République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de
la Recherche Scientifique
Université Ibn Khaldoun – Tiaret –

Faculté des **S**ciences et de la **T**echnologie et **S**ciences de la **M**atière

Département des **S**ciences et de la **T**echnologie

MEMOIRE EN VUE DE L'OBTENTION DU DIPLÔME DE MAGISTER

SPECIALITE : Informatique

OPTION : **S**ystème d'**I**nformation et de la **C**onnaissance

Présenté par :

BENMESSAOUD Abdelkader

THÈME :

Méthodes à noyau pour l'apprentissage dans le cas des données non structurées

SOUTENU LE 23/06/2014, devant le jury composé de :

Mr Youcef DAHMANI	Maître de conférences	Université de Tiaret	Président
Mr Mohamed YAGOUBI	Professeur	Université de Laghouat	Examinateur
Mr Nasreddine LAGRAA	Maître de conférences	Université de Laghouat	Examinateur
Mme Hadda CHERROUNE	Maître de conférences	Université de Laghouat	Encadreur

Année Universitaire : 2013/2014

Dédicaces

À mes chers parents

que je ne pourrai jamais assez les remercier

À ma femme et mes enfants El-Amine et Essedik

À mes sœurs et mes frères

À toute ma famille et mes amis

À toute la communauté islamique

Je dédie ce modeste travail.

Remerciement

Je tiens tout d'abord à exprimer ma profonde reconnaissance et mes vifs remerciements à Mme Bouzouad Hadda, Maître de conférences à l'université de Laghouat, pour avoir accepté de diriger ce travail, pour sa disponibilité et ses précieuses orientations.

Mes remerciements s'adressent aussi à Mr. Ziadi Djelloul, Professeur d'université de Rouen, pour son aide et ses éclaircissements inestimables.

Mes remerciements s'adressent aussi à Mr. Dahmani Youcef, Maitre de conférences à l'ESI, d'avoir accepté de présider le jury. Pr. Yagoubi Mohamed Bachir, Professeur à l'université de Laghouat et Dr. Lagraa Nasreddine trouvent ici ma reconnaissance, pour leurs disponibilités malgré leurs occupations, pour avoir accepté d'évaluer ce travail.

Je n'oublierai pas de remercier très cordialement Mr. A. Nehar, Maitre assistant à l'université de Djelfa, pour son grand aide pour la rédaction et la correction de ce mémoire et pour les aspects pratiques et les outils aux quels il m'a initié et faciliter l'utilisation.

Je remercie également Y. Benallia, M. Benyagoub et O. Bachir pour l'investissement de leurs temps pour la lecture et la correction de ce mémoire.

Je remercie vivement tous les enseignants et le personnel de l'ESI, l'université de Tiaret qui ont participé à notre formation dans le cadre de l'école doctorale ou durant notre cursus universitaire.

Enfin, je tiens à remercier toute ma famille pour son support et son soutien, sans oublier mes amis de la promo 2010 (SIC & IRM) pour leurs sympathies.

ملخص

تعتبر (*SVM*) الطريقة الأكثر شيوعا فيما يعرف بالطرق المعتمدة على النواة و التي تشهد نجاحا كبيرا في السنوات الأخيرة. هذا النجاح مرده أساسا إلى استقلال هذه الطرق عن الأبعاد التمثيلية للبيانات التي تعتبر كبيرة جدا و أيضا في مرونة استخدام دوال النواة. هذه الخصائص تجعل من هذه الطرق خيارا جيدا لمختلف مهام التعلم الآلي ومنها بالخصوص مهام تصنيف النصوص.

تصنيف النصوص كان من التطبيقات الأولى لهذه الطريقة أين تفوقت من حيث النتائج على كل الطرق التقليدية للتعلم الآلي، مما جعلها مرجعا في هذا الميدان.

الميزة الرئيسية لطريقة *SVM* هو المرونة في اختيار النواة المناسبة لطبيعة البيانات واستخدامها على أنواع مختلفة من البيانات المعقدة (الرسوم البيانية، الأشجار،...)، في هذا السياق تم إدخال بعض دوال النواة الخاصة بالبيانات النصية (دوال السلاسل النصية) مسجلا بذلك تحسنا في الأداء العام لأنظمة التصنيف الآلي للنصوص.

في ما يخص تصنيف النصوص العربية فقد تم اعتماد منهجيات مختلفة في بعض الأعمال التي استندت معظمها على مجموعات نصية غير قياسية و متوسطة الحجم. أما بالنسبة لطريقة *SVM* فقد تم إعتمادها في القليل من الأعمال على أساس تمثيل خطي للبيانات متجاهلين بذلك الجوانب المتعلقة بترتيب الكلمات في النص و علاقاتها بعضها مع البعض. إضافة الى ذلك لم تستثمر أي دراسة حتى الآن دوال النواة المعرفة على البيانات النصية مكتفين في معظم هذه الدراسات بدوال النواة التقليدية والعامة.

في هذه الدراسة، اعتمدنا في تمثيل البيانات على المحولات (*transducers*). هذا التمثيل لا يسمح بالأخذ بعين الاعتبار الجوانب المتعلقة بترتيب الكلمات في النص و علاقاتها بعضها مع البعض فقط بل أيضا بتطبيق أو تنفيذ فعال لدوال النواة الخاصة بالبيانات النصية.

تم إجراء دراسة تجريبية لهذا النظام لتقييم النتائج أين أظهرت هذه الأخيرة تحسنا، مقارنة مع الطريقة التمثيلية الخطية الكلاسيكية، في *دقة التصنيف* (*Precision*) ولكن على حساب *الاستدعاء* (*Recall*).

يمكن القول كخلاصة أن سلسلة نصية معتدلة الطول مناسبة لأنظمة التصنيف الآلي للنصوص باللغة العربية وهذا بالنسبة للأنظمة أين معيار *الدقة في التصنيف* هو المعيار المهيمن.

الكلمات المفتاحية :

دوال النواة الخاصة بالسلاسل النصية ، التصنيف الآلي للنصوص، المحولات، لغة عربية.

Résumé

Les machines à vecteurs de support (*SVM*) représentent la méthode la plus répandue dans la littérature des méthodes connues sous le nom des méthodes à noyau qui connaissent un énorme succès depuis quelques années. Ce succès est dû principalement à l'indépendance de ces méthodes vis-à-vis de la haute dimensionnalité de l'espace d'attributs ou de représentation des données et à la flexibilité dans l'utilisation des fonctions noyaux.

La classification de textes a été parmi les premiers domaines d'application de cette méthode où leurs performances ont supplanté les algorithmes classiques. L'atout majeur des *SVM* est la flexibilité dans le choix des noyaux appropriées pour la nature des données et de leurs utilisation sur des données de type complexe (Graphes, Arbres,...).

Pour les données textuelles, différents noyaux ont été introduits (noyau P-spectre, toutes-sous-séquences, sous-séquence avec pénalité sur l'écart, ...) permettant une amélioration dans les performances globales d'un système de classification de textes.

Concernant la classification de textes en langue arabe, différentes approches ont été mises en œuvre dans quelques travaux. La plupart de ces travaux ont été menés sur des corpus non standards et de taille moyenne. Quant aux *SVM*, elles étaient adoptées dans quelques travaux sur la base d'une représentation vectorielles des données négligeant ainsi les aspects ordre et cooccurrence des termes. Encore, mise à part les noyaux usuels, aucune étude n'a encore exploité d'autres noyaux, notamment ceux adaptés aux données textuelles.

