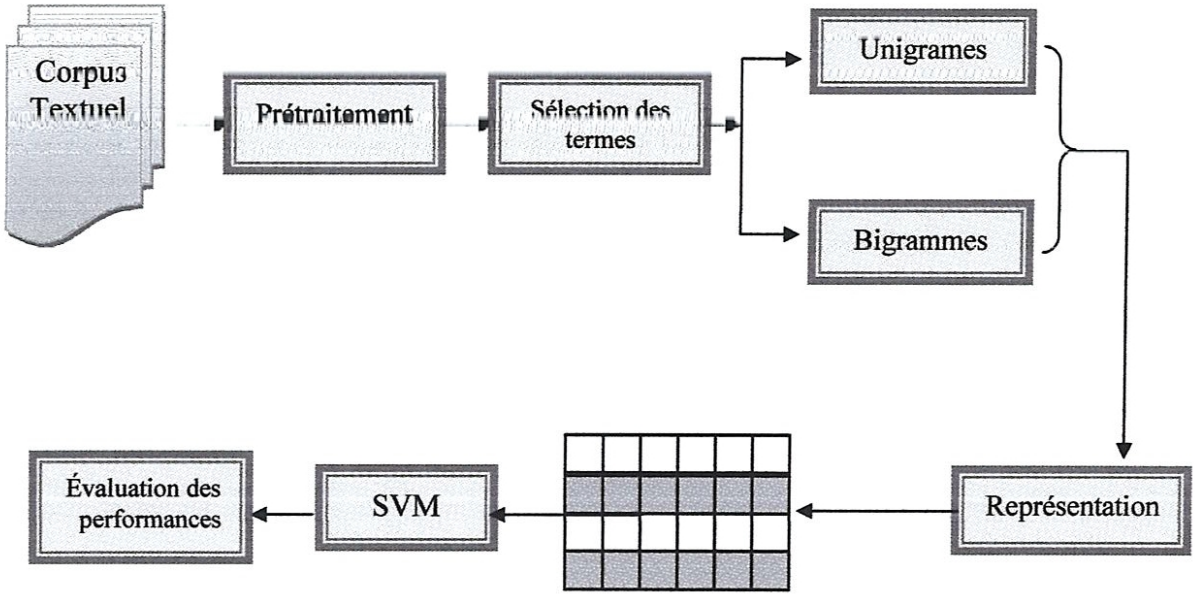

Figure 4-1: Chaine du traitement effectué

4.3.1 Le prétraitement

La première étape de cette phase consiste à convertir le corpus au format textuel. Ensuite chaque texte est nettoyé tout en éliminant les marques de ponctuation, les marques diacritiques, les chiffres et les différents caractères qui ne sont pas des lettres. Cette étape est appelée également l'étape de nettoyage. Une fois cette tâche réalisée, on passe directement à la phase de normalisation, son objectif est de maitriser la variation dans la façon de représentation des textes en arabe.

La normalisation consiste à :

- Remplacer le أ ou ء ou آ par ا.
- Remplacer la séquence ىء par ئ.
- Remplacer ى par ي.
- Remplacer ة par ه.

Enfin, le filtrage est le dernier processus dans cette phase. Son objectif est d'éliminer tous les mots qui ne participent pas activement au sens du document. (Les mots outils, les

conjonctions de subordination, les jours de la semaine ou les mois, les chiffres écrits en lettres)(Figure 4-2).

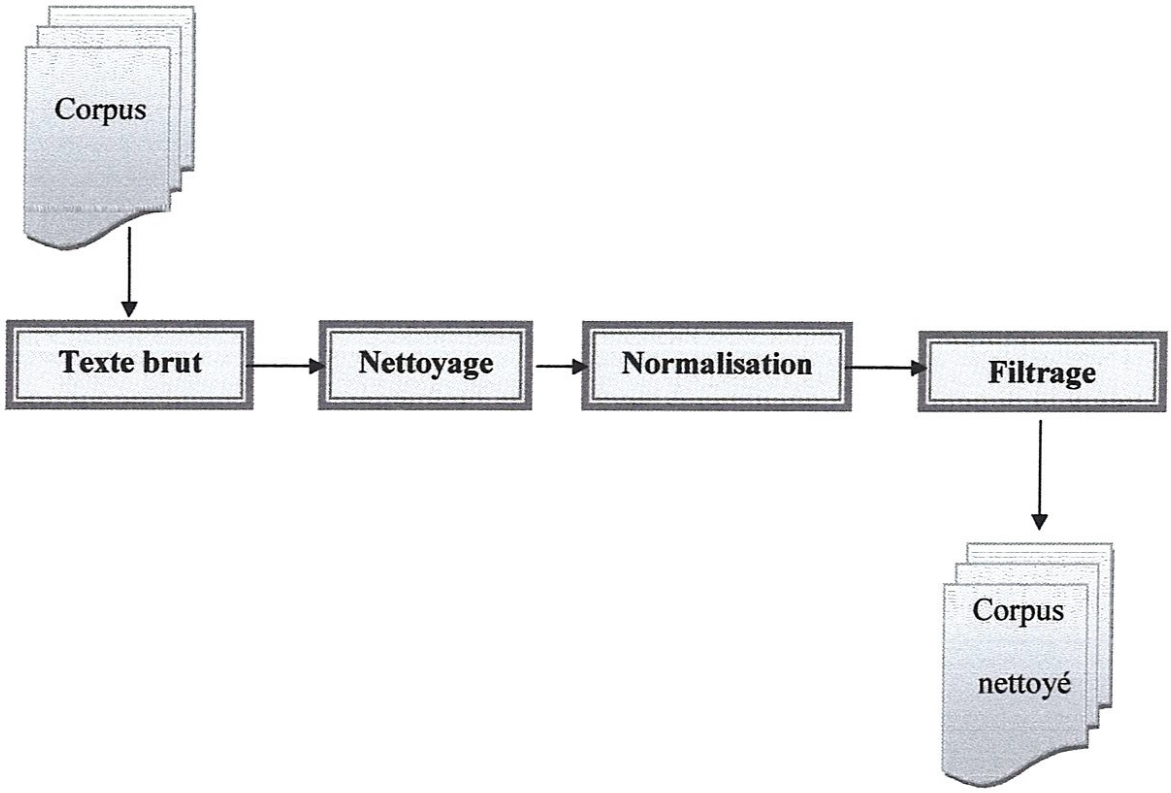

Figure 4-2:processus du prétraitement

4.3.2 Extraction des bigrammes

C'est la phase la plus critique dans notre étude. Elle consiste à trouver la liste des bigrammes les plus représentatifs et les plus informatifs pour chaque catégorie. Dans notre étude, L'extraction des bigrammes est faite en se basant sur l'algorithme décrit ci-dessous. Son idée de base est simple, elle consiste à trouver la liste des bigrammes dont l'apport informationnel est jugé élevé en plus leur fréquence documentaire doit être supérieur à un certain seuil fixé préalablement.

Intuitivement, ce que notre algorithme cherche à trouver dans un premier temps est la liste des unigrames les plus informatifs de chaque catégorie en employant la méthode

information gain décrit dans le premier chapitre pour la sélection des termes. La liste obtenue est considérée comme l'ensemble de base de termes.

La deuxième étape consiste à faire balayer tout les documents pour une deuxième fois, afin d'extraire les bigrammes dont les composants (les deux mots qui constituent le bigramme) figurent dans l'ensemble de base des termes. On obtient ainsi un ensemble de bigrammes candidats. Cet ensemble va subir un autre processus de filtrage, et seuls les bigrammes dont la fréquence documentaire est élevée seront sélectionnés, pour former le lexique des bigrammes, qui sera utilisé avec les mots simples pour former le vocabulaire d'indexation.

Le nombre total des bigrammes sélectionné ne dépasse pas 2 % du nombre des unigrames, de cette façon, on peut éviter le problème de la grande dimensionnalité de l'espace de représentation qui peut être posé.

Le pseudo code de notre algorithme est le suivant :

Algorithme2 Extraction des bigrammes

1. Trouver S= {liste des unigrames U, en employant infogain }
2. Soit B= ensemble vide
3. Pour chaque document d faire
4. {
5. Pour chaque deux mots adjacents (w1,w2) faire
6. {
7. Si (w1 ϵ S)et (w2 ϵ S) alors ajouter le bigramme w1+w2 à B
8. }
9. Pour chaque bigramme b dans B faire {
10. If (idf (b)< idf-bigramme) alors
11. Supprimer le bigramme b de l'ensemble B
12. }

Il faut noter que plusieurs travaux [MESLEH, 2007] [Yang, 1997] ont montré que *information gain* et *mutuel information* sont les meilleures méthodes de sélection des termes. La raison pour laquelle nous avons choisi d'utiliser *information gain* dans notre étude.

4.3.3 La représentation des documents

Le modèle de représentation que nous avons choisi est le modèle vectoriel. Selon ce modèle, chaque document est représenté par un vecteur de termes pondéré sachant qu'un terme peut être un mot simple (unigramme) comme il peut être un bigramme. Pour la pondération des termes nous avons utilisé la pondération TF x IDF qui combine la densité de terme dans le document multiplié par l'inverse de sa fréquence documentaire (idf). Ce dernier mesure le degré discriminatoire de termes (un terme présent dans tous les documents du corpus ne peut pas être discriminant).

4.3.4 La construction du classifieur

Pour construire notre classifieur, nous avons procédé par des techniques d'apprentissage supervisé qui produit des modèles de prédiction, qui une fois évalués et jugés acceptables par l'utilisateur, peuvent alors être utilisés comme un moyen automatique de catégorisation pour les nouveaux textes. Pour aboutir à un modèle de prédiction, le principe consiste à fournir à l'algorithme d'apprentissage des exemples de textes préclassés que nous appellerons ensemble d'apprentissage. Dans notre cas, c'est l'ensemble des vecteurs généré dans l'étape précédente. Le résultat d'un apprentissage est un modèle noté M et un taux d'erreur ε en généralisation estimée sur échantillon test ou par validation croisée.

Le classifieur que nous avons choisi d'utiliser pour implémenter notre approche est les machines à vecteurs support (SVM) décrit dans le deuxième chapitre. ce choix est motivé par la nette supériorité des SVM et sa capacité à traiter des espaces de données de grande dimensionnalité.

4.3.5 Évaluation des performances

Les mesures de performances standard utilisées dans notre étude sont la précision et le rappel, ainsi le f-score décrits au niveau du deuxième chapitre.

Ces mesures sont calculées à partir des données de test. Généralement, le corpus est découpé en deux parties, l'une servant à l'apprentissage et l'autre au test. Suivant le découpage effectué les résultats peuvent être significativement différents. Pour éviter ce problème, nous avons opté pour la méthode de validation croisée qui consiste à découper le corpus en « p » parties (p est fixé à 10 dans notre système). Le test se fait en 10 phases. A chaque phase « 9 » ensembles sont regroupés pour former la base d'apprentissage, et le classifieur est testé sur le dernier ensemble. Le taux de performances est calculé comme la moyenne des taux sur chaque essai.

4.4 Conclusion

Ce chapitre a pour objectif de présenter notre approche pour construire un système de catégorisation automatique de textes arabes.
Après avoir évoqué la problématique qu'on veut entamer, une description détaillée de l'approche proposée est faite. Cette approche qui tend à combiner les avantages liés à l'utilisation des bigrammes avec la représentation à base de mots.

Nous avons essayé à travers ce travail d'améliorer les approches de représentation des documents, tout en en se basant sur des descripteurs plus informatifs que les mots. Donc étudier l'impact des bigrammes avec les unigrames en matière d'indexation des documents arabes nous parait une piste intéressante à explorer.

Pour implémenter cette approche une méthodologie bien précise a été suivie. Nous avons présenté les différentes phases de notre étude tout en se focalisant sur la phase la plus critique dans notre approche, qui est la phase d'extraction des bigrammes les plus informatifs et les plus discriminants.
Les résultats obtenus ainsi les détails techniques de l'implémentation seront exposés dans le chapitre suivant.

Chapitre 5

Expérimentations et Résultats

Sommaire

5.1 Introduction

Afin de juger la pertinence de l'approche proposée au niveau du chapitre précédent, une série d'expérimentations a été menée sous un environnement d'apprentissage, et en utilisant un corpus textuel en langue arabe. L'objectif du présent chapitre est de détailler le processus d'implémentation ainsi les résultats obtenus. Tout d'abord, nous allons présenter le corpus en question ainsi que les différents outils utilisés, puis le processus d'expérimentation et la méthodologie d'évaluation mise en place. Et enfin les résultats obtenus, et des discussions des interprétations qu'on peut en faire.

5.2 Présentation du corpus

Un corpus est une grande collection de textes mémorisés sous forme électronique rassemblés selon un ensemble de *critères spécifiques* avec un *objectif* d'étude précis.

"Corpora is a collection of pieces of language that are selected and ordered according to explicit linguistic criteria in order to be used as a sample of the language".

« John Sinclair en 1996 »

Les corpus sont des ressources importantes aussi bien pour l'enseignement que pour la recherche. En ce qui concerne la langue arabe, la plus part des corpus sont payants ce qui complique la tâche devant les chercheurs arabes. Les corpus les plus élaborés sur le web sont édités par LDC (Language Data Consortium : Université de Pennsylvanie) et ELRA (Européen language ressources association). Les corpus les plus populaires sont ceux édités par l'AFP (Agence France Presse) sous forme de dépêches, et les archives des journaux quotidiens tels que journal El Hayat, qui sont connus pour la publication annuelle sous forme de CD ROM de textes archivés.

Certaines références des corpus les plus connus en langue arabe sont présentées dans le tableau 5-1.