Dans ce mémoire, nous avons opté pour une représentation des documents sous forme de transducteur. Une telle représentation permet, non seulement la prise en charge des aspects liés à l'ordre et la cooccurrence des éléments d'un texte mais aussi une implémentation efficace des noyaux de séquences. Dans ce contexte, nous avons réalisé un système de classification de textes combinant les *SVM* avec ces noyaux.

Une étude empirique de ce système a été menée pour l'évaluation de ce dernier. Les résultats rapportés montrent une amélioration par rapport à l'approche *sac de mots* de la *Précision* de classification mais au détriment du *Rappel*. Une séquence de termes d'une longueur modérée est convenable pour les systèmes de classification de textes en langue arabe visant à augmenter la *Précision*.

Mots clés : *SVM*, noyaux de séquences, classification de textes, langue arabe, transducteurs.

Abstract

Support Vector Machine (*SVM*) are the most popular algorithm of a class called kernel methods. This class of methods knew a huge success in the recent few years. Their success is mainly due to the independence of these methods against the high dimensionality of the feature space or data representation and the flexibility in the use of kernel functions.

The text classification was among the first area of applications of this method where their performances have supplanted traditional algorithms. The major advantage of *SVM* is the flexibility in choosing the appropriate kernel for the nature of this data and their use on complex data types (graphs, trees, ...).

For textual data, different kernels were introduced (*P-spectrum*, *All-SubSequences*, *Gap-Weighted Subsequence kernel*, ...) to improve the overall performance for text classification.

Relating to the Arabic texts classification, different approaches have been implemented. Most of these studies have been conducted on non-standard corpus of medium size. For *SVM*, they were adopted in some work where the representation of the data was in vector space model and neglecting order and co-occurrence of terms. In other side, except usual kernels, no study has yet exploited other kernels, especially those adapted to textual data.

In this work, we opted for transducers to represent documents. This representation doesn't allow only support aspects of order and co-occurrence of elements in a text but also an efficient implementation of kernel sequences. In this context, we carried out a system for text classification combining *SVM* with these kernels.

An empirical study was conducted to evaluate the classification system. The reported results show an improvement of *Precision* in classification but to the detriment of the *Recall*. To increase *Precision*, a sequence of terms of moderate length is suitable for Arabic texts classification.

Key-words : *SVM*, Subsequences kernels, Text classification, Arabic, Transducers.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Contexte

Le développement technologique qui s'est accéléré ces deux dernières décennies a entraîné un accroissement massif de tout type de données (satellitaires, génomiques, financiers, documentaires ...) qui sont à présent échangées et stockées en masse. Pour les entreprises, un tel volume, en perpétuelle croissance, constitue une importante mine de données à exploiter dans le but d'acquérir une meilleure connaissance. Or, ce volume a atteint une complexité difficilement maîtrisable par un être humain. En conséquence, une demande croissante s'est manifestée pour des solutions et des outils efficaces pour conserver, traiter, analyser, chercher et classer ces données, afin d'assister les entreprises et les utilisateurs submergés par cette masse d'informations.

Au fil des années, ces traitements se sont diversifiés et se sont variés (numérisation, archivage, filtrage, indexation, prévision, aide à la décision ...) nécessitant ainsi une nouvelle approche quant à leurs mise en œuvre remplaçant l'ancienne méthode basée sur le codage direct des règles de décision par un expert. Ces règles sont difficiles à programmer à l'avance, même si cela était possible, elles devraient être régulièrement modifiées par l'expert pour qu'elles reflètent la réalité actuelle et c'est ainsi qu'une nouvelle approche s'est imposée comme un axe majeur pour ce type d'application [Mit97]. Cette approche est l'apprentissage automatique.

L'apprentissage automatique (*Machine Learning*) permet de construire, dans un processus inductif, des modèles de prédiction à partir d'un échantillon de données préalablement traité dans le cas de l'apprentissage supervisé ou de découvrir des structures en ne nécessitant aucune information préalable dans le cas de l'apprentissage non supervisé [CM02]. Dans le cadre de l'apprentissage supervisé, le but est de produire une fonction de décision à partir d'un algorithme d'apprentissage (apprenant) et des données (exemples d'apprentissage) de façon à ce que cette fonction réalise un minimum d'erreurs sur les données disponibles (risque empirique faible) et que les prédictions réalisées sur des nouvelles données soient le plus éloignées d'une prédiction aléatoire (risque réel faible).

La classification de textes (T.C) constitue un champ très actif s'inscrivant dans la problématique d'apprentissage supervisé [Seb02]. Dans un système de classification de textes, les documents textes doivent subir, en premier lieu des prétraitements consistant à éliminer les signes de ponctuations et les mots peu informatifs (mots outils). Les termes restants sont généralement soumis à une normalisation linguistique. Cette normalisation consiste à regrouper

les variantes d'un même mot dans un seul groupe selon une approche légère (light stemming) ou une autre plus efficace mais plus lourde à mettre en œuvre (lemmatisation ou extraction de la racine) [Gué06]. Le recours à la réduction de dimension qui peut se baser sur des métriques statistiques ou issues de la théorie d'information s'avère nécessaire pour faire face à la haute dimensionnalité qui caractérise la T.C. Comme résultat de cette phase un document sera représenté par un vecteur de termes pondérés. La pondération des termes consiste à calculer un poids pour chaque terme pour valoriser sa contribution à la sémantique discriminante d'un document.

Enfin, un classifieur est construit par l'apprentissage des caractéristiques de chaque catégorie à partir d'un ensemble de documents d'apprentissage et testé sur un ensemble de tests en comparant les décisions prises par ce classifieur et ceux codés dans le corpus afin de mesurer l'efficacité de ce dernier. Différents types d'algorithmes d'apprentissage supervisés (K-ppv, Naïve Bayes, arbres de décision, réseaux de neurones, ...) ont été intensivement utilisés pour la construction des classifieurs.

Les *SVM* introduites par Boser [BGV92], sont une classe d'algorithmes qui combinent des techniques d'optimisation statistique avec les fonctions noyaux. Dans leur version la plus simple, elles construisent un hyperplan séparateur entre deux ensembles de points, tout en essayant de maximiser la distance (marge) entre les points les plus proches de ces deux ensembles. Le problème d'optimisation est formalisé uniquement à partir des produits scalaires entre objets. Cette formalisation est derrière l'innovation majeur des *SVM* qui est l'utilisation des noyaux. Ces derniers permettent de plonger les objets dans un espace de dimension plus grande favorisant la possibilité de trouver un bon séparateur.

Les noyaux sont des fonctions de similarité respectant certaines propriétés mathématiques. Ils peuvent être utilisés avec des algorithmes d'apprentissage linéaires tels que les *SVM* pour extraire des relations non-linéaires. Le succès des noyaux pour l'apprentissage automatique s'explique aussi par la possibilité de définir des noyaux sur des données non-numériques complexes telles que les graphes et les arbres.

En effet, dans plusieurs cas (bio-informatique, images, graphes et documents textes) les données ne peuvent pas être décrites par le modèle de l'espace vectoriel (*VSM* : Vector Space Model) i.e par un vecteur d'attributs sans perte d'informations importantes durant le processus de transformation qui peut être lui aussi coûteux et très complexe [LSST+02].

Passer à une représentation des données radicalement différente du *VSM* est devenu désormais possible avec les noyaux. Différents types de noyaux ont été proposés ces dernières années pour le traitement des séquences et des données structurées (arbres, graphes, ...) [Gär03].

Pour le traitement des séquences, les noyaux proposés reposent le plus souvent sur le comptage des sous-séquences communes aux objets comparés. Ce comptage peut tenir compte de l'ordre et de l'écart entre les sous-séquences de ces objets permettant ainsi de prendre en considération les aspects sémantiques qui caractérisent les données de certains domaines (documents textes, bio-séquences ...). Pour ces domaines, différents noyaux ont été introduits pour améliorer les performances générales de la classification (noyau P-spectre, toutes-sous-séquences, sous-séquence de longueur fixe, sous-séquence avec pénalité sur l'écart) [LEC+04] [LSST+02].

Les résultats expérimentaux de ces différents noyaux montrent une amélioration dans les performances globale des systèmes basés sur les *SVM* combinés avec ces différents noyaux notamment quand les bons paramètres sont sélectionnés [LSST+02].

Problématique

Dans ce mémoire nous nous intéressons à la classification de textes en langue arabe. Concernant ce domaine, certaines langues posent des difficultés supplémentaires. Pour le cas de la langue arabe, la phase de prétraitement doit faire face à la représentation morphologique complexe et à la richesse flexionnelle de cette langue [SA08]. Les défis de langue arabe peuvent se résumer comme suit :

- Variation orthographique où certaines caractères changent de formes selon leurs positions dans un mots.
- Phénomène d'agglutination où des prépositions, des pronoms, ... se collent au début ou à la fin d'un mot.
- Caractère d'infliction et dérivation où des noms, des adjectifs et des verbes peuvent être dérivés en nombre important d'une seule racine augmentant ainsi l'ambiguïté des mots.

Ces aspects et bien d'autres, rendent difficile et imprécise, la tâche de la segmentation et de la lemmatisation.

En plus de ces difficultés liées à la morphologie, on peut citer aussi le manque d'outils et des ressources consacrés à la langue arabe.

Concernant les études réalisées pour la classification des textes en langue arabe, on constate un faible intérêt pour ce domaine traduit par le peu de travaux inscrits dans ce cadre. De plus, la plupart de ces travaux ont été réalisés sur des corpus non standard et de taille modérée limitant ainsi l'intérêt des résultats rapportés et parfois créant des confusions concernant les conclusions tirées (par exemple l'effet de stemming sur les performances générales d'un système de classification).

Quant à l'utilisation des *SVM* comme méthode de classification, on peut signaler d'abord le nombre limité des systèmes de classification implémentés sur la base de cette méthode et notamment l'absence totale de l'exploration d'une variété de noyaux (mise à part les noyaux usuelles).

Objectif

L'objectif de ce mémoire est d'investir dans les deux facettes des *SVM* pour réaliser un système de classification de textes avec une approche différente par rapport aux travaux réalisés : D'une part, dans l'élégance de la démarche mise en œuvre par les *SVM* permettant de faire face à la grande dimensionnalité de l'espace de la représentation des textes tout en garantissant de

bonnes performances de généralisation ; d'autre part, dans la puissance des noyaux qui peuvent être choisis selon la nature des données et utilisés sur des données complexes.

En effet, la plupart des algorithmes d'apprentissage supervisé sont conçus pour travailler avec une représentation vectorielles des données (*VSM*). Dans ce type de codage les entités textuelles sont transformées sous forme d'une représentation de vecteurs. L'avantage d'une telle représentation est la simplicité, car elle n'utilise que les fréquences d'apparition des mots dans les documents. Cependant, ce modèle néglige tout aspect lié à l'ordre et la cooccurrence de ces éléments. Par contre, les *SVM* peuvent être utilisés avec n'importe quelle représentation adoptées pour les textes.

A fin de répondre à cet objectif, nous proposons un classifieur à base de noyau en utilisant une représentation des textes sous forme de transducteur. Une telle représentation permet de prendre en compte les aspects ordre et cooccurence incorporés dans les documents textes. Nous avons aussi exploité quelques noyaux appropriés pour ce type de données.

Structure du document

Le présent document est décomposé en 2 parties :

Première partie

Le premier chapitre est consacré à l'apprentissage automatique et la classification de textes. Dans la première partie nous nous somme focalisés sur l'apprentissage supervisé où nous avons rappelés brièvement quelques principes inductifs utilisés par les méthodes d'apprentissage supervisé et les différents risques (empirique, réel et structurel) sur lesquels se base ces différentes méthodes pour concevoir leurs solutions. Le deuxième partie de ce chapitre est consacré au domaine de la classification de textes. Ce chapitre vise à présenter les différentes étapes nécessaires pour construire un système de classification de textes, son évaluation ainsi que les principaux algorithmes d'apprentissages utilisés dans ce domaine. Nous avons aussi, consacré une partie aux différents modes de représentation des documents textes.

L'état d'art sur les différents travaux réalisés pour la classification de textes en langue arabe est présenté dans le deuxième chapitre. D'abord nous avons présenté la langue arabe et ses défis morphologiques. Ensuite, nous avons détaillé les différentes normalisations linguistique employées et les différentes travaux réalisés pour le développement d'un système de classification de textes en langue arabe.

La méthode des séparateurs à vaste marge (*SVM*) est exposée dans le troisième chapitre. La première partie est consacré à la représentation de la démarche mise en œuvre par les *SVM* pour réaliser une discrimination binaire entre deux classes ainsi que le passage à une représentation des données dans une dimension plus élevé grâce aux noyaux. Les concepts des fonctions noyaux ont fait l'objet de la deuxième partie de ce chapitre.

Deuxième partie

Cette partie est consacrée à la conception de notre système de classification et l'étude empirique de celui-là.

Dans le chapitre quatre, nous avons mené en premier lieu une étude de quelques noyaux définis pour les données textuelles dans le but de choisir les noyaux appropriés pour la classification de textes en langue arabe. Une partie de ce chapitre est consacré aux concepts de transducteurs et noyaux rationnels qui permettent une implémentation efficace de différents noyaux pour données textuelles.

Ensuite, l'étude empirique est présentée dans le dernier chapitre où nous avons exposé les étapes entreprises pour la réalisation de notre système de classification ainsi que les différents résultats obtenus par les différentes approches adoptées.

Première partie

Classification de textes et SVM

Chapitre 1

L'apprentissage et classification de textes

1.1 L'apprentissage

L'apprentissage est une vocation essentielle de l'être humain. Il consiste à acquérir un nouveau comportement à la suite d'un entraînement (habituation, conditionnement ...) à travers un processus d'acquisition de pratiques, de connaissances, de compétences et d'attitudes par l'observation, l'imitation, l'essai et la répétition. Le dictionnaire Larousse [1] définit l'apprentissage comme étant l'"*Ensemble des processus de mémorisation mis en œuvre par l'animal ou l'homme pour élaborer ou modifier les schèmes comportementaux spécifiques sous l'influence de son environnement et de son expérience*".

Concernant le domaine scientifique relativement jeune de l'informatique, l'objectif de l'apprentissage est d'imiter, à l'aide d'algorithmes exécutés par des ordinateurs, la capacité qu'ont les êtres vivants à apprendre par l'exemple [DMS+11]. Dans cette philosophie l'apprentissage concerne toute tentative de développement, d'analyse et d'implémentation de méthodes automatisables qui permettent à une machine d'évoluer grâce à un processus d'apprentissage. Par conséquent ces méthodes ne sont pas élaborées par des moyens algorithmiques classiques c.à.d. à partir de règles écrites à la main où l'on code ligne après ligne la séquence d'instructions que l'ordinateur aurait à suivre mais plutôt la démarche consiste à développer des techniques pour que les programmes puissent s'entraîner à partir d'exemples.

D'une façon plus précise, cette discipline englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle [SG02]. Elle invoque deux disciplines : celle issue de l'intelligence artificielle, qualifiée de *symbolique* et celle issue des statistiques, qualifiée de *numérique*. Ce mémoire s'inscrit dans la deuxième approche.