Corpus	Source	Forme	Taille	Utilisation	Origine
Buckwalter Arabic Corpus 1986-2003	Tim Buckwalter	Written	2.5 to 3 M words	Lexicography	Public ressources on the web
Leuven Corpus (1990-2004)	Catholic University Leuven, Belgium	Written and Spoken	3 M words Spoken : 700000	Arabic_Dutch Dutch_arabic Learner's Dictionary	Internet sources radio, TV, primary school books
Arabic Newswire Corpus (1994)	University of Pennsylvania LDC	Written	80M words	Education and the developement of technology	Agence France Press, Xinhua new agency, and Umma Press.
CALLFRIEND corpus(1995)	University of Pennsylvania LDC	Convesati--onal	60 telephone conversation	Developement of language identification technology	Egyptian native speakers
NijmegenCorpus (1996)	Nijmegen university	Written	Over 2M words	Arabic-Dutch Dutch-arabic dictionary	Magazines and fiction
CALLHOME corpus (1997)	University of pennsylvania LDC	Conversa--tional	120 telephone conversation	Speech recognition produced from telephone lines	Egyptian native speakers
CLARA(1997)	Charles University, Prague	Written	50M Words	Lexicographic purposes	Periodicals, books, internet sources from 1975-present
Egypt (1999)	John Hopkins University	Written	Unknown	MT	A parallel corpus of the Quran in English and Arabic
Broadcast News Speech(2000)	University of Pennsylvania LDC	Spoken	More than 110 broadcast	Speech recognition	News broadcast from the radio of voice of America
DINAR corpus (2000)	Nijmegen Univ, sotetel, cooordination of Lyon2 Univ	Written	10Mwords	Lexicography, general research, NLP	Unknown

An-Nahar Corpus (2001)	ELRA	Written	140 M words	General resarch	An-Nahar Newspaper (Lebanon)
Arabic Gigaword (2002)	University of pennsylvania LDC	Written	Around 400 M words	Naturel language processing, information retrieval, lang uage modeling	Agence France press, al-Hayat news agency, an-Nahar news agency, xinhua news agency.
E-A Parallel Corpus (2003)	University of kuwait	Written	3M words	Teaching translation & lexicography	Publication from kuwait national Council
General scientific Arabic Corpus (2004)	UMIST, UK	Written	1.6 M words	Investigating Arabic Compounds	www.kisr.edu.kw/science
Classical Arabic Corpus(CAC) (2004)	UMIST, UK	Written	5 M words	Lexical analysis research	www.muhaddith.org www.alwarag.com
Multilingual corpus (2004)	UMIST, UK	Written	11.5M words (Arabic 2.5M)	Translation	IT-specialized websites- computer system and online software help-one book
SOTETEL corpus	SOTETEL-IT , Tunisia	Written	8M words	Lexicography	Literature, academic and journalistic material
Corpus of Contemporary Arabic (CCA) (2004)	University of Leeds	Written and spoken	Around 1Mwords	TAFL and information retrieval	Websites and online magazines
DARPA Babylon levantine arabic speech and transcripts (2005)	University of pensylvania LDC	Spoken	About 2000 telephone calls	Machine translation, speech recognition	Fisher style telephone speech collection

Tableau 5-1: Les corpus disponibles en langue arabe (source : Latifa Al sulaiti home page)

Pour effectuer nos expérimentations, nous nous sommes retournés vers la construction manuelle de corpus. Après une longue recherche sur Internet, il s'est avéré difficile à trouver des corpus en langue arabe avec une taille importante offerts gratuitement. Cette situation nous a incité à construire notre corpus manuellement en utilisant les archives des journaux arabes comme journal al-Hayat et la presse algérienne.

Le corpus construit comporte 1800 documents répartis dans 10 rubriques spécifiques, suivant les critères de répartition des sujets des journaux : rubrique général, rubrique actualités, rubrique tourisme, rubrique informatique, rubrique économie, rubrique politique, rubrique culture, rubrique sciences, rubrique religion et rubrique Sport (Tableau 5-2).

classe	Nombre de documents	Taille en Mo
Général	150	3.20
Actualité	140	3.08
Économie	170	3.83
Politique	210	4.06
Culture	150	3.33
Science	150	3.12
Informatique	180	3.75
Tourisme	220	4.23
religion	230	4.28
Sport	200	4.10
Total	1800	36.98

Tableau 5-2: structure du corpus d'apprentissage

5.3 Présentation de l'environnement d'apprentissage utilisé

RapidMiner est la version récente de l'environnement Yale (yet another learning environment), qui est un *java open source* pour solution *data mining.* Il a été conçu par l'équipe de l'intelligence artificielle de l'université de Dormund en 2001. Il est accueilli par *SourceForge* depuis 2004.

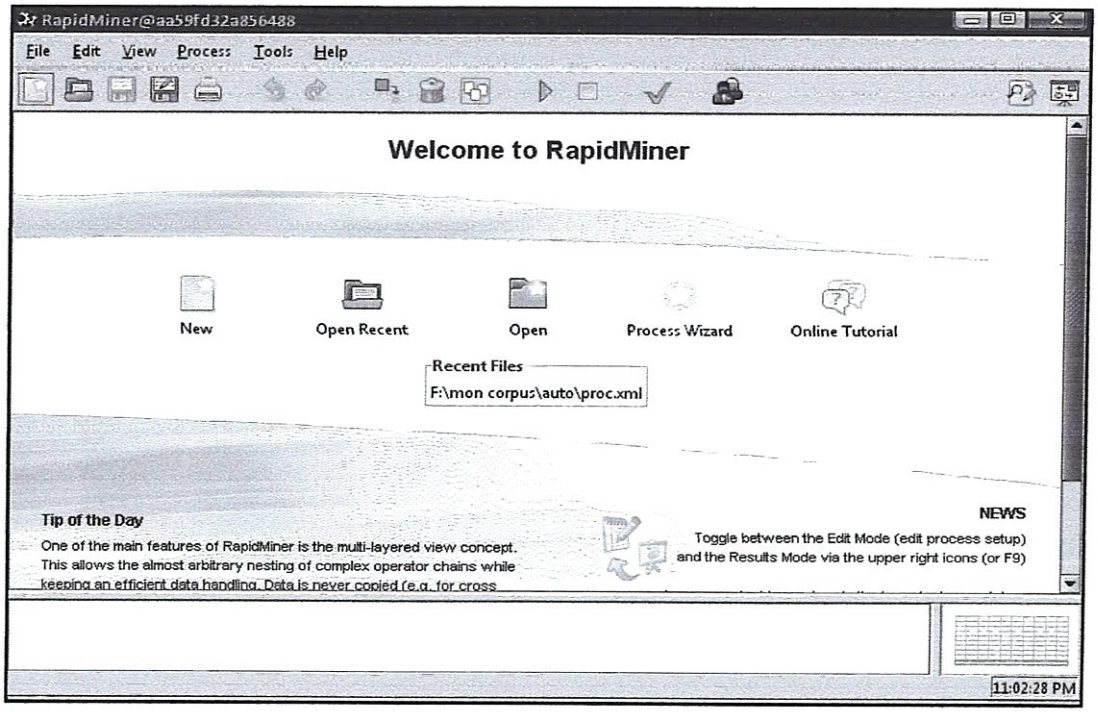

Figure 5-1:interface de l'environnement *RapidMiner*

RapidMiner fournit quatre modules (*plug-in*), et plus de 500 opérateurs nécessaires pour les différentes tâches de l'apprentissage automatique. Ces opérateurs sont regroupés en catégories hiérarchiques comme suit :

- Opérateurs des entrées/sortie (IO) : dédiés à la manipulation des attributs, la génération des exemples sources, la création des modèles et autre.
- Opérateurs d'apprentissage : supervisé et non supervisé.
- Opérateurs de prétraitement/POS-traitement : sélection, pondération, filtre.
- Opérateurs de validation : évaluation des performances, validation simple et croisée et beaucoup d'autres comme les opérateurs OLAP, WEB, META, visualisation.

Les *plug-in* de *RapidMiner* sont : text, value serie, data stream, CRF.

- Text *(WVTOOL: Word Vector Tool)* : utilisé pour créer une représentation vectorielle de textes fournis en entrée.
- Value Serie : offre les outils nécessaires pour l'extraction automatique d'attributs à partir des séries de données.

- Data Stream : offre les opérateurs nécessaires pour l'exploration des flux de données.
- CRF (*Conditional Random Field*).

5.4 Processus d'expérimentation

Le processus général de nos expérimentations s'articule sur quatre phases essentielles à savoir : la construction du vocabulaire d'indexation, la représentation vectorielle des documents, la construction du classifieur, et l'évaluation des performances.

5.4.1 La construction du vocabulaire d'indexation

Afin de construire ce vocabulaire, nous avons développé un outil en langage java, qui reçoit en entré un répertoire de textes sous format HTML et produit en sortie un vocabulaire composé des Unigrames et bigrammes. Cet outil intègre deux modules essentiels : module du prétraitement et module d'extraction des bigrammes.

5.4.1.1 Le module du prétraitement

S'occupe de différents prétraitements linguistiques de corpus (Élimination des marques de ponctuation, les marques diacritiques les chiffres, la normalisation des caractères, et la suppression des mots outils).

Le résultat obtenu à ce niveau est un répertoire de texte nettoyé et normalisé, mais n'est pas radicalisé.

Pour des fins de comparaison ultérieures, et afin d'évaluer l'impact de *stemming* sur les performances des systèmes de catégorisation avec l'approche proposée, nous avons décidé de créer un répertoire de textes radicalisés, ou nous avons employé *AL Stem* développé par Darwish comme outils de *stemming*.

5.4.1.2 Le module d'extraction des bigrammes

C'est l'implémentation de l'algorithme d'extraction des bigrammes décrit dans le chapitre précédent. A ce niveau nous avons employé la bibliothèque Wvtool.jar qui est une bibliothèque java. Dans son implémentation originale, cette bibliothèque est destinée pour manipuler des mots simples (Unigrame). Afin de l'adapter avec nos besoins, et de la rendre

capable de manipuler des bigrammes, certaines modifications ont été faites au niveau des classes principales de cette bibliothèque et plus particulièrement la classe VTTokenizer.java.

Pour sélectionner les termes les plus informatifs, nous avons utilisé les opérateurs *InfoGainWeighting* et *AttributeWeightSelection* de l'environnement *RapidMiner* (Figure 5-2)

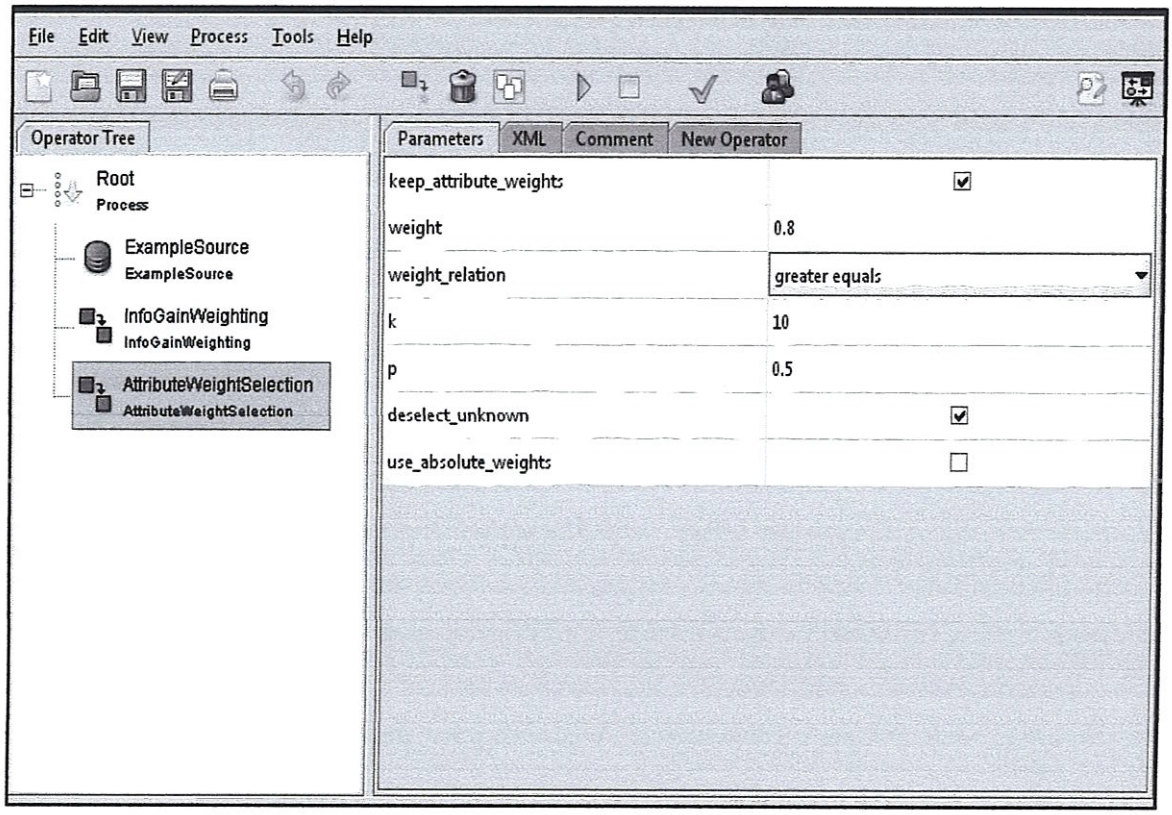

Figure 5-2: Sélection des termes

5.4.2. La représentation vectorielle des documents

À ce niveau, nous avons employé l'opérateur *WVTOOL* fourni par le *plugin Text* de l'environnement *RapidMiner* (figure 5-3).

Trois paramètres de cet opérateur ont été configurés :

- **Source** : spécifier le chemin d'accès de répertoire de texte à utiliser.
- **Type** : spécifier le type de textes (TXT, XML, HTML, PDF…).
- **Encodage** : UTF-16.

Trois Dataset sont générés comme résultats de cette étape :

- Dataset de l'approche proposée (sans stemming).
- Dataset Unigrame.
- Dataset de l'approche proposée (avec stemming).