L'apprentissage automatique a pris un rôle de plus en plus essentiel dans le domaine scientifique. Aujourd'hui les nombreux domaines d'application témoignent du succès pour ces mé-

1. La version électronique disponible sur l'adresse : http ://www.larousse.fr consultée le 23 août 2012.

thodes : fouille de données, reconnaissance de la parole, reconnaissance de caractères manuscrits, bio-informatique, traitement automatique du texte et bien d'autres.

Dès qu'un phénomène, qu'il soit physique, biologique ou autre, est trop complexe ou trop bruité pour accéder à une description analytique débouchant sur une modélisation déterministe, un ensemble d'approches doit être élaboré afin d'en décrire au mieux le comportement à partir d'une série d'observations. La spécificité des algorithmes d'apprentissage numérique réside dans le fait qu'ils sont capables d'extraire automatiquement des relations pertinentes entre les données et de les utiliser sur de nouvelles données. Ce processus est inductif, il tente de généraliser les relations extraites d'un échantillon de données (données d'apprentissage) à tout l'espace de données de la source ayant généré ces mêmes données.

L'apprentissage permet de résoudre deux problèmes principaux :

1. **La prédiction** qui consiste à générer une valeur pour une variable d'un individu, appelée "variable cible", en fonction des autres variables du même individu. Un individu est un point multidimensionnel où chaque dimension est décrite par une variable. Pour ce type de problème, un modèle est construit lors de la phase d'apprentissage, en extrayant la relation qui définit la variable cible en fonction des autres variables. La prédiction est ensuite effectuée en appliquant le modèle aux données.

2. **Le regroupement** qui permet de classer les données en catégories. Il existe deux types d'approches pour ce problème. La première, supervisée, qui est le classement (en anglais *categorization* ou *classification*). Elle repose sur l'utilisation d'un échantillon de données classées en catégories afin de construire un modèle décrivant les relations entre un individu et une classe. La seconde approche, non supervisée, est la classification (clustering). Les données pourront, alors, être regroupées en groupe d'individus selon leurs similarités.

1.1.1 Différents types d'apprentissage

Les algorithmes d'apprentissage peuvent se catégoriser selon le mode d'apprentissage qu'ils emploient :

- **Apprentissage supervisé** : il vise le type de problème où il y a la présence d'une variable (Y) à expliquer ou d'une forme à reconnaître qui a été, observée sur les mêmes objets, conjointement avec X. Autrement l'apprentissage supervisé tente de trouver une fonction ϕ susceptible, au mieux selon un critère à définir, de reproduire Y ayant observé X : $Y = \phi(X) + \epsilon$, où ϵ symbolise le bruit ou l'erreur de mesure.
  Pour la tache de classement, Y est une variable discrète finie indiquant l'étiquette associée à X. Dans le cas d'un classement binaire (problème à deux classes) Y prend généralement une valeur dans $\{-1, +1\}$. Pour la tache de prédiction Y est une variable réelle.
- **Apprentissage non supervisé** : Contrairement au premier type d'apprentissage, celui là se caractérise par l'absence d'une variable à expliquer. Il a comme objectif la recherche

d'une typologie ou taxinomie de l'ensemble des observations i.e. le regroupement de celles-ci en classes homogènes tout en étant les plus dissemblables entre elles (clustering).

Il existe un autre type d'apprentissage situé entre l'apprentissage supervisé et l'apprentissage non supervisé. Les tâches réalisées dans ce type d'apprentissage sont les mêmes que celles réalisées en apprentissage supervisé, à la différence qu'elles font usage des données non étiquetées. Il est employé pour améliorer les performances en combinant les données étiquetées et non étiquetées comme par exemple en classification semi-supervisée.

1.1.2 Apprentissage supervisé

On parle d'apprentissage supervisé si les classes sont prédéterminées et les exemples connus, le système apprend à classer selon un modèle de classement. Le choix de ce modèle dépend du principe inductif adopté que nous essayerons d'expliquer dans la suite de cette section.

1.1.2.1 Modèle général de l'apprentissage supervisé

L'apprentissage supervisé est constitué de trois composantes (Figure 1.1) [CM02] :

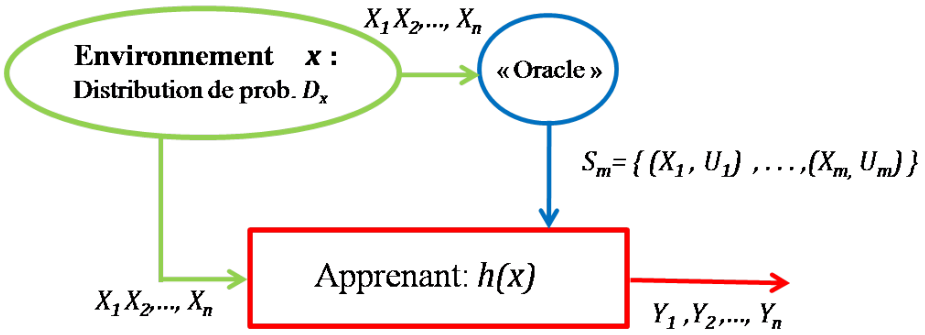

FIGURE 1.1 – Composantes d'un système d'apprentissage supervisé.

- **Environnement** : il engendre un échantillon de données ou des formes x_i tirées indépendamment et de manière identiquement distribuées suivant une distribution D_X.
- **Oracle** : (superviseur, professeur ou nature) qui retourne une étiquette ou réponse désirée u_i pour chaque forme x_i suivant une distribution de probabilité $F(u/x)$ inconnue.
- **Apprenant** : capable de réaliser une fonction h (fonction hypothèse) prise dans un espace de fonction H (l'espace des hypothèses) telle que la sortie produite par l'apprenant vérifie : $y_i = h(x_i)$ pour $h \in H$.

L'apprenant reçoit d'une part un échantillon de données $\{x_1, x_2, \ldots, x_n\}$; où x_i appartient à X l'espace de représentation ou de description des entrées attachées à chacun de ces formes x_i. Il reçoit d'autre part une étiquette (ou supervision) u_i suivant une fonction f ; fonction cible appartenant à une famille de fonctions F (l'espace des fonctions cibles).
L'apprenant cherche une fonction hypothèse h (h^* optimale) suivant un principe inductif (voir

sec. 1.1.2.3) dans l'espace des fonctions H aussi proche que possible de f : $(h(x_i) = y_i \rightsquigarrow f(x_i) = u_i)$.

1.1.2.2 Risque réel et risque empirique

Dans la tâche d'apprentissage l'apprenant cherche à minimiser la distance (proximité) entre f et la fonction h recherchée. Pour évaluer cette distance une mesure à été introduite. Il s'agit d'une fonction perte ou coût l qui pour chaque entrée x_i et réponse désirée u_i, elle mesure $l(u_i, h(x_i))$; le coût d'avoir pris la réponse ou la décision $y_i = h(x_i)$ à la place de u_i. La fonction de perte la plus couramment utilisée [DMS+11], notamment pour les problèmes de la classification, est la fonction de perte *zéro-un* l_z qui est définie par :

$$l_z = \begin{cases} 0 & \text{lorsque } y_0 = y \\ 1 & \text{lorsque } y_0 \neq y \end{cases} \tag{1.1}$$

Où $y_0 = h(x)$ est la classe attribuée par le classifieur tandis que y est la classe réelle. L'espérance de cette fonction définie le *risque réel*. Pour une fonction de perte l donnée le risque réel est :

$$R_{reel}(h) = \int_{z=(x,y)\in Z} l(h(x_i), u_i) dp Dz(x, u)$$

L'objectif d'un apprenant idéal A^* ; pour une fonction de coût l et une distribution Dz fixée, est alors de choisir l'hypothèse h^* qui minimise le risque réel :

$$h^* = arg\ min_{h\in H} R_{reel}(h)$$

h^* est la fonction qui minimise le risque réel relativement à la fonction de coût choisie (qui approxime f au mieux). $R_{reel}(h^*)$ représente le risque d'erreur minimale liée à l'hypothèse h^* relativement à l'espace d'hypothèses H utilisé. Ce risque dépend donc fortement de l'espace d'hypothèses H utilisé. Le risque réel ne peut être qu'estimé ; conséquence de l'ignorance de la distribution Dz. Une autre mesure appelée *risque empirique* permet l'estimation du risque lié à une hypothèse h à partir d'un ensemble S_m de m exemples. Il est égal à :

$$R_{emp}(h) = \frac{1}{m} \sum_{i=1}^{m} l(h(x_i), y_i) \text{ pour } (x_i, y_i) \in S_m$$

L'hypothèse qui minimise le risque empirique est alors :

$$h^*_{emp} = arg\ min_{h\in H} R_{emp}(h)$$

h^*_{emp} est généralement différente de h^*. En effet pour un espace d'hypothèses quelconque,

il n'y a pas forcément convergence entre la sélection de l'hypothèse qui minimise le risque empirique et celle qui minimise le risque réel [CM02]. Par contre lorsque le nombre d'exemples (S_m) tend vers l'infini il y a convergence entre le risque empirique et le risque réel pour une hypothèse h donnée :

$$\lim_{m\to\infty} R_{emp}(h) = R_{reel}(h)$$

La raison de l'être c'est que lorsque m tend vers l'infini, S_m sert à évaluer l'ensemble des risques liés à toutes les hypothèses de H sur les quelles l'hypothèse optimale est sélectionnée.

1.1.2.3 Principe inductif

La capacité de généralisation est un souci majeur dans l'apprentissage supervisé ; en effet étant donné un nombre fini d'exemples, beaucoup de solutions sont possibles ; elles donnent toutes les bonnes réponses sur les exemples d'apprentissage mais des réponses différentes sur de nouveaux cas.
Le pouvoir de généralisation d'un algorithme d'apprentissage est dépendant du *principe inductif* qu'il réalise et de l'espace des hypothèses qui correspond à l'ensemble des fonctions de décision réalisables.

Le *principe inductif* dicte ce que doit vérifier la meilleure hypothèse en fonction de l'échantillon d'apprentissage, de la fonction de perte et éventuellement d'autres critères. Il s'agit d'un objectif idéal. Il consiste à préciser comment vérifier qu'une hypothèse $h \in H$ est la meilleure relativement à un ensemble d'exemples S_m. Généralement, cela correspond à définir une relation d'ordre entre les hypothèses.
Cependant, il faut distinguer la méthode ou algorithme d'apprentissage du principe inductif. L'algorithme d'apprentissage décrit une réalisation effective du principe inductif qui ne précise pas les détails algorithmiques pour produire l'hypothèse h^*, ni les problèmes calculatoires qui peuvent en découler. Ainsi il peut donc exister différents algorithmes qui réalisent le même principe inductif.

Trois grandes familles peuvent être mises en évidence lors d'une catégorisation des différents principes inductifs [CM02] :

- **Minimisation du Risque Empirique (MRE) :**
  c'est probablement le principe inductif le plus utilisé en apprentissage automatique. L'idée de base c'est que si une hypothèse h s'accorde aux données fournies, alors elle s'accordera aux données futures. Dans ce principe l'apprenant cherche à sélectionner l'hypothèse qui minimise le risque empirique puisque selon ce principe on suppose que plus le risque empirique est faible plus le pouvoir généralisateur de l'hypothèse est grand. Les réseaux de neurones, par exemple, utilise implicitement un espace d'hypothèses vaste qui peut approximer n'importe quelle fonction.
- **Choix de l'hypothèse la plus probable :**

Ce principe consiste à sélectionner l'hypothèse la plus probable, i.e. l'hypothèse la plus vraisemblable étant donné S.

C'est le principe de décision bayésien. Si on définit à priori une distribution de probabilité sur l'espace des hypothèses H, l'examen de l'échantillon d'apprentissage S permet de modifier cette distribution par les règles de révision des probabilités de Bayes, ce qui permet par la suite, de choisir l'hypothèse la plus probable a posteriori. Ce principe est utilisé dans les k-plus proches voisins [CH67].

– **Compression d'information :**

L'idée de la sélection de l'hypothèse qui comprime au mieux l'information contenues dans S, est de produire une hypothèse à la fois simple et proche des données. Cela peut être réalisé en éliminant les redondances présentes dans les données S afin d'extraire les régularités sous-jacentes.

La difficulté est de pouvoir exprimer la complexité d'une hypothèse, relativement à une autre. La limitation du nombre de neurones cachés dans les réseaux de neurones ou l'élagage des arbres de décision sont des exemples de ce principe. Dans la plus part des cas, l'hypothèse optimale correspond à un compromis entre adéquation avec les données et complexité réduite.

1.1.3 Théorie d'apprentissage de Vapnik-Chervenonkis

Vapnik et Chervonenkis [Vap95] ont montré que le risque empirique seul n'était pas un bon estimateur du risque théorique : par exemple pour une petite taille d'exemple d'apprentissage une large déviation est possible et le risque du sur-apprentissage est très probable.

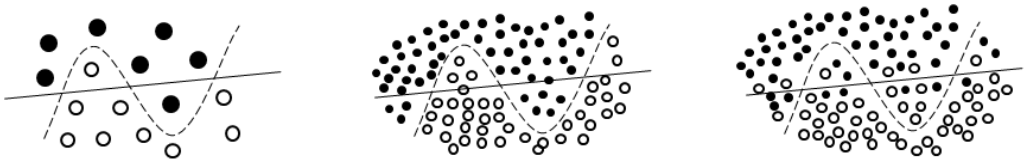

FIGURE 1.2 – Phénomènes du sous-apprentissage et sur-apprentissage [CST10].

La figure 1.2 illustre les phénomènes de sous-apprentissage et sur-apprentissage. Si la taille de l'ensemble d'apprentissage est modérée (schéma de gauche), les séparateurs S1 (ligne continu) et S2 (ligne discontinu) peuvent être juste et S2 est plus complexe mais présente moins d'erreurs. Si la taille de données devient plus importante, la distinction entre séparateurs est nettement meilleurs. En effet si le séparateur S2 est juste alors S1 est en sous-apprentissage (l'image du milieu) et si S1 est juste alors S2 va être en sur-apprentissage (l'image de droite).

Selon la théorie de Vapnik-Chervenonkis, pour éviter le sur-apprentissage il faut restreindre la complexité de la classe F où la fonction hypothèse est choisie. Si une fonction est simple (par exemple linéaire) et explique la plupart des données ; elle est préférable à une autre fonction complexe [MMR+01].