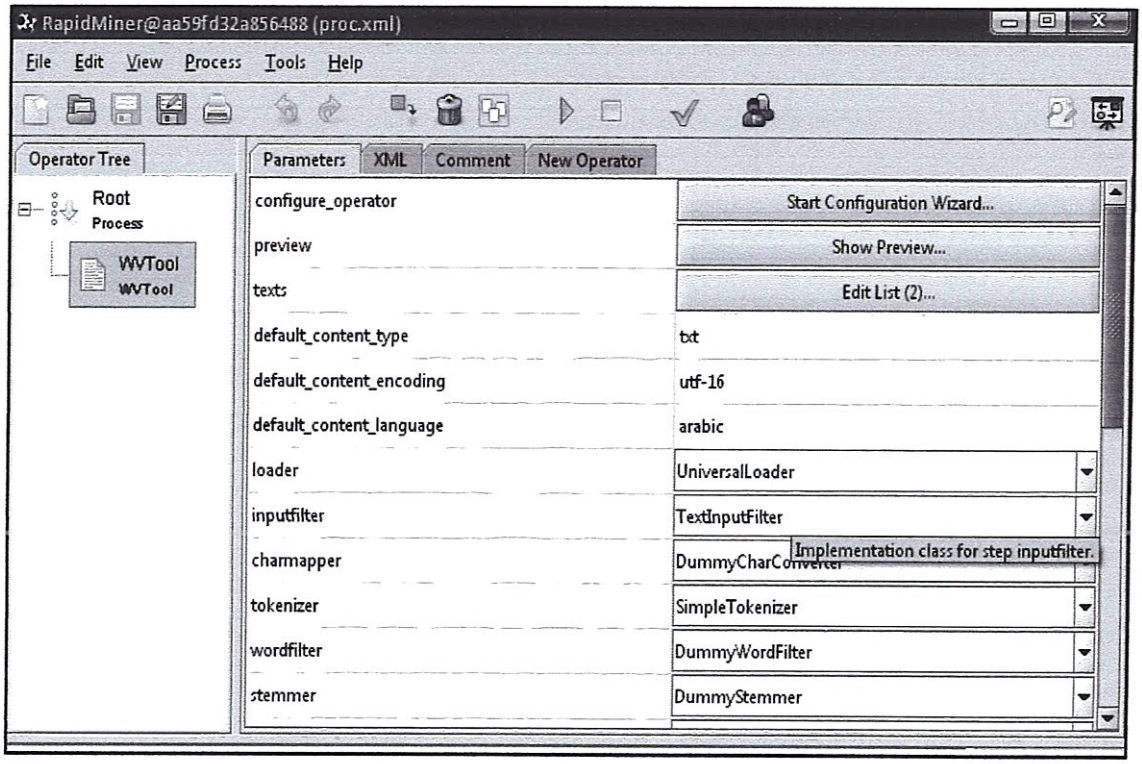

Figure 5-3: Génération des vecteurs TF-IDF

5.4.3 La construction du classificateur

Pour la construction du classificateur SVM, nous avons déroulé l'opérateur *LibSVMLearner* offert par l'environnement *RapidMiner* (Figure 5-4).

Quelques paramètres et opérateurs ont été configurés :

- *Exemplesource* : pour la sélection de Dataset utilisé, dans notre cas les trois Dataset générés à l'étape précédente.
- *SVM-type* : pour spécifier le type de SVM, s'agit-il d'une simple classification, régression, ou estimation de distribution.

➢ *Kernel-type* : pour spécifier le type de noyau à utiliser (linéaire, polynomiale,.. etc).

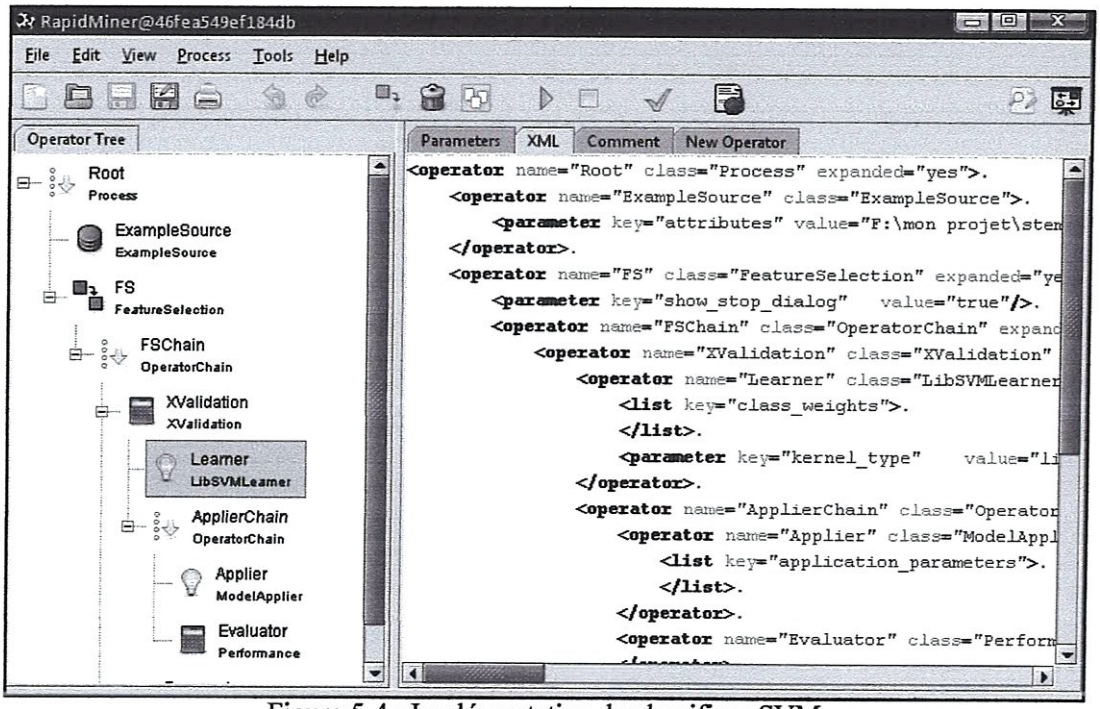

Figure 5-4 : Implémentation du classifieur SVM

Comme les SVM sont un classifieur binaire, nous avons été obligés de construire autant de classifieurs que de catégories. Cela signifie qu'on a envisagé à construire dix (10) classifieurs pour les dix catégories du corpus, tel que chaque classifieur est testé sur les trois bases (dataset) générées à l'étape précédente.

5.4.4 Évaluation des performances de l'approche proposée

Les mesures de performance utilisées pour évaluer notre approche sont la précision et le rappel et le F-score. Étant donné que les performances de l'approche proposée sont comparées avec les performances d'une approche à base de mots (unigrames).

Les résultats obtenus sont présentés dans le tableau 5-3.

Catégorie	Approche	Nbre d'attributs	Rappel	Précision	F -score
général	Unigrames + bigrammes	16046	93,56	78,15	85,16
	Unigrames	15728	88,89	72,59	79,92
actualité	Unigrames + bigrammes	10200	79,15	89,07	83,82
	Unigrames	10000	77,13	83,25	80,07
économie	Unigrames + bigrammes	16585	93,62	98,18	95,40
	Unigrames	16260	89,27	91,58	90,41
politique	Unigrames + bigrammes	14640	97,89	95,89	96,88
	Unigrames	14353	93,09	85,24	88,99
culture	Unigrames + bigrammes	14945	90,07	96,04	92,96
	Unigrames	14652	86,48	93,62	89,91
science	Unigrames + bigrammes	13304	94,22	88,68	91,37
	Unigrames	13043	88,45	73,12	80,06
informatique	Unigrames + bigrammes	10772	98,78	84,79	91,25
	Unigrames	10561	96,3	76,58	85,32
tourisme	Unigrames + bigrammes	11792	91,15	89,17	90,15
	Unigrames	11561	83,45	81,22	82,32
religion	Unigrames + bigrammes	12750	99,85	98,09	98,96
	Unigrames	12500	96,12	92,67	94,36
sport	Unigrames + bigrammes	9209	91,12	93,02	92,06
	Unigrames	9028	89,23	91,23	90,22

Tableau 5-3: performances de l'approche proposée.

- **Moyennes des mesures de performances**

approche	**Macro-rappel**	**Macro-précision**	**Macro F-score**
Unigrames +bigrammes	92,94	91,10	91,85
Unigrames	88,84	84,11	86,16

Tableau 5-4: moyenne des mesures de performances.

1. rappel

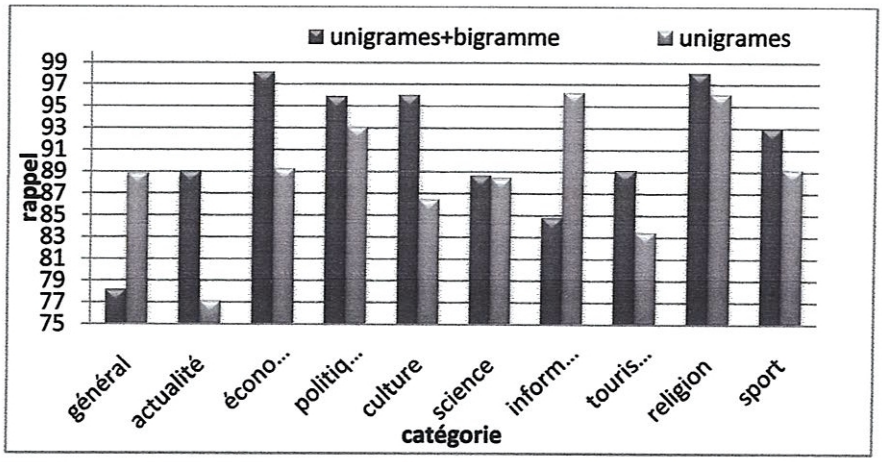

Figure 5-5: le rappel pour les deux approches

2. précision

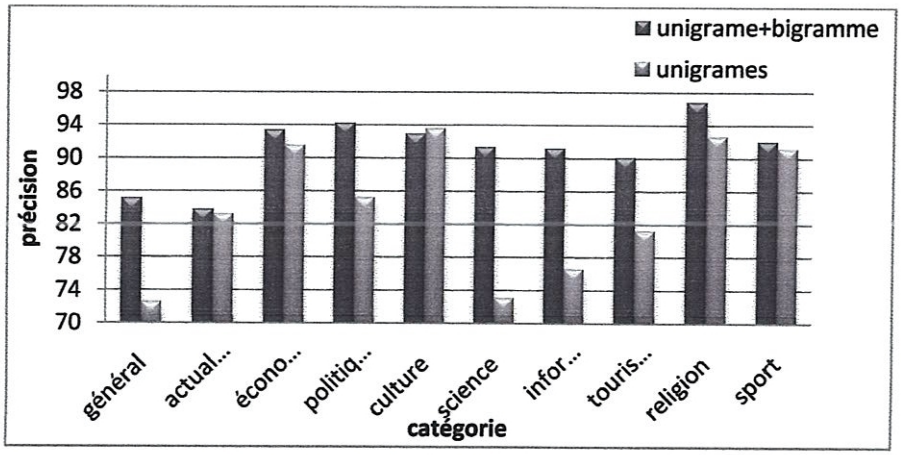

Figure 5-6: la précision pour les deux approches

3. F-score

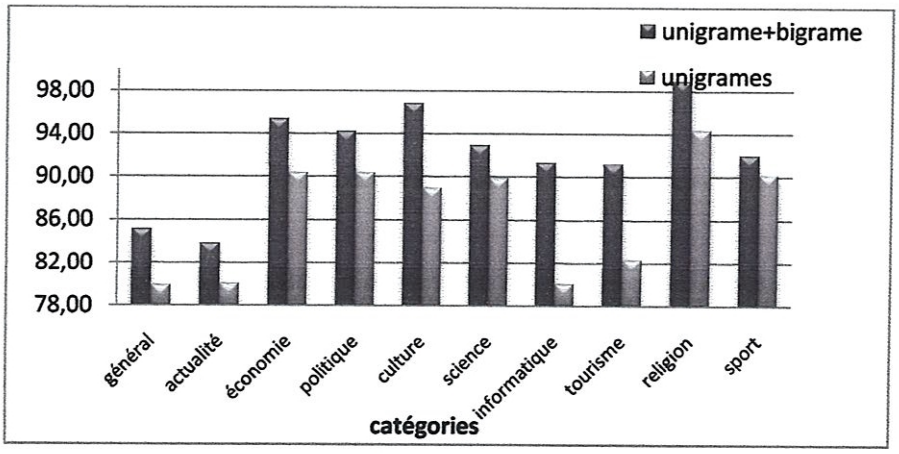

Figure 5-7: le F-score pour les deux approches

- **Discussion**

D'après les résultats obtenus (Tableau 5-3) (Tableau 5-4), on peut constater que les performances de l'approche proposée sont meilleures par rapport à l'approche à base de mots. Une amélioration en termes des trois mesures de performances est évidemment bien constatée.

- En terme du rappel, ce gain est l'ordre de 4.1 %.
- Pour la précision, il est de l'ordre de 7 %.
- Enfin, pour le F-score moyen, ce gain est de l'ordre de 5.69 %.

Il faut signaler que le taux de cette amélioration n'est pas identique pour toutes les catégories, on remarque que pour la catégorie sport, on obtient une amélioration en terme de F-score de l'ordre de 2 %, cependant pour la catégorie science l'amélioration est de l'ordre 11 %, donc l'ajout des bigrammes pour certaines catégories est plus bénéfique que pour les autres. Remarquons que les catégories science, informatique et tourisme ont bien bénéficié de l'ajout de bigrammes que les autres catégories.

Nous pensons que cela est dû essentiellement au nombre de concepts qui doit être décrit par les bigrammes dans la catégorie. Pour quelques catégories il n y pas assez de concepts à décrire par les bigrammes, les mots peuvent décrire presque tous les termes de domaine sans ambiguïté et dans ce cas l'ajout des bigrammes ne sera pas avec un grand intérêt.

Remarquons aussi que l'ajout des bigrammes est plus bénéfique pour l'amélioration de la précision que pour le rappel. Les bigrammes permettent la désambigüisation des termes d'indexation, cela va conduire à une amélioration de la précision globale du système. Ce qui confirme notre intuition.

Donc en résumé, nous pouvons dire que, suite au gain apporté en terme des trois mesures de performance, nous pouvons confirmer et affirmer que l'utilisation des bigrammes avec les mots simples (unigrames) comme des descripteurs pour la représentation des documents conduit a une amélioration des performances en catégorisation automatique de textes arabes.