La complexité ou le pouvoir séparateur d'une classe de fonction F est quantifié par la dimension de Vapnik-Chervenonkis notée $VC - dimension$. Ce pouvoir séparateur est donné par le nombre maximum de points pouvant être séparés par une fonction de cette famille quel que soit l'étiquetage des points. Par exemple, dans un espace à deux dimensions, un séparateur linéaire peut, au maximum, séparer 3 points pour toutes combinaisons d'étiquetage de ces points (Figure 1.3). Par contre, on est incapable de le faire pour 4 points. Plus généralement,

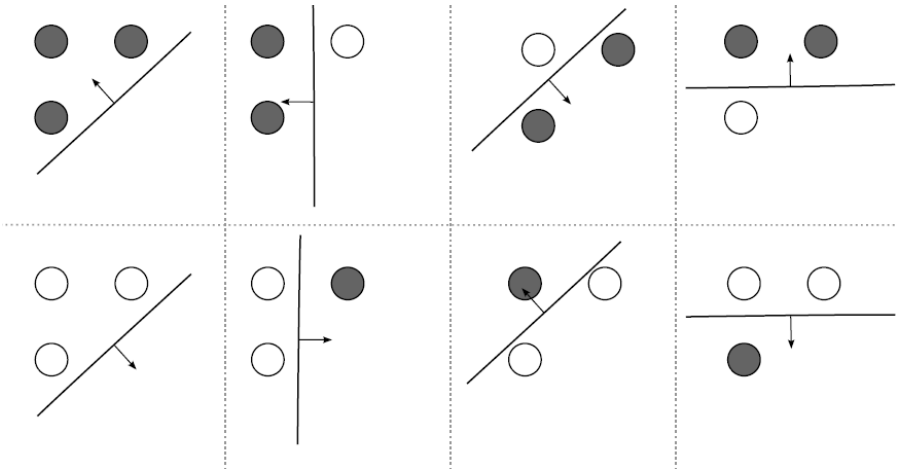

FIGURE 1.3 – Nb max. de points pouvant être séparés par un hyperplan.

la $VC - dimension$ d'un hyperplan séparateur de points dans un espace à n dimension est $n + 1$.

En utilisant La VC-dimension, le risque réel peut être approché avec une meilleure borne. Dans le cas du classement, nous avons avec une probabilité $1 - \eta$, la borne suivante [Vap95] :

$$R_{reel} \leq R_{emp} + \sqrt{\frac{h(log(2M/h) + 1) - log(\eta/4)}{M}}$$

Le membre droit de cette inégalité appelé le *risque garanti* est composé de deux terme : le risque empirique et une quantité qui dépend du rapport M/h appelé *intervalle de confiance* puisqu'il représente la différence entre le risque empirique R_{emp} et le risque réel R_{reel}.
Si le rapport M/h est suffisamment grand le principe MRE suffit pour garantir une faible valeur du risque réel. Par contre si le rapport M/h est petit le *risque garanti* prend une valeur importante et le MRE ne suffit pas pour garantir un R_{reel} minimal.

Pour remédier à ce problème le principe de *minimisation du risque structurel* (MRS) est introduit par Vapnik. Ce principe consiste à créer une structure dans l'ensemble d'hypothèses avec une VC-dimension croissante ($F_1 \subset F_2 \cdots \subset F_n$) et de choisir dans un premier temps le sous-ensemble F_i de fonctions avec la VC-dimension h_i minimisant la somme du risque empirique et l'*intervalle de confiance*. Puis dans un deuxième temps, la fonction $f_\alpha \in F_i$ minimisant le risque empirique sera choisie pour être la fonction optimale.

1.2 Classification de textes

Avec l'avènement d'Internet et la vulgarisation des moyens permettant la productivité et le partage des documents sous forme digitale, une grande quantité de données textuelles (articles, rapports, études, e-mails, messages de forums, enquêtes clients, fiches de centres d'appel, descriptifs de produits ...) est mise à la disponibilité des entreprise. Exploiter efficacement cette grande masse et extraire les informations pertinentes à partir de ces données à un coût raisonnable est devenu un enjeu important nécessitant ainsi des systèmes de traitement de l'information de plus en plus sophistiqués.

Pour répondre à ce problème plusieurs axes de recherches ont été explorés. L'approche la plus ancienne a été sans nul doute issue de l'ingénierie des connaissances. Dans les années 90, cette approche a cédé la place aux approches par apprentissage numérique [Mit97]. Dans cette approche, la classification ou la catégorisation de textes a pris de l'ampleur et au fil des années, les applications invoquant cette discipline se sont multipliées et diversifiées. L'intérêt grandissant pour cette discipline est aussi motivé par le fait qu'un bon nombre de problèmes liés aux données textuelles peuvent soit être ramenés à des problèmes de classification soit faire appel à des tâches de classification [Suj07] visant à faciliter l'accès à l'information (détection et classification d'opinions, résumé automatique ...).

Avant de présenter la classification de textes, nous allons donner un aperçu de la classification dans sa généralité.

Le terme de classification est associé à la notion d'abstraction ; une classification permet de synthétiser des informations dans des groupes plus généraux. C'est une forme d'abstraction, puisque l'on va mettre de côté des descriptions plus exactes des objets et ne faire ressortir que les traits particuliers que certains d'entre eux ont en communs.

La classification est définie comme étant " *l'action de classifier et le résultat de cette action* " [2] ou " *la distribution en classes et suivant un certain ordre* " [3].

Bien qu'il existe depuis longtemps, le terme " *classification* " est un terme ambigu en français puisqu'il ne distingue pas entre l'action de créer des classes : " *classer* " (diviser et repartir en classes) et l'action d'affecter un élément dans une classe : " *classifier* " (repartir selon une classification) [Nak07]. Encore les termes " classification " et " catégorisation " sont souvent indifféremment utilisés dans la littérature scientifique [Réh11].

Dans ce mémoire, nous utilisons indifféremment, le terme classification ou catégorisation pour désigner l'action d'affectation d'un ensemble d'individus dans des classes prédéfinies.

2. Dictionnaire de l'Académie française, 8^e édition ,1992
3. Dictionnaire de l'académie Française 5^e, 1798 à l'adresse http ://portail.atilf.fr/dictionnaires/ACADEMIE/CINQUIEME/search.form.fr.html. Consulté le 19 Décembre 2012

1.2.1 Domaines et applications

La classification ou la catégorisation des textes (T.C : *Text Classification* ou *topic spotting*) est une discipline au carrefour de deux disciplines importantes : la recherche d'information R.I (*Information Retrieval*) et l'apprentissage automatique (ML : *Machine Learning*) [Seb99]. Comme elle partage un certain nombre de caractéristiques avec d'autres disciplines telles que la fouille de données (*Data Mining*) où la motivation est l'extraction de connaissances à partir de textes (Figure 1.4 [Hab08]).

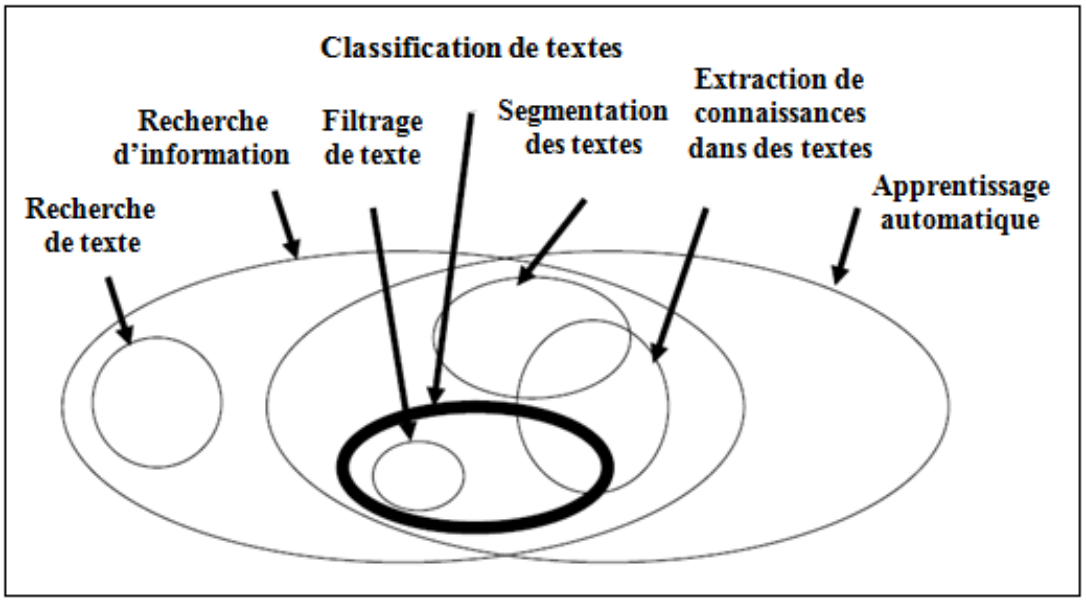

FIGURE 1.4 – La T.C au carrefour d'autres disciplines.