5.4.5 Impact du *stemming* sur les performances avec l'approche proposée

Afin d'évaluer l'influence du stemming sur les performances en catégorisation automatique, nous avons décidé d'effectuer les mêmes expérimentations sur des textes radicalisés, et de comparer les résultats avec ceux obtenus en utilisant des textes à l'état brut (sans stemming). Les résultats obtenus sont présentés dans le tableau 5-5.

Catégorie	Approche proposée	Nbre d'attributs	Rappel	Précision	F- score
Général	Sans stemming	16046	93,56	78,15	85,16
	Avec stemming	6579	88,75	76,45	82,14
Actualité	Sans stemming	10200	79,15	89,07	83,82
	Avec stemming	4488	77,13	83,25	80,07
Economie	Sans stemming	16585	93,62	98,18	95,40
	Avec stemming	6468	92,56	91,45	92,00
Politique	Sans stemming	14640	97,89	95,89	96,88
	Avec stemming	5856	96,17	87,96	91,88
Culture	Sans stemming	14945	90,07	96,04	92,96
	Avec stemming	6426	87,66	93,23	90,36
Science	Sans stemming	13304	94,22	88,68	91,37
	Avec stemming	5188	89,44	81,78	85,44
Informatique	Sans stemming	10772	98,78	84,79	91,25
	Avec stemming	4308	97,03	74,36	84,20
Tourisme	Sans stemming	11792	91,15	89,17	90,15
	Avec stemming	4952	89,77	86,95	88,34
Religion	Sans stemming	12750	99,85	98,09	98,96
	Avec stemming	4717	97,21	89,03	92,94
Sport	Sans stemming	9209	91,12	93,02	92,06
	Avec stemming	4056	92,84	89,25	91,01

Tableau 5-5: Influence du *stemming* sur les performances avec l'approche proposée

- **Moyennes des mesures de performances**

Approche proposée	Macro-rappel	Macro-précision	Macro f-mesure
Sans stemming	92,94	91.10	91,85
Avec stemming	91,15	85,42	88,01

Tableau 5-6: moyenne des mesures de performances

- **Discussion**

Ce qu'on peut constater d'après le Tableau 5-5 et le tableau 5-6, est l'influence du stemming sur la qualité des résultats. On remarque que le stemming conduit à une baisse des performances de l'ordre de 1.79 %, 5.68 % et 3.84 % pour les trois mesures (rappel, précision et F-score) par rapport à ceux obtenus en utilisant des mots à l'état brut (sans stemming). Cependant, il conduit à une compression considérable de la taille du vocabulaire d'indexation ainsi du temps du calcul, ce qui laisse à penser que le stemming est un outil plutôt efficace pour la réduction de dimension que pour l'amélioration des performances en catégorisation automatique de textes arabes.

Théoriquement on dit que le stemming est un prétraitement qui permet de réduire le nombre de termes d'indexation, et donc le temps de calcul et d'améliorer les performances en terme de rappel. Nous pensons que cette deuxième fonction de stemming est superflue, en raison des résultats obtenus. Le stemming permet de mettre une correspondance entre des mots de la même famille, comme il peut mettre en relation des mots qui ne devraient pas l'être, c.-à-d des mots ayant la même famille lexicale, mais de signification complètement différente. Nous pensons que l'omission des voyelles en langue arabe contribue d'une façon non négligeable à ce niveau du problème.

Si l'utilisation de stemming est assez controversée, elle semble globalement permettre une amélioration des performances en recherche d'information [Larkey, 2005] [Larkey, 2002] mais nous pensons qu'elle est inutile en catégorisation automatique.

Enfin, il est à noter que les résultats obtenus dépend aussi du type de classifieur ainsi de la qualité des termes d'indexation. Plusieurs travaux ont montré l'efficacité des SVM et sa capacité à traiter des espaces de données de grande dimensionnalité, ce constat est prouvé aussi par les résultats de nos expérimentations sur des documents en langue arabe.

5.4 Conclusion

Dans ce chapitre, nous avons fixé comme objectif de présenter les différentes expérimentations effectuées, pour évaluer la validité de l'approche proposée au niveau de chapitre précédent.
Pour cela, et avant de rentrer dans les détails de l'implémentation, nous avons présenté le corpus sur lequel nous avons effectué nos expérimentations, ainsi que l'environnement d'apprentissage utilisé, et les différents outils mis à notre disposition.

Puis nous avons procédé à l'évaluation de l'approche proposée. En premier lieu, nous avons comparé ses performances avec l'approche à base de mots. D'après les résultats obtenus, nous avons pu mettre en évidence l'influence des bigrammes et leur apport informationnel en matière d'indexation des documents sur les performances des systèmes de catégorisation automatique de textes arabes.

En second lieu, nous avons étudié l'impact du stemming en catégorisation automatique de textes. Suite à l'examen des résultats obtenus, on peut juger la pertinence de ce prétraitement pour la réduction de la taille du vocabulaire d'indexation, mais son emploi pour l'amélioration des performances en catégorisation automatique de textes arabe, peut être néfaste qu'avantageux, ce qui est prouvé aussi par les travaux de [MESLEH, 2007].

Conclusion et perspectives

Au terme de ce mémoire, nous avons évoqué la problématique de la catégorisation automatique de textes en langue arabe. Nous avons fixé comme objectif de tester l'impact des bigrammes en matière d'indexation des documents, et de confirmer que son emploi avec les unigrames peut améliorer les performances des systèmes de catégorisation de textes arabes. Ces objectifs ont été atteints finalement.

Durant cette étude, nous avons pu remarquer que la phase la plus critique dans la construction d'un système de catégorisation automatique, réside dans la phase du prétraitement. Quoique cette phase nous sembla insignifiante au début, mais il apparait ensuite qu'elle est primordiale. Le choix de termes d'indexation (mots simples, stem, ngramme), le stemming, ainsi la sélection des termes les plus pertinents sont des tâches sur lesquelles se basent les performances d'un système de catégorisation.

Quant à l'approche proposée, les résultats obtenus ont permis de démontrer que la mise en place d'un tel mécanisme pouvait donner lieu à une amélioration des performances. Certes, les hausses des performances observées au cours de nos expérimentations ne sont pas négligeables, au contraire, il faut savoir que, dans le domaine de la catégorisation automatique de textes, de telles améliorations sont en général très bien reçues.

Une autre conclusion à tirer, concerne l'influence de stemming sur les performances des systèmes de catégorisation automatique de textes. Les résultats obtenus ont permis de démontrer que le stemming quoiqu'il apporte une réduction importante de la taille du vocabulaire d'indexation, il conduit en parallèle à une baisse des performances, ce qui laisse à penser que le stemming est un outil plutôt efficace pour la réduction de dimension que pour l'amélioration des performances en catégorisation automatique, et que son emploi dans ce domaine pour améliorer les performances peut être plus néfaste qu'avantageux.

Cette conclusion a été également mise en évidence aussi dans les travaux de Leopold en utilisant les SVM [Leopold, 2002], qui a prouvé que le stemming est plutôt utile en recherche d'information, mais inutile en catégorisation automatique, ce qui confirme l'analyse que nous avons effectuée.

Nos travaux comportent évidemment certaines limites ouvrant la voie à d'autres avenues de recherche. Les multiples variables intervenant dans le processus de catégorisation créent un nombre quasi infini de configurations de paramètres à tester. Malheureusement, le temps, lui, n'est pas infini, et il a été nécessaire de fixer certains paramètres pour en étudier d'autres plus en profondeur. Évidemment, il aurait été intéressant d'observer le comportement de notre approche sur d'autres corpus de textes de taille plus grande et avec d'autres classifieurs.

De ce fait, l'une de nos perspectives futures serait de tester la validité de cette approche en utilisant les MSVM (multiple support Victor machine), qui est la version multiclasse des SVM.

Il serait intéressant aussi de penser à combiner plusieurs classifieurs. Une hybridation des méthodes permet de combiner les avantages de l'apprentissage non supervisé pour préitéqueter les données, et de l'apprentissage supervisé pour généraliser les résultats et mieux classer les nouveaux documents peut donner des résultats intéressants.

En ce qui concerne la représentation des documents, une multitude de travaux ont déjà porté sur la traditionnelle représentation *« sac de mots »*. Quoi qu'il en soit, il est toujours légitime de tenter de développer de nouvelles façons pour représenter les documents.

En un mot, la catégorisation automatique de textes est un domaine loin d'en être à ses débuts, mais qui présente encore plusieurs défis. Il s'agit d'une technologie ayant le potentiel de soutenir des applications très utiles et intéressantes, mais démontrant certaines lacunes qui doivent être résolues, parmi lesquelles : le coût de constitution des corpus textuels. Il est intéressant de chercher des solutions alternatives à l'usage d'ensembles d'entrainement volumineux et coûteux à construire.

Dans un même ordre d'idée, il est intéressant de penser à construire des systèmes de catégorisation à base d'une ontologie de domaine. Cette dernière va jouer le rôle du classifieur. De cette façon, aucun algorithme d'apprentissage n'est employé, et l'usage d'un corpus textuel n'est plus nécessaire parce qu'il n'y' aura pas une phase d'apprentissage, de plus la classification d'un document devient beaucoup plus basée sur la sémantique que sur les statistiques.

Corpus	Source	Forme	Taille	Utilisation	Origine
Buckwalter Arabic Corpus 1986-2003	Tim Buckwalter	Written	2.5 to 3 M words	Lexicography	Public ressources on the web
Leuven Corpus (1990-2004)	Catholic University Leuven, Belgium	Written and Spoken	3 M words Spoken : 700000	Arabic_Dutch Dutch_arabic Learner's Dictionary	Internet sources radio, TV, primary school books
Arabic Newswire Corpus (1994)	University of Pennsylvania LDC	Written	80M words	Education and the developement of technology	Agence France Press, Xinhua new agency, and Umma Press.
CALLFRIEND corpus(1995)	University of Pennsylvania LDC	Convesati--onal	60 telephone conversation	Developement of language identification technology	Egyptian native speakers
NijmegenCorpus (1996)	Nijmegen university	Written	Over 2M words	Arabic-Dutch Dutch-arabic dictionary	Magazines and fiction
CALLHOME corpus (1997)	University of pennsylvania LDC	Conversa--tional	120 telephone conversation	Speech recognition produced from telephone lines	Egyptian native speakers
CLARA(1997)	Charles University, Prague	Written	50M Words	Lexicographic purposes	Periodicals, books, internet sources from 1975-present
Egypt (1999)	John Hopkins University	Written	Unknown	MT	A parallel corpus of the Quran in English and Arabic
Broadcast News Speech(2000)	University of Pennsylvania LDC	Spoken	More than 110 broadcast	Speech recognition	News broadcast from the radio of voice of America
DINAR corpus (2000)	Nijmegen Univ, sotetel, cooordination of Lyon2 Univ	Written	10Mwords	Lexicography, general research, NLP	Unknown

An Nahar Corpus (2001)	ELRA	Written	140 M words	General resarch	An Nahar Newspaper (Lebanon)
Arabic Gigaword (2002)	University of pennsylvania LDC	Written	Around 400 M words	Naturel language processing, information retrieval, lang uage modeling	Agence France press, al Hayat news agency, an-Nahar news agency, xinhua news agency.
E-A Parallel Corpus (2003)	University of kuwait	Written	3M words	Teaching translation & lexicography	Publication from kuwait national Council
General scientific Arabic Corpus (2004)	UMIST, UK	Written	1.6 M words	Investigating Arabic Compounds	www.kisr.edu.kw/science
Classical Arabic Corpus(CAC) (2004)	UMIST, UK	Written	5 M words	Lexical analysis research	www.muhaddith.org www.alwarag.com
Multilingual corpus (2004)	UMIST, UK	Written	11.5M words (Arabic 2.5M)	Translation	IT-specialized websites- computer system and online software help-one book
SOTETEL corpus	SOTETEL-IT , Tunisia	Written	8M words	Lexicography	Literature, academic and journalistic material
Corpus of Contemporary Arabic (CCA) (2004)	University of Leeds	Written and spoken	Around 1Mwords	TAFL and information retrieval	Websites and online magazines
DARPA Babylon levantine arabic speech and transcripts (2005)	University of pensylvania LDC	Spoken	About 2000 telephone calls	Machine translation, speech recognition	Fisher style telephone speech collection

Tableau 5-1: Les corpus disponibles en langue arabe (source : Latifa Al sulaiti home page)

Pour effectuer nos expérimentations, nous nous sommes retournés vers la construction manuelle de corpus. Après une longue recherche sur Internet, il s'est avéré difficile à trouver des corpus en langue arabe avec une taille importante offerts gratuitement. Cette situation nous a incité à construire notre corpus manuellement en utilisant les archives des journaux arabes comme journal al-Hayat et la presse algérienne.

Le corpus construit comporte 1800 documents répartis dans 10 rubriques spécifiques, suivant les critères de répartition des sujets des journaux : rubrique général, rubrique actualités, rubrique tourisme, rubrique informatique, rubrique économie, rubrique politique, rubrique culture, rubrique sciences, rubrique religion et rubrique Sport (Tableau 5-2).

classe	Nombre de documents	Taille en Mo
Général	150	3.20
Actualité	140	3.08
Économie	170	3.83
Politique	210	4.06
Culture	150	3.33
Science	150	3.12
Informatique	180	3.75
Tourisme	220	4.23
religion	230	4.28
Sport	200	4.10
Total	1800	36.98

Tableau 5-2: structure du corpus d'apprentissage

5.3 Présentation de l'environnement d'apprentissage utilisé

RapidMiner est la version récente de l'environnement Yale (yet another learning environment), qui est un *java open source* pour solution *data mining.* Il a été conçu par l'équipe de l'intelligence artificielle de l'université de Dormund en 2001. Il est accueilli par *SourceForge* depuis 2004.

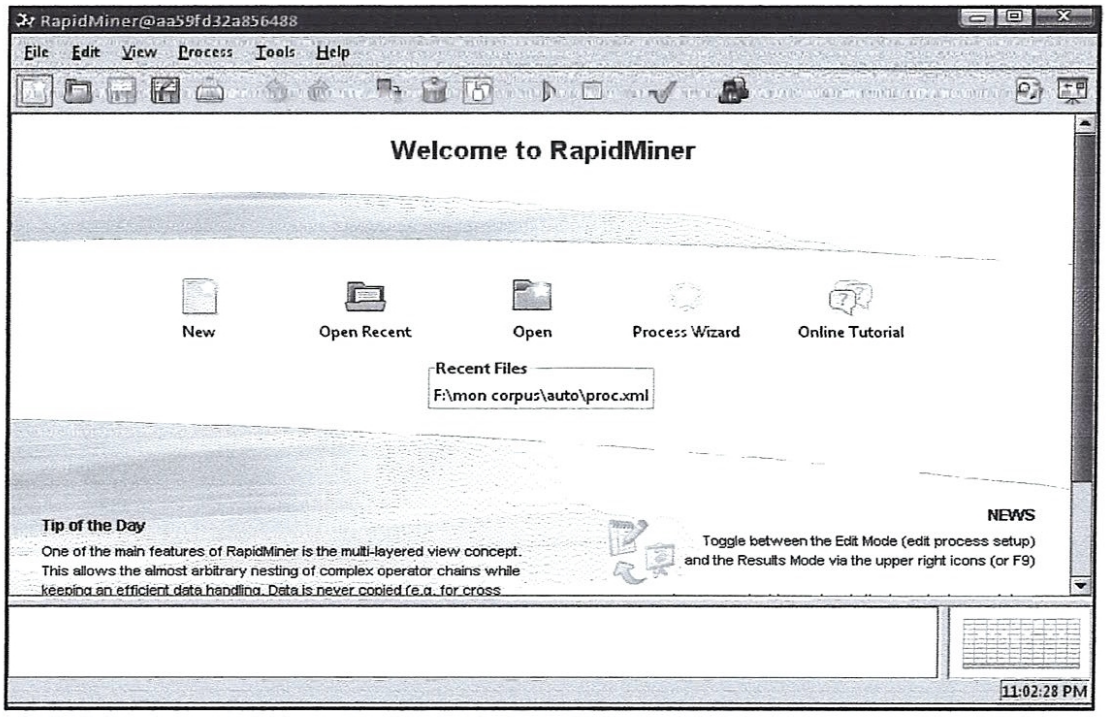

Figure 5-1:interface de l'environnement *RapidMiner*

RapidMiner fournit quatre modules (*plug-in*), et plus de 500 opérateurs nécessaires pour les différentes tâches de l'apprentissage automatique. Ces opérateurs sont regroupés en catégories hiérarchiques comme suit :

- Opérateurs des entrées/sortie (IO) : dédiés à la manipulation des attributs, la génération des exemples sources, la création des modèles et autre.
- Opérateurs d'apprentissage : supervisé et non supervisé.
- Opérateurs de prétraitement/POS-traitement : sélection, pondération, filtre.
- Opérateurs de validation : évaluation des performances, validation simple et croisée et beaucoup d'autres comme les opérateurs OLAP, WEB, META, visualisation.

Les *plug-in* de *RapidMiner* sont : text, value serie, data stream, CRF.

- Text *(WVTOOL: Word Vector Tool)* : utilisé pour créer une représentation vectorielle de textes fournis en entrée.
- Value Serie : offre les outils nécessaires pour l'extraction automatique d'attributs à partir des séries de données.

- Data Stream : offre les opérateurs nécessaires pour l'exploration des flux de données.
- CRF (*Conditional Random Field*).

5.4 Processus d'expérimentation

Le processus général de nos expérimentations s'articule sur quatre phases essentielles à savoir : la construction du vocabulaire d'indexation, la représentation vectorielle des documents, la construction du classifieur, et l'évaluation des performances.

5.4.1 La construction du vocabulaire d'indexation

Afin de construire ce vocabulaire, nous avons développé un outil en langage java, qui reçoit en entré un répertoire de textes sous format HTML et produit en sortie un vocabulaire composé des Unigrames et bigrammes. Cet outil intègre deux modules essentiels : module du prétraitement et module d'extraction des bigrammes.

5.4.1.1 Le module du prétraitement

S'occupe de différents prétraitements linguistiques de corpus (Élimination des marques de ponctuation, les marques diacritiques les chiffres, la normalisation des caractères, et la suppression des mots outils).

Le résultat obtenu à ce niveau est un répertoire de texte nettoyé et normalisé, mais n'est pas radicalisé.

Pour des fins de comparaison ultérieures, et afin d'évaluer l'impact de *stemming* sur les performances des systèmes de catégorisation avec l'approche proposée, nous avons décidé de créer un répertoire de textes radicalisés, ou nous avons employé *AL Stem* développé par Darwish comme outils de *stemming*.

5.4.1.2 Le module d'extraction des bigrammes

C'est l'implémentation de l'algorithme d'extraction des bigrammes décrit dans le chapitre précédent. A ce niveau nous avons employé la bibliothèque Wvtool.jar qui est une bibliothèque java. Dans son implémentation originale, cette bibliothèque est destinée pour manipuler des mots simples (Unigrame). Afin de l'adapter avec nos besoins, et de la rendre

capable de manipuler des bigrammes, certaines modifications ont été faites au niveau des classes principales de cette bibliothèque et plus particulièrement la classe VTTokenizer.java.

Pour sélectionner les termes les plus informatifs, nous avons utilisé les opérateurs *InfoGainWeighting* et *AttributeWeightSelection* de l'environnement *RapidMiner* (Figure 5-2)

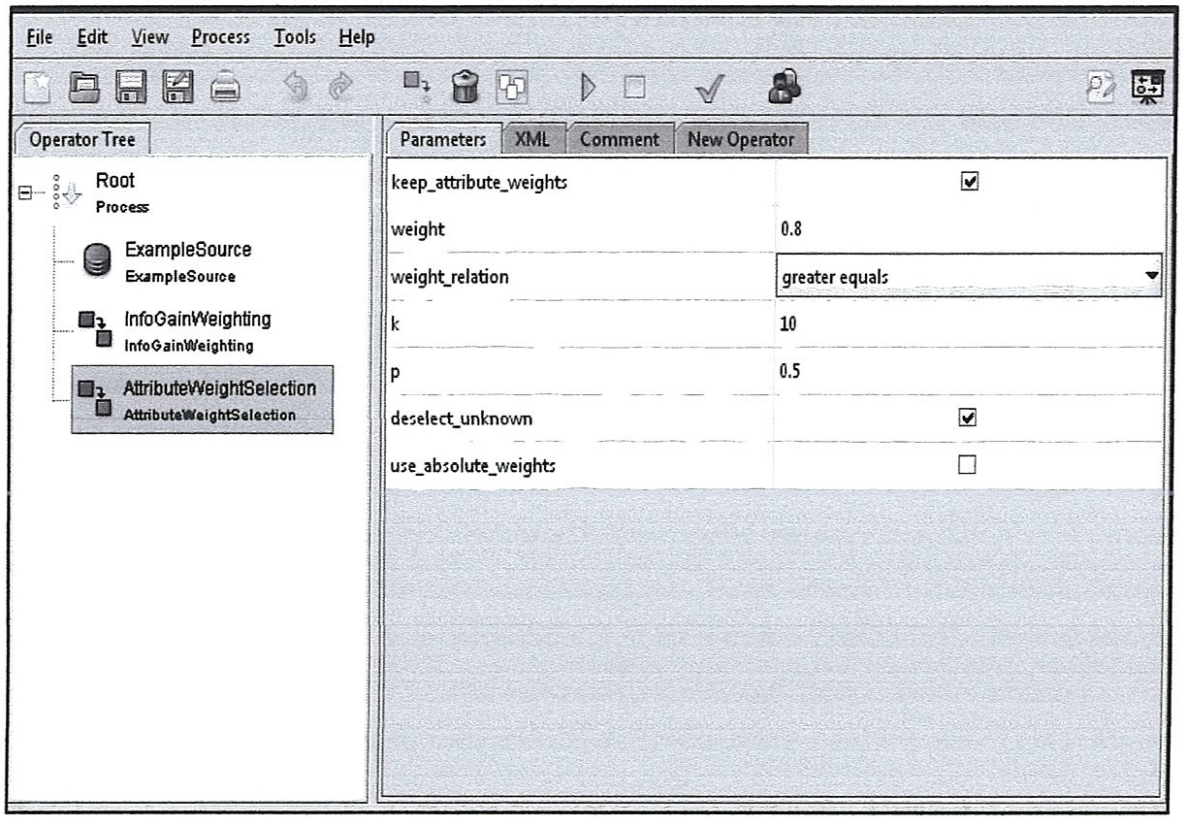

Figure 5-2: Sélection des termes

5.4.2. La représentation vectorielle des documents

À ce niveau, nous avons employé l'opérateur *WVTOOL* fourni par le *plugin Text* de l'environnement *RapidMiner* (figure 5-3).

Trois paramètres de cet opérateur ont été configurés :

- **Source** : spécifier le chemin d'accès de répertoire de texte à utiliser.
- **Type** : spécifier le type de textes (TXT, XML, HTML, PDF…).
- **Encodage** : UTF-16.

Trois Dataset sont générés comme résultats de cette étape :

- Dataset de l'approche proposée (sans stemming).
- Dataset Unigrame.
- Dataset de l'approche proposée (avec stemming).

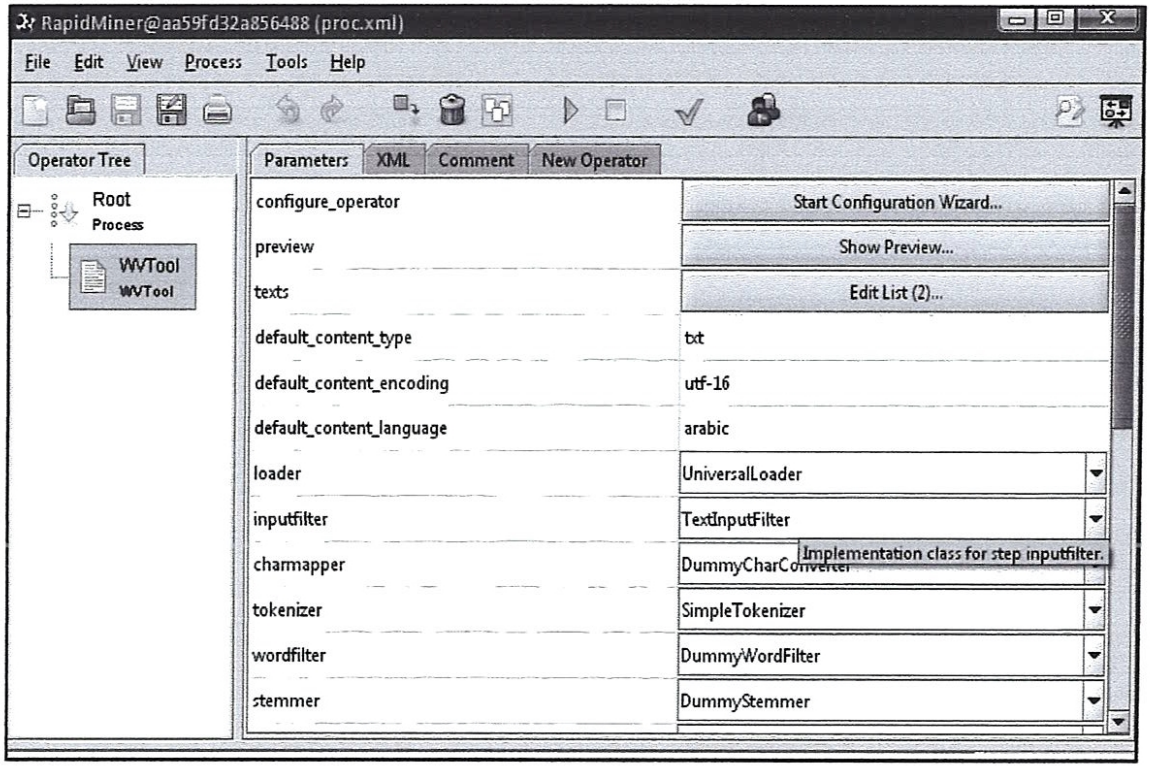

Figure 5-3: Génération des vecteurs TF-IDF

5.4.3 La construction du classificateur

Pour la construction du classificateur SVM, nous avons déroulé l'opérateur *LibSVMLearner* offert par l'environnement *RapidMiner* (Figure 5-4).

Quelques paramètres et opérateurs ont été configurés :

- *Exemplesource* : pour la sélection de Dataset utilisé, dans notre cas les trois Dataset générés à l'étape précédente.
- *SVM-type* : pour spécifier le type de SVM, s'agit-il d'une simple classification, régression, ou estimation de distribution.

➢ *Kernel-type* : pour spécifier le type de noyau à utiliser (linéaire, polynomiale,.. etc).