Jusqu'àux années 80, l'approche dominante pour la T.C, était celle basée sur l'expert invoquant l'ingénierie des connaissances (KE : Knowledge engineering). Un expert était chargé de définir manuellement des règles de type *IF <condition> Then <catégorie>* sur la manière de classer et d'extraire l'information. Toutefois avec l'augmentation de la quantité de données au début des années 90, cette approche a cédé le terrain à une nouvelle approche basée sur l'apprentissage automatique qui se caractérise par l'indépendance vis-à-vis du domaine étudié.

La T.C a été considérée pendant longtemps comme un champ de la R.I non seulement parce que les mêmes techniques (stemming, pondération, évaluation ...) sont utilisées dans les deux disciplines, mais aussi parce que le regroupement des documents (*clustering*) a été utilisé pour améliorer la recherche d'informations en se basant sur l'hypothèse qui stipule que les documents ayant un contenu similaire sont aussi pertinents pour la même requête [MRS08]. Ce n'est qu'au début des années 90 que la T.C a été considéré comme un champ de recherches à part entière, et c'est à partir de là où elle a pris de l'ampleur. Le nombre de publication scientifique concernées par cette discipline et les différents domaines ainsi que les différents applications invoquant cette discipline témoignent de cette ampleur.

En effet, les applications de la T.C se sont multipliées et diversifiées allant de l'indexation des documents et des articles scientifiques, au routage d'e-mails, la diffusion sélective d'information pour les faire parvenir aux destinataires en fonction de leurs profils individuels correspondant à leurs centres d'intérêt [LPY94], le filtrage de spam [AKCS00] [CS96], jusqu'a l'analyse des sentiments (*Opinion Mining*) où l'on cherche à analyser des opinions et des attitudes à l'égard de certains sujets ou produits [ES06]. La T.C est aussi utilisée dans d'autres contextes comme la classification hiérarchique (*style Yahoo*) des documents, l'identification de la langue d'un document et le résumé automatique de texte qui peut être vu comme un problème de catégorisation où chaque phrase d'un document doit être étiquetée selon les étiquettes "pertinent" ou "non pertinent" [HIMM02].

1.2.1.1 Définition

La T.C s'intéresse à l'activité d'étiquetage des textes en langage naturelle dans un ensemble prédéfini de catégories thématiques. Elle consiste à construire un classifieur de textes où un système automatique est capable d'assigner des textes dans une (*non-overlapping case*) ou plusieurs (*overlapping case*) catégories ou classes [Seb99]. La T.C est une discipline orientée contenu (content-oriented) ; elle se base uniquement sur le contenu des documents et non sur leurs métadonnées (date publication, auteur,...).

Formellement, la T.C est la tâche qui cherche à affecter une valeur binaire 0,1 à chaque entrée de la matrice représentée par le tableau [Tab 1.1].

	d_1	...	d_j	...	d_n
c_1	a_{11}	...	a_{1j}	...	d_{1n}
...	...	...	...	...	...
c_i	a_{i1}	...	a_{ij}	...	d_{in}
...	...	...	...	...	...
c_m	a_{m1}	...	a_{mj}	...	d_{mn}

TABLE 1.1 – Matrice de décision

Les colonnes $D = d_1, \ldots, d_n$ constituent les documents à affecter, et les lignes $C = c_1, \ldots, c_m$ représentent les labels des catégories prédéfinies. Ces labels sont simplement un texte symbolique (politique, science, sport, culture... etc.) et ils ne fournissent aucune connaissance qui peut être utilisée dans le processus de construction du classificateur.
Cette matrice est appelée *matrice de décision* où a_{ij} correspond à une décision d'affectation : $(a_{ij} = 1)$ ou non affectation $(a_{ij} = 0)$ du document d_j à la catégorie c_i.

1.2.1.2 L'apprentissage automatique pour la T.C

L'apprentissage automatique est devenu depuis les années 90 l'approche dominante pour la T.C [Seb02] pour son efficacité, sa préservation de l'effort de l'expert et sa portabilité d'un domaine vers un autre. La T.C dans ce paradigme est un problème d'apprentissage supervisé où l'on connaît à l'avance les catégories et les instances d'apprentissage pré-classées dans ces catégories.

Dans cette approche, un processus inductif général appelé algorithme d'apprentissage (*Learner*) construit automatiquement un classifieur en apprenant, sur la base d'un ensemble de documents pré-classifiés (par un expert), les caractéristiques des catégories [Seb02]. Ce processus inductif doit s'assurer qu'un nouveau document doit avoir les mêmes caractéristiques que ceux classifiés dans la catégorie où il sera classifié.

Les documents d'apprentissage appartiennent à un corpus initial $D_0 = d_1, \ldots, d_s$ pré-classifiés dans les catégories $c_1, \ldots, c_m$. Pour des raisons d'évaluation du classifieur induit, le corpus initial est divisé en deux sous ensemble de taille quelconque : ensemble d'apprentissage (Training set - $T_r = d_1, \ldots, d_g$) et ensemble de validation ou de test (Test set - $T_e = d_{g+1}, \ldots, d_s$). Dans ce cas la matrice de décision correspond à la matrice représentée par le tableau [Tab 1.2]

	d_1	...	d_g	d_{g+1}	...	d_s
c_1	ca_{11}	...	ca_{1g}	$ca_{1(g+1)}$	...	d_{1s}
...	...	...	...	...	...	...
c_i	ca_{i1}	...	ca_{i_g}	$ca_{i(g+1)}$	...	d_{is}
...	...	...	...	...	...	...
c_m	ca_{m1}	...	ca_{mg}	$ca_{m(g+1)}$	...	d_{ms}

TABLE 1.2 – Matrice de décision pour l'apprentissage supervisé

ca_{ij} correspond à une décision d'affectation : ($ca_{ij} = 1$) ou non affectation ($ca_{ij} = 0$) du document d_j à la catégorie c_i.
Les documents d_j , dont la décision d'affectation est égale à la valeur 1 ($ca_{ij} = 1$), sont désignés par le terme "exemples positifs" de la catégorie c_i. Les exemples négatifs de la catégorie c_i sont ceux qui ont la valeur nulle comme décision d'affection.

1.2.2 Les étapes de la classification de textes

La construction d'un système automatique de classification se fait en trois étapes :

- Le choix de la représentation à adopté et prétraitement des textes.
- L'apprentissage à travers l'algorithme ou la méthode choisi.
- L'évaluation du système construit.

1.2.3 Représentation des données

Dans un système de classification les données doivent être préparées afin d'être utilisées comme entrée d'un système d'apprentissage. Pour cela, il est nécessaire d'avoir à priori une bonne connaissance sur les données, afin de pouvoir mettre l'accent sur les éléments pertinents tout en évitant les informations inutiles. Les éléments pertinents pourront ensuite être exploités lors de la phase d'apprentissage.