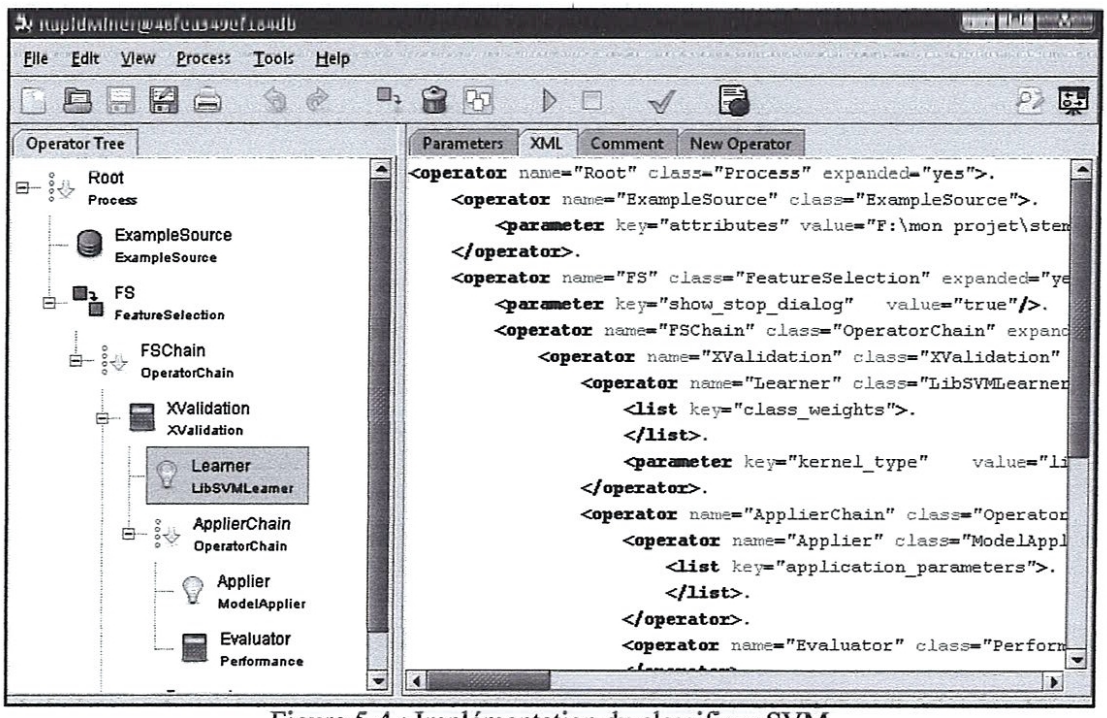

Figure 5-4 : Implémentation du classifieur SVM

Comme les SVM sont un classifieur binaire, nous avons été obligés de construire autant de classifieurs que de catégories. Cela signifie qu'on a envisagé à construire dix (10) classifieurs pour les dix catégories du corpus, tel que chaque classifieur est testé sur les trois bases (dataset) générées à l'étape précédente.

5.4.4 Évaluation des performances de l'approche proposée

Les mesures de performance utilisées pour évaluer notre approche sont la précision et le rappel et le F-score. Étant donné que les performances de l'approche proposée sont comparées avec les performances d'une approche à base de mots (unigrames).

Les résultats obtenus sont présentés dans le tableau 5-3.

Catégorie	Approche	Nbre d'attributs	Rappel	Précision	F -score
général	Unigrames + bigrammes	16046	93,56	78,15	85,16
	Unigrames	15728	88,89	72,59	79,92
actualité	Unigrames + bigrammes	10200	79,15	89,07	83,82
	Unigrames	10000	77,13	83,25	80,07
économie	Unigrames + bigrammes	16585	93,62	98,18	95,40
	Unigrames	16260	89,27	91,58	90,41
politique	Unigrames + bigrammes	14640	97,89	95,89	96,88
	Unigrames	14353	93,09	85,24	88,99
culture	Unigrames + bigrammes	14945	90,07	96,04	92,96
	Unigrames	14652	86,48	93,62	89,91
science	Unigrames + bigrammes	13304	94,22	88,68	91,37
	Unigrames	13043	88,45	73,12	80,06
informatique	Unigrames + bigrammes	10772	98,78	84,79	91,25
	Unigrames	10561	96,3	76,58	85,32
tourisme	Unigrames + bigrammes	11792	91,15	89,17	90,15
	Unigrames	11561	83,45	81,22	82,32
religion	Unigrames + bigrammes	12750	99,85	98,09	98,96
	Unigrames	12500	96,12	92,67	94,36
sport	Unigrames + bigrammes	9209	91,12	93,02	92,06
	Unigrames	9028	89,23	91,23	90,22

Tableau 5-3: performances de l'approche proposée.

- **Moyennes des mesures de performances**

approche	**Macro-rappel**	**Macro-précision**	**Macro F-score**
Unigrames +bigrammes	92,94	91,10	91,85
Unigrames	88,84	84,11	86,16

Tableau 5-4: moyenne des mesures de performances.

1. rappel

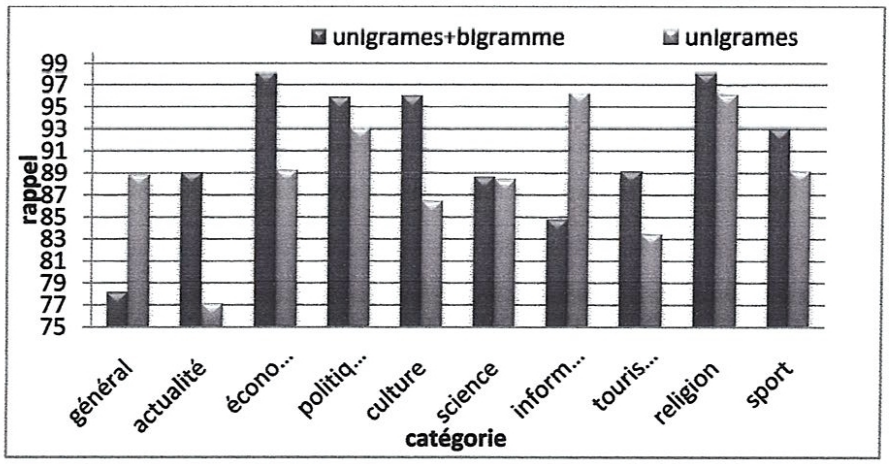

Figure 5-5: le rappel pour les deux approches

2. précision

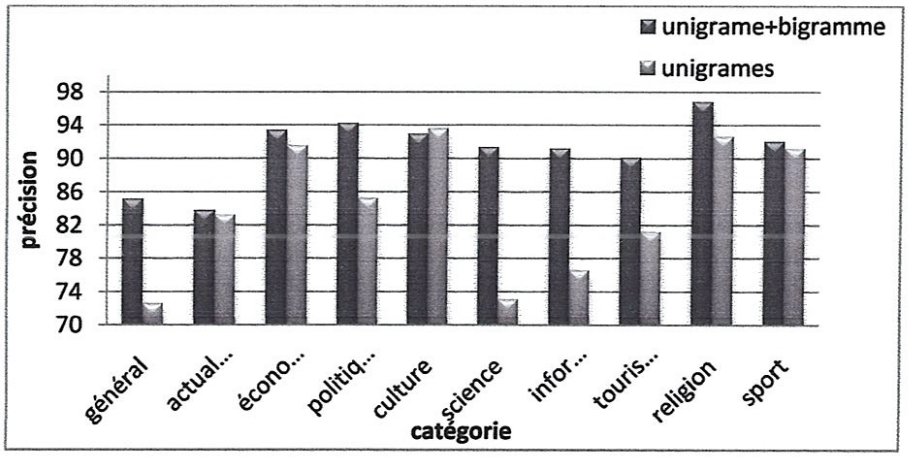

Figure 5-6: la précision pour les deux approches

3. F-score

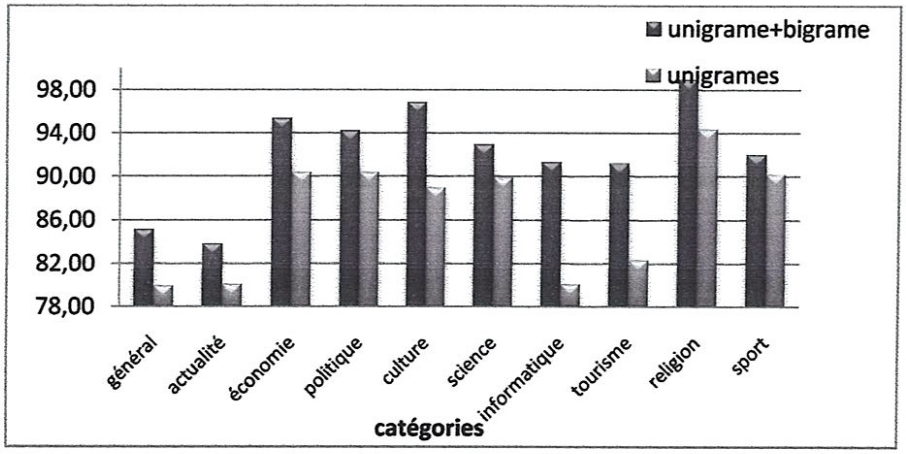

Figure 5-7: le F-score pour les deux approches

- **Discussion**

D'après les résultats obtenus (Tableau 5-3) (Tableau 5-4), on peut constater que les performances de l'approche proposée sont meilleures par rapport à l'approche à base de mots. Une amélioration en termes des trois mesures de performances est évidemment bien constatée.

- En terme du rappel, ce gain est l'ordre de 4.1 %.
- Pour la précision, il est de l'ordre de 7 %.
- Enfin, pour le F-score moyen, ce gain est de l'ordre de 5.69 %.

Il faut signaler que le taux de cette amélioration n'est pas identique pour toutes les catégories, on remarque que pour la catégorie sport, on obtient une amélioration en terme de F-score de l'ordre de 2 %, cependant pour la catégorie science l'amélioration est de l'ordre 11 %, donc l'ajout des bigrammes pour certaines catégories est plus bénéfique que pour les autres. Remarquons que les catégories science, informatique et tourisme ont bien bénéficié de l'ajout de bigrammes que les autres catégories.
Nous pensons que cela est dû essentiellement au nombre de concepts qui doit être décrit par les bigrammes dans la catégorie. Pour quelques catégories il n y pas assez de concepts à décrire par les bigrammes, les mots peuvent décrire presque tous les termes de domaine sans ambiguïté et dans ce cas l'ajout des bigrammes ne sera pas avec un grand intérêt.

Remarquons aussi que l'ajout des bigrammes est plus bénéfique pour l'amélioration de la précision que pour le rappel. Les bigrammes permettent la désambigüisation des termes d'indexation, cela va conduire à une amélioration de la précision globale du système. Ce qui confirme notre intuition.

Donc en résumé, nous pouvons dire que, suite au gain apporté en terme des trois mesures de performance, nous pouvons confirmer et affirmer que l'utilisation des bigrammes avec les mots simples (unigrames) comme des descripteurs pour la représentation des documents conduit a une amélioration des performances en catégorisation automatique de textes arabes.

5.4.5 Impact du *stemming* sur les performances avec l'approche proposée

Afin d'évaluer l'influence du stemming sur les performances en catégorisation automatique, nous avons décidé d'effectuer les mêmes expérimentations sur des textes radicalisés, et de comparer les résultats avec ceux obtenus en utilisant des textes à l'état brut (sans stemming). Les résultats obtenus sont présentés dans le tableau 5-5.

Catégorie	Approche proposée	Nbre d'attributs	Rappel	Précision	F- score
Général	Sans stemming	16046	93,56	78,15	85,16
	Avec stemming	6579	88,75	76,45	82,14
Actualité	Sans stemming	10200	79,15	89,07	83,82
	Avec stemming	4488	77,13	83,25	80,07
Economie	Sans stemming	16585	93,62	98,18	95,40
	Avec stemming	6468	92,56	91,45	92,00
Politique	Sans stemming	14640	97,89	95,89	96,88
	Avec stemming	5856	96,17	87,96	91,88
Culture	Sans stemming	14945	90,07	96,04	92,96
	Avec stemming	6426	87,66	93,23	90,36
Science	Sans stemming	13304	94,22	88,68	91,37
	Avec stemming	5188	89,44	81,78	85,44
Informatique	Sans stemming	10772	98,78	84,79	91,25
	Avec stemming	4308	97,03	74,36	84,20
Tourisme	Sans stemming	11792	91,15	89,17	90,15
	Avec stemming	4952	89,77	86,95	88,34
Religion	Sans stemming	12750	99,85	98,09	98,96
	Avec stemming	4717	97,21	89,03	92,94
Sport	Sans stemming	9209	91,12	93,02	92,06
	Avec stemming	4056	92,84	89,25	91,01

Tableau 5-5: Influence du *stemming* sur les performances avec l'approche proposée

- **Moyennes des mesures de performances**

Approche proposée	Macro-rappel	Macro-précision	Macro f-mesure
Sans stemming	92,94	91.10	91,85
Avec stemming	91,15	85,42	88,01

Tableau 5-6: moyenne des mesures de performances

- **Discussion**

Ce qu'on peut constater d'après le Tableau 5-5 et le tableau 5-6, est l'influence du stemming sur la qualité des résultats. On remarque que le stemming conduit à une baisse des performances de l'ordre de 1.79 %, 5.68 % et 3.84 % pour les trois mesures (rappel, précision et F-score) par rapport à ceux obtenus en utilisant des mots à l'état brut (sans stemming). Cependant, il conduit à une compression considérable de la taille du vocabulaire d'indexation ainsi du temps du calcul, ce qui laisse à penser que le stemming est un outil plutôt efficace pour la réduction de dimension que pour l'amélioration des performances en catégorisation automatique de textes arabes.

Théoriquement on dit que le stemming est un prétraitement qui permet de réduire le nombre de termes d'indexation, et donc le temps de calcul et d'améliorer les performances en terme de rappel. Nous pensons que cette deuxième fonction de stemming est superflue, en raison des résultats obtenus. Le stemming permet de mettre une correspondance entre des mots de la même famille, comme il peut mettre en relation des mots qui ne devraient pas l'être, c.-à-d des mots ayant la même famille lexicale, mais de signification complètement différente. Nous pensons que l'omission des voyelles en langue arabe contribue d'une façon non négligeable à ce niveau du problème.

Si l'utilisation de stemming est assez controversée, elle semble globalement permettre une amélioration des performances en recherche d'information [Larkey, 2005] [Larkey, 2002] mais nous pensons qu'elle est inutile en catégorisation automatique.

Enfin, il est à noter que les résultats obtenus dépend aussi du type de classifieur ainsi de la qualité des termes d'indexation. Plusieurs travaux ont montré l'efficacité des SVM et sa capacité à traiter des espaces de données de grande dimensionnalité, ce constat est prouvé aussi par les résultats de nos expérimentations sur des documents en langue arabe.

5.4 Conclusion

Dans ce chapitre, nous avons fixé comme objectif de présenter les différentes expérimentations effectuées, pour évaluer la validité de l'approche proposée au niveau de chapitre précédent.
Pour cela, et avant de rentrer dans les détails de l'implémentation, nous avons présenté le corpus sur lequel nous avons effectué nos expérimentations, ainsi que l'environnement d'apprentissage utilisé, et les différents outils mis à notre disposition.

Puis nous avons procédé à l'évaluation de l'approche proposée. En premier lieu, nous avons comparé ses performances avec l'approche à base de mots. D'après les résultats obtenus, nous avons pu mettre en évidence l'influence des bigrammes et leur apport informationnel en matière d'indexation des documents sur les performances des systèmes de catégorisation automatique de textes arabes.

En second lieu, nous avons étudié l'impact du stemming en catégorisation automatique de textes. Suite à l'examen des résultats obtenus, on peut juger la pertinence de ce prétraitement pour la réduction de la taille du vocabulaire d'indexation, mais son emploi pour l'amélioration des performances en catégorisation automatique de textes arabe, peut être néfaste qu'avantageux, ce qui est prouvé aussi par les travaux de [MESLEH, 2007].

Conclusion et perspectives

Au terme de ce mémoire, nous avons évoqué la problématique de la catégorisation automatique de textes en langue arabe. Nous avons fixé comme objectif de tester l'impact des bigrammes en matière d'indexation des documents, et de confirmer que son emploi avec les unigrames peut améliorer les performances des systèmes de catégorisation de textes arabes. Ces objectifs ont été atteints finalement.

Durant cette étude, nous avons pu remarquer que la phase la plus critique dans la construction d'un système de catégorisation automatique, réside dans la phase du prétraitement. Quoique cette phase nous sembla insignifiante au début, mais il apparait ensuite qu'elle est primordiale. Le choix de termes d'indexation (mots simples, stem, ngramme), le stemming, ainsi la sélection des termes les plus pertinents sont des tâches sur lesquelles se basent les performances d'un système de catégorisation.

Quant à l'approche proposée, les résultats obtenus ont permis de démontrer que la mise en place d'un tel mécanisme pouvait donner lieu à une amélioration des performances. Certes, les hausses des performances observées au cours de nos expérimentations ne sont pas négligeables, au contraire, il faut savoir que, dans le domaine de la catégorisation automatique de textes, de telles améliorations sont en général très bien reçues.

Une autre conclusion à tirer, concerne l'influence de stemming sur les performances des systèmes de catégorisation automatique de textes. Les résultats obtenus ont permis de démontrer que le stemming quoiqu'il apporte une réduction importante de la taille du vocabulaire d'indexation, il conduit en parallèle à une baisse des performances, ce qui laisse à penser que le stemming est un outil plutôt efficace pour la réduction de dimension que pour l'amélioration des performances en catégorisation automatique, et que son emploi dans ce domaine pour améliorer les performances peut être plus néfaste qu'avantageux.

Cette conclusion a été également mise en évidence aussi dans les travaux de Leopold en utilisant les SVM [Leopold, 2002], qui a prouvé que le stemming est plutôt utile en recherche d'information, mais inutile en catégorisation automatique, ce qui confirme l'analyse que nous avons effectuée.

Nos travaux comportent évidemment certaines limites ouvrant la voie à d'autres avenues de recherche. Les multiples variables intervenant dans le processus de catégorisation créent un nombre quasi infini de configurations de paramètres à tester. Malheureusement, le temps, lui, n'est pas infini, et il a été nécessaire de fixer certains paramètres pour en étudier d'autres plus en profondeur. Évidemment, il aurait été intéressant d'observer le comportement de notre approche sur d'autres corpus de textes de taille plus grande et avec d'autres classifieurs.

De ce fait, l'une de nos perspectives futures serait de tester la validité de cette approche en utilisant les MSVM (multiple support Victor machine), qui est la version multiclasse des SVM.

Il serait intéressant aussi de penser à combiner plusieurs classifieurs. Une hybridation des méthodes permet de combiner les avantages de l'apprentissage non supervisé pour préitéqueter les données, et de l'apprentissage supervisé pour généraliser les résultats et mieux classer les nouveaux documents peut donner des résultats intéressants.

En ce qui concerne la représentation des documents, une multitude de travaux ont déjà porté sur la traditionnelle représentation *« sac de mots »*. Quoi qu'il en soit, il est toujours légitime de tenter de développer de nouvelles façons pour représenter les documents.

En un mot, la catégorisation automatique de textes est un domaine loin d'en être à ses débuts, mais qui présente encore plusieurs défis. Il s'agit d'une technologie ayant le potentiel de soutenir des applications très utiles et intéressantes, mais démontrant certaines lacunes qui doivent être résolues, parmi lesquelles : le coût de constitution des corpus textuels. Il est intéressant de chercher des solutions alternatives à l'usage d'ensembles d'entrainement volumineux et coûteux à construire.

Dans un même ordre d'idée, il est intéressant de penser à construire des systèmes de catégorisation à base d'une ontologie de domaine. Cette dernière va jouer le rôle du classifieur. De cette façon, aucun algorithme d'apprentissage n'est employé, et l'usage d'un corpus textuel n'est plus nécessaire parce qu'il n'y' aura pas une phase d'apprentissage, de plus la classification d'un document devient beaucoup plus basée sur la sémantique que sur les statistiques.

Bibliographie

[Aljlayl, 2002] M.Aljlayl, O. Frieder. **On Arabic Search: Improving the Retrieval Effectiveness via a Light Stemming Approach**, *In 11th International Conference on Information and Knowledge Management (CIKM), pp.340-347, Virginia (USA), (2002).*

[Androutsopulos, 2000] L.Androutsopoulos, J.Koutsias, K.V. Chandrinos. **An experimental comparison of naive Bayesian and keywordbased anti-spam filtering with personal e-mail messages.** *Belkin, N. J., Ingwersen, P, et Leong, M.-K., editors, Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval, PP. 160.167, Athens, GR. ACM Press, New York, US(2000).*

[Aas, 1999] K. Aas, L.Eikvil. **Text categorization a survey.** *Technical report, Norwegian Computing Center (1999).*

[Apté, 1998] C. Apté, F. Damerau, S. M. Weiss. **Text mining with decision rules and decision trees.** *Proceedings of the Conference on Automated Learning and Discovery,Workshop 6: Learning from Text and the Web* (1998).

[Aha, 1997] D.Aha. **Lazy Learning.** *Kluwer Academie Publisher, from artificial intelligence review edition. Volume 11(1-5) (1997).*

[Alkharashi, 1994] J.Al-kharashi, M.W.Evens. **Comparring words stem and root as index terms in an Arabic information retrival system.** *Journal of the American Society for information Science (JASIS) 45(8),USA(1994).*

[Angeline, 1993] P.J.Angeline, J. B. Pollack. **Hierarchical RAAMs. A Uniform Modular Architecture.** *Technical report, Ohio State University, USA (1993).*

[Bawaneh, 2008] J.Bawaneh, M.S. Alkoffash, A.I.Al-Rabea. **Arabic Text Classification using K-NN and Naive Bayes.** *Journal of Computer Science 4 (7): 600-605,ISSN 1549-3636 (2008).*

[Bong, 2005]C.Bong, T.Wong. **an examination of feature selection framework in text categorization.** *Second Asia information retrieval symposium, AIRS 2005, Jeju Island, Korea, October 13-15 proceedings(2005).*

[Boulkanadel, 2005] S.Boulkanadel. **utilisation des syntagmes nominaux dans un systeme de recherché d'information en langue arabe.** *LINA FRE CNRS 2729, Université de Nantes 2. France (2005).*

[Baldi, 2003] P.Baldi, P.Frasconi , P.Smith. **modeling the internet and the web : probalistic methode and algorithme.** *ISBN :0-470-84906-1,USA(2003).*

[Baoli, 2003] L.Baoli, Y.Shiwen, L.Qin. **An omproved k-nearest neighbor algorithm for text categorization.***Proceedings of the 20th International Conference on Computer Processing of Oriental Languages, China(2003).*

[Baloul, 2002] S. Baloul, M. Alissali, M. Baudry, P. Boula. **Interface syntaxe-prosodie dans un système de synthèse de la parole à partir du texte en arabe.** *24es Journées d'Étude sur la Parole, 24-27 Nancy, pp.329-332(2002).*

[Buckwalter, 2002] T.Bucwalter.**Buckwalter Morphological Analyser Version 1.0.** http://www.qamus.org (2002).

[Biskri, 2001] I.Biskri, S.Delisle. **Les n-grams de caractères pour l'extraction de connaissance dans des bases de donnée textuelles multilingues.** *In TALN. Pages 93102 (2001).*

[Bodo, 1999] D.Bodo. **A Re-Unification of Two Competing Models for Document Retrieval.** *Journal of the American Society for Information Science, 50(1):49-64 (1999).*

[Burges, 1998] C.Burges. **A tutorial on support vector machine for pattern recognition.** *Data Mining and Knowledge Discovery, 2(2): 121-167 (1998).*

[Cavnar, 1994] W.Cavnar, J.Trenkle. **N-gram-based text categorization.** *Proceedings of SDAIR-94, 3rd Annual Symposium on document Analysis and Information Retrieval, PP. 161.175, Las Vegas,US (1994).*

[Caropreso, 2001] M.Caropreso, S.Matwin, F.Sebastiani. **A learner independent evaluation of the usefulness of statistical phrases for automated text categorization.** *In Chin, A. G., editor, Text Databases and Document Management: Theory and Practice, PP. 78.102. Idea Group Publishing, Hershey, US (2001).*

[Cohen, 1996] A. Chen, F. Gey .**Building an Arabic Stemmer for Information Retrieval.** *Proceedings of the Eleventh Text Retrieval Conference (TREC 2002). National Institute of Standards and Technology, Nov 18-22, 2002, PP. 631-640(1996).*

[Cohen, 2002] S. Cohen, Y. Kanza, Y. Kogan, Y. Sagiv. **a search and query language for XML.** *Journal of the American Society for Information Science, 53(6):454-466 (2002).*

[Cover, 1967] T.M.Cover, P.E.Hart. **Nearest neighbour pattern classification.** *IEEE Transactions on Information Theory, 13(1):1179-1184 (1967).*

[Cooley, 1999] **R.**Cooley. **Classification of news stories using Support vector Machines.** *IJCAI'99 Workshop on Text Mining, Stockholm, Sweden (1999).*

[Cooper, 1993] W.Cooper, A. Chen, F. C. Gey. **Full Text Retrieval based on Probalistic Equations with Coefficients by Logistic Regression.** *In D. K. Harman, editor, NIST Special Publication 500-215 : The Second Text REtrieval Conference (TREC-2), pages*

57{66, aithersburg, MD, 1993}. Department of Commerce, National Institute of Standards and. (1993).

[Djelullu, 2008] K.Djelullu, H.F.Merouani. **les machines à vecteurs support dans la catégorisation de textes arabes.** *Université 8 mai 1945 Guelma (2008).*

[Diab, 2004] M.Diab, K.Hacioglu,D.Jurafsky. **Automatic tagging of Arabic Text.** *From Raw Text to Base Phrase Chunks. Stanford University,USA. (2004).*

[Denoyer, 2004] L.Denoyer. **Apprentissage et inférence statistique dans les bases de documents structurés : application aux corpus de document textuels.** *Thèse de doctorat, université paris 6, France (2004).*

[Douzidia, 2004] F.S.Douzidia. **Résumé automatique de textes arabes.** *Université de Montéral, Canada(2004)*

[Darwish, 2002] K. Darwish. **Building a Shallow Arabic Morphological Analyzer in One Day.** *Proceedings of the workshop on Computational Approaches to Semitic Languages in the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02), Philadelphia, PA, USA. PP. 47-54.(2002).*

[Débili, 2002] F.Débili, H.Achour, F.Souici. **La langue arabe et l'ordinateur : de l'étiquetage grammatical à la voyellation automatique.** *Correspondances de l'IRMC, N° 71, pp. 10-28 (2002).*

[Drucker, 1999] H.Drucker. Donghui Wn, Vladimir Vapnik. **Support Vector Machines for spam categorization.** *IEEE Transactions on Neural networks. 10(5):1048-1054 (1999).*

[Dumais, 1998] S. Dumais, J. Platt, D. Heckerman , M. Sahami. **Inductive learning algorithms and representations for text categorization.** *Gardarin, G., French, J. C., Pissinou, N., Makki, K., et Bouganim, L., editors, Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management, PP. 148.155,Bethesda, US. ACM Press, New York, US(1998).*

[Drucker, 1996] H.Drucker, C.Cortes. **Boosting decision trees.** *Advances in Neural Information Processing Systems, page 479-485 (1996).*

[Domingos, 1996] P.Domingos, M.J. Pazzani. **Beyond independence : Conditions for the optimality of the simple bayesian classifier.** *In International Conference on Machine learning.page 105-112 (1996).*

[Dietterich, 1995] Dietterich, Eun Kong. **Machine learning bias, statistical bias, and statistical variance of decision tree algorithme.** *Technical report, department of computer science, Oregon state university (1995).*

[Dagan, 1995] I.Dagan, S.Engelson. **Commettee based sampling for training probabilistic classifiers.** *In International Conference on Machine Learning, page 150-157 (1995).*

[EL-Halees, 2007] A. M. El-Halees. **Arabic Text Classification Using Maximum Entropy.** The Islamic *University Journal (Series of Natural Studies andEngineering)Vol. 15, No.1, pp 157-167, ISSN 1726-6807(2007).*

[EL-Halees, 2006] A. M. El-Halees. ***Mining Arabic Association Rules for Text Classification.*** *In the proceedings of the first international conference on Mathematical Sciences. Al-Azhar University of Gaza, Palestine, 15 -17 (2006).*

[El-kassas, 2005] D.El- kassas. **Une etude contrastive de l'arabe et du francais dans un perspective de generation multilingue.** *UFR Linguistique, Université Paris 7 – Denis Diderot, France (2005).*

[El-kourdi, 2004] M.El-Kourdi, A.Bensaid , T.Rachidi. **Automatic Arabic Document categorization Based on the Naïve Bayes Algorithm.** *20th International Conference on Computational Linguistics.August, Geneva (2004).*

[Forsyth, 1999] R.S. Forsyth. **New directions in text categorization.** *In Gammerman, editor, Causal models and intelligent data management, PP 151.185. Springer Verlag, Heidelberg (1999).*

[Fuhr, 1991] N. Fuhr, C.Buckley. **A probabilistic learning approach for document indexing.** *In ACM Transactions on Information Systems, volume 9, PP. 223.248. (1991).*

[Gilli, 1998] Y. Gilli. **Texte et fréquence.** *Number 360. Université de Besançon, Paris (1988).*

[Hmeidi, 2008] I.Hmeidi, B.Hawashin, E.El-Qawasmeh, **Performance of KNN and SVM classifiers on full word Arabic articles.** *Science direct, Advanced Engineering Informatics 22 106–111 (2008).*

[Hammou, 2002] B.Hammo, H.Abu-Salem, S.Lytinen, M.Evens. **A Question Answering System to Support the Arabic Language.***Workshop on Computational Approaches to Semitic Languages. ACL,Philadelphia, PA, pp. 55-65(2002).*

[Han, 2001] E.Han, G.Karypis, V. Kumar. **Text categorization using weight adjusted k-nearest neighbour classification.** *Proceeding of the 5th Pacific-Asia Conference on knowledge discovery and Data Mining (PAKDD),2035:53-65 (2001).*

[Hayes, 1990] P.Hayes,S. Weinstein. **A system for content-based indexing of a database of news stories.** *In Rappaport, A and Smith, R. editor, Proceeding of IAAI-90, 2nd Conference on Innovative Application of Artificial Intelegence, pages 79-66. AAAI Press, Menlo Park, US (1990).*

[Iyer, 2000] R.Iyer, D. Lewis, R. Schapire,Y. A.Singer. **Boosting for document routin**. *In Agah, A., Callan, J., Rundensteiner, E., editors, gProceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management, PP 70.77, McLean, US. ACM Press, New York, US. (2000).*

[Jalam, 2003] R.Jalam. **Apprentissage automatique et categorisation de texts multilingue.** *Univeristé Lumière Lyon2.France (2003).*

[Jones, 2000] K. Jones, S.Walker, S.E. Robertson. **A probabilistic model of information retrieval.** *Development and comparative experiments–part2. Information processing and Management, 36(6) 809-840 (2000).*

[Joachim, 1998] T.Joachims .**Text categorization with support vector machines learning with many relevant features.** *In Nédellec, C. et Rouveirol, C., editors, Proceedings of ECML-98, 10th European Conference on Machine Learning, PP 137.142, Chemnitz, Springer Verlag, Heidelberg. Published in the "Lecture Notes in Computer Science" series, number 1398 (1998).*

[Khreisat, 2006] L.Khreisat, **Arabic text classi.cation using N-Gram frequency statistics. A comparative study.** *In Proceedings of the international conference on data mining (DMIN), Nevada, USA, pp. 78–82 (2006).*

[Khoja, 2001] S.Khoja. **APT: Arabic Part-of-speech Tagger.** ***In*** *Actes de l'atelier des étudiants de Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL2001), Carnegie Mellon University, Pittsburgh,Pennsylvania, http://zeus.cs.pacificu.edu/shereen/NAACL.pdf (2001).*

[Kim, 2000] Y.Kim S.Hahn, B.T. Zhang. **Text filtering by boosting Naïve Bayes classifiers.** *In ACM Press, editor, Proceedings of the 23rd International ACM SIGIR Conference on Research and Development in Information Retrieval, page 1968-175. (2000).*

[Khoja, 1999] S.Khoja, R.Garside . **Stemming Arabic text.** *Computing Department, Lancaster University, Lancaster (1999).*

[Lam, 1998] W.Lam. **Using a generalized instance set for automatic text categorization.** *In Proceedings of SIGIR-98, 21th ACM International Conference on Research and Developement In Information Retrieval, pages 81-89 (1998).*

[Kiraz, 1996] G.A.Kiraz. ***Analysis of the Arabic Broken Plural and Diminutive.*** *In Proceedings of the 5th International Conference and Exhibition on Multi-lingual Computing.Cambridge,UK (1996).*

[Larkey, 2006] S.Larkey, L.Ballesteros, M.Connell. **Light Stemming for Arabic Information Retrieval.** *Univ of Massachusets; Deot of computer science, USA (2006).*

[Larkey, 2005] S.Larkey, L.Ballesteros, M.Connell. **improving stemming for Arabic information retrievl.** *Univ of Massachusets, Deot of computer science, USA (2005).*

[Larkey, 2002] S.Larkey ,L.Ballesteros , M. Connell. **Improving Stemming for Arabic Information Retrieval: Light Stemming and Cooccurrence Analysis.** *In Proceedings of the 25th Annual International Conference on Research and Development in Information Retrieval (SIGIR 2002), Tampere, Finland, August, pp. 275-282 (2002).*

[Leopold, 2002] E.Leopold, J.Kindermann. **Text categorization with support vector machines. How to represent texts in input Space.** *Machine learning. 46:423-444 (2002).*

[Lefèvre, 2000] P.Lefèvre. **la recherché d'information du texte integrale au thesaurus.** *Hermès Science, paris (2000).*

[lewis, 1998] D.D.Lewis. **Naive Bayes at forty the independence assumption in information retrieval.** *In Nédellec, C. et Rouveirol, C., editors, Proceedings of ECML-98, 10th European Conference on Machine Learning, PP. 4.15, Chemnitz, DE. Springer Verlag, Heidelberg. Published in the "Lecture Notes in Computer Science" series, number 1398 (1998).*

[lewis, 1996] D.D. Lewis, R.E.Schapire, J.Pallan. **Training algorithms for linear text classifiers.** *In Proceeding of SIGIR-96,19th ACM International Conference on Research and Development in information Retrieval. Pages 298-306.Zurich.Suisse, ACM Press (1996).*

[Liddy, 1994] E.D.Liddy,W.Paik. **Texte categorization for multiple users based on semantic feature from a machine readable dictionary.***ACM Transaction on information systems, 12(3) pp.278-295(1994).*

[Littlestone,1994] N.Littlestone, M.Warmuthh. **The weighted majority algorithm.** *Information and computation,108(2):212-261 (1994).*

[Lewis, 1992] D.D.Lewis. **An evaluation of phrasal and clustered representations on a text categorization task.** *In Belkin, N. J., Ingwersen, P., et Pejtersen, A. M., editors, Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval, PP. 37.50, Kobenhavn, DK. ACM Press, New York, US (1992).*

[Porter, 1980] M.Porter. **an algorithm for suffix stripping.** *http//www.tartarus.org/ martin/ porterStemmer/def.txt (1980).*

[Plantié, 2006] M. Plantié. **Extraction automatique de connaissances pour la décision multicritère.** *Ecole Nationale Supérieure des Mines de Saint-Etienne (2006).*

[MESLEH, 2007] M.MESLEH. **Chi Square Feature Extraction Based Svms Arabic Language Text Categorization System.** *Journal of Computer Science 3 (6): 430-435(2007).*

[Maamouri, 2004] M *Maamouri*, A BIES, T Buckwalter **The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus.** In *Proceedings of the NEMLAR International Conference on Arabic Language Resources and Tools*, pages 102-109, Cairo, Egypt, September (2004).

[Morin, 2002] A.Morin. **Factorial corespondance analysis: a duel approach for semantics and indexing.** *In Proceedings of the Conference Compstat.(2002).*

[Manning, 1999] C.Manning, H.Schütze. **Foundations of Statistical Natural Language Processing.** *Cambridge,Massachusetts: The MIT Press. (1999).*

[Mladeni'c, 1998] D.Mladeni´c. **Feature subset selection in text learning.***In Nédellec, C. et Rouveirol, C., editors, Proceedings of ECML-98, 10th European Conference on Machine Learning, PP 95.100, Chemnitz, DE. Springer Verlag, Heidelberg. Published in the "Lecture Notes in Computer Science" series, number 1398 (1998).*

[McCallum,1998] A.McCALLUM, K.Nigam. **A comparaison of event models for Naive Bayes text classification.** *AAAI-98 Workshop on learning for Text Categorization.(1998).*

[Moulinier, 1996] I.Moulinier. **Une approche de la catégorisation de textes par l'apprentissage symbolique.** *PhD thesis, Université Paris 6, Paris (1996).*

[Masand, 1992] B.Masand, G. Linoff, D. Waltz. **Classifying news stories using memory based reasoning**. *In Proceeding of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 59-65 (1992).*

[Maron ,1961] M. Maron. **Automatic indexing: an experimental inquiry.** *Journal of the association for computing machinery, 8(3):404-417 (1961).*

[Ng, 1997] H.Ng, W.Goh, K.Low. **Feature selection, perceptron learning, and a usability case study for text categorization.** *In Belkin, N. J., Narasimhalu, A. D., et Willett, P., editors, Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information Retrieval, PP. 67.73, Philadelphia, US. ACM Press, New York, US (1997).*

[Quinlan, 1996] J. R. Quinlan. **Bagging, boosting, and C4.5.** *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 725-730 (1996).

[Sebastiani, 2002] F.Sebastiani. **Machine Learning in Automated Text Categorization.** *Consiglio Nazionale delle Ricerche, Italy (2002).*

[Sawaf, 2001] Sawaf, J. Zaplo, H. Ney. **Statistical Classification Methods for Arabic News Articles.** *Arabic Natural Language Processing, Workshop on the ACL. Toulouse, France.(2001).*

[Sable, 2000] C.Sable, V.Hatzivassiloglou. **Text based approaches for non topical image categorization.** *International journal of digital libraries,3(3): 261-275 (2000).*

[Sahami, 1999] M. Sahami. **Using Machine Learning to Improve Information access.** *PhD thesis, Computer Science Department, Stanford University (1999).*

[Schapire, 1998] R.Schapire, Y.Singer, A Singhal. **Boosting and Rocchio applied to text filtering.** *In Croft, W. B., Moffat, A., van Rijsbergen, C. J., Wilkinson, R., et Zobel, J., editors, Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval, PP. 215.223, Melbourne, AU. ACM Press, New York, US (1998).*

[Schapire, 1997] R.Schapire, Y.Freund, P. Bartlett,W. Lee. **Boosting the margin: a new explanation for the effectiveness of voting methods.** *In proceedings of the 14th International Conference on Machine Learning, page 322-330.Morgan Kaufmann. (1997).*

[Salzberg, 1997] S.Salzberg. **on comparing classifiers. Pitfalls to avoid and a recomanded approach.***Data mining and knowledge Discovry, 1(3): 317-328.(1997).*

[Schütze, 1995] H. Schütze, D.Hull, J.Pedersen. **A comparison of classifiers and document representations for the routing problem.** *In Fox, E. A., Ingwersen, P., et Fidel, R., editors, Proceedings of SIGIR-95, 18th ACM International Conference on Research and development in Information Retrieval, PP. 229.237, Seattle, US. ACM Press, New York, US (1995).*

[Schmid, 1994] H.Schmid. **probabilistic part of speech tagging using decision trees.** *International Conference on New Methods in Language Processing, Manchester,UK (1994).*

[Salton, 1975] G.Salton, A.Wong, C.Yong. **A vector space model for information retrieval.** *Communication of the ACM. 18(11):613-620 (1975).*

[Shannon, 1948] C.Shannon. ***A mathematical theory of communication.*** *Bell Systems Technical Journal(27), pp 379-423 & 623-656 (1948).*

[Tuerlinckx,2004] Laurence Tuerlinckx. **La lemmatisation de l'arabe non classique.** *JADT: 7es journées internationals d'analyse statistique des données textuelle, France (2004).*

[Tzeras, 1993]K.Tzeras, S.Hartmann. **Automatic indexing based on Bayesian inference networks.** *In R.Korfhage, E.Rasmussen and P.Willet editor, proceeding of SIGIR-93, 16th*

International Conference on Research and Development in Information Retieval, page 22-34. Pittsburgh. US. ACM Press , new York.US (1993).

[Vinot, 2004] R.Vinot. **Classification automatique de texts dans des categories non thématique.***These de doctorat. Ecole nationale superieur de telecommunication (2004).*

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

[Yang, 1997] Y.Yang. **An evaluation of statistical approaches to text ategorization.** *Information Retrieval, 1(1/2): PP. 69.90 (1997).*

[Yang, 1994] Y. Yang .**Expert Network .Effective and efficient learning from human decision in text categorization and retrieval.** *In proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 13-22 (1994).*

[Zaragoza, 1999] H. Zarogoza. **Modèles dynamiques d'apprentissage numérique pour l'accèsà l'information textuelle.** *Thèse de l'université Paris VI (1999).*

Résumé *: la catégorisation automatique de textes est un domaine de recherche en plein essor en raison de l'explosion de la quantité d'information disponible sous format électronique et la difficulté d'accéder à l'information pertinente parmi toutes celles qui sont accessibles. Son principal enjeu est de rendre une application informatique capable d'assigner d'une façon autonome une catégorie à un document en se basant sur son contenu. Pour décrire le contenu des documents, la quasi-totalité des systèmes actuels se base sur la représentation sac de mots en raison de sa simplicité. Néanmoins avec une telle représentation le sens des termes dans la majorité des cas reste ambigu, de plus la description de certains concepts nécessite l'utilisation de quelques mots pris simultanément, mais pas séparément ; dans ce cas, l'utilisation des mots simple (unigrames) pour décrire ces concepts va engendrer une ambigüité sémantique*

L'objectif de ce mémoire est de proposer une approche qui tente de réduire cette ambigüité et d'améliorer les performances des systèmes de catégorisation de textes arabes en se basant sur des descripteurs plus informatifs et plus discriminants que les mots. L'idée de base de cette approche consiste à bénéficier des avantages liés à utilisation des ngrammes et plus précisément les unigrames et les bigrammes ayant un apport informationnel élevé pour la représentation des documents ; et de tester leur influence sur les performances globales des systèmes de catégorisation de textes arabes.

Afin d'évaluer cette approche nous utilisons comme classifieur les machine à vecteur support (SVM) et comme base d'apprentissage un corpus textuel en langue arabe. Notons que le choix des SVM est dû essentiellement à leur robustesse ainsi leur capacité à traiter des espaces de données de grande dimensionnalité.