La pertinence ne dépend pas uniquement du choix de ces éléments mais aussi des relations (sémantiques) qui peuvent existaient entre eux. Dans la littérature, plusieurs schémas ont été proposés pour mettre l'accent sur cet aspect et pour y parvenir, un système doit d'abord choisir la composante ou l'élément de base (séquence de caractères, mot, phrase) à adopter, ensuite adopter le types de représentation (arbre, automate, transducteur) convenable pour les fins souhaitées.

1.2.3.1 Choix des composantes de la représentation

Les documents textes peuvent être représentés selon les niveaux de granularité suivants :

1. **Représentation basée sur les caractères**
   Dans ce mode de représentation, l'accent est mis sur les séquences de caractères. L'un des modèles les plus connus est le modèle N-grammes basé sur la notion de N-grammes introduite par *Shanon* en 1948 [Sha01]. Dans ce modèle, le document est représenté par un ensemble de termes dont chacun est composé de N caractères. Ces termes sont obtenus en déplaçant une fenêtre de n cases sur le texte et à chaque déplacement on prend une image de cette fenêtre qui constituera un N-grammes.
   Le modèle N-grammes est souvent utilisé dans les domaines comportant des termes composés comme le domaine biomédical dont les noms de molécules sont souvent composés de séquences de caractères désignant une similarité (ou composition) avec d'autres molécules. Le N-grammes est alors capable de capturer ces similarités.

   Ce niveau est relativement primaire, puisque le texte est considéré comme une séquence de symboles dénuée de toutes significations linguistiques. Toutefois, ce modèle présente l'avantage d'être indépendant des langues alors que les systèmes basés sur les mots sont dépendants des langues ("recherche de racine" et lemmatisation sont spécifiques à chaque langue). De même, la plupart des techniques de N-grammes n'exigent pas une segmentation préalable du texte en mots. Ceci est intéressant pour le traitement des séquences ADN par exemple ou les langues où les frontières entre mots ne sont pas fortement marquées comme le Chinois, l'Arabe et l'Allemand. De plus, les mots étant décomposés en sous-chaînes de N caractères, le N-grammes est plus robuste aux erreurs orthographiques [Cav94] et aux déformations liées à l'utilisation de systèmes de reconnaissance optique de caractères (OCR) où la reconnaissance optique est souvent imparfaite [MLS99].

2. **Représentation basée sur les mots**

La représentation d'un document en utilisant uniquement des mots (ou des termes) est probablement la plus utilisée [AE99]. Le modèle de l'espace vectoriel (VSM : Vector Space Model) utilise ce niveau de représentation. Ce modèle a été introduit par *Salton* en 1975[MRS08] pour résoudre des tâches d'indexations pour la recherche documentaire. La figure 1.5 illustre la représentation du VSM.

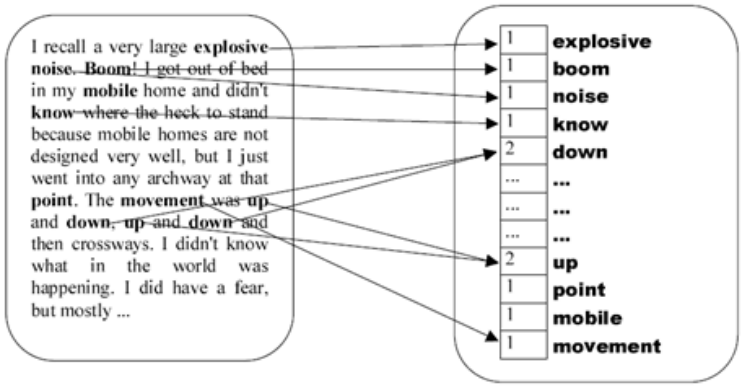

FIGURE 1.5 – La représentation d'un document textuel avec le modèle de l'espace vectoriel.

Bien que nous utilisons une représentation différente, nous présenterons ce modèle pour nous permettre d'illustrer les étapes des prétraitements dans un systèmes de classification.

Dans le VSM, Un document D_k peut être représenté par un vecteur V de termes appelés descripteurs en codant l'ensemble des textes en un tableau "Documents - Descripteurs". Ce tableau est constitué des poids w_{kj} de l'ensemble des descripteurs pour l'ensemble des documents. w_{kj} détermine, par exemple, la fréquence d'occurrence du terme t_i, dans le document D_k ou simplement une valeur binaire indiquant la présence ou l'absence de ce terme dans le document.

Outre les bonnes performances des systèmes utilisant ce modèle, le succès de ces modèles repose sur la simplicité d'encodage des données du fait que la représentation est vectorielle. Ainsi, tous les algorithmes d'apprentissage numérique peuvent être utilisés pour résoudre des problèmes textuels avec cette représentation. Néanmoins cette représentation des textes entraîne une perte d'information car elle exclut toute analyse grammaticale et toute notion de distance entre les mots (la position des mots est ignorée) et c'est la raison pour laquelle cette représentation est appelée "*sac de mots*".

3. **Représentation basée sur les séquences de mots**
   Ce niveau peut être perçu comme un mélange des deux premiers niveaux. Un terme, dans cette représentation, n'est plus un mot unique mais un groupe de mots composé d'un ou plusieurs mots. L'objectif de cette représentation est de tenir compte de l'aspect linguistique des termes car les séquences de mots (ou les phrases) sont plus informatives que les mots seuls. Par exemple : "apprentissage automatique" ou "classification de textes" qui désignent chacun un sens particulier qui peut être perdu si on les décomposent.

Généralement la mise en œuvre d'un système reposant sur ce type de représentation, nécessite des informations linguistiques telles que des lexiques (*lexicon*) ou *thésaurus*. En outre, certains modèles ajoutent des informations morpho-syntaxiques pour affiner le calcul de la similarité en augmentant le degré de détail. Parmi ces modèles, nous citons les arbres syntaxiques et les graphes.
Cette représentation a l'avantage de conserver l'information relative à la position du mot dans la phrase où certains auteurs proposent d'utiliser les séquences de mots comme unité [SHP95], mais les résultats ne sont pas concluants. Cela peut être justifié par la dégradation des qualités statistiques, bien que les qualités sémantiques soient conservées : le grand nombre de combinaisons possibles entraîne des fréquences faibles et trop aléatoires [Lew92].

4. **Représentation sémantique**
   La représentation basée sur la sémantique tente de modéliser le sens induit par le document. Les modèles basés sur cette représentation utilisent des concepts pour désigner les différents sens présents dans le document. Pour permettre une telle représentation les termes sont projetés sur un thésaurus ou un lexicon comme *Wordnet* [Jai04]. L'inconvénient majeur est essentiellement l'extraction des concepts (la définition des concepts et les relations d'association entre concepts et termes linguistiques). Encore les bases lexicale n'existent pas pour toutes les langues pour permettre une telle représentions [Réh11].

1.2.3.2 Modèle de représentation

Nous avons cité dans la section précédente que certaines représentation des données textuelles nécessitent des modèles spécifiques tels que les arbres syntaxiques et les graphes pour permettre une prise d'un niveau plus riche et plus précis de l'information contenue dans ce type de données.
Nous allons maintenant présenter quelques modèles utilisés dans le cadre des problématiques liées aux données textuelles.

1. **Arbres syntaxiques**
   Il est courant de considérer, pour des fins différentes, un objet textuel sous forme d'un ou plusieurs arbres. Les arbres d'analyse syntaxique constituent un exemple très répandu.
   Un arbre d'analyse syntaxique représente la structure syntaxique d'une phrase. Il est obtenu en décomposant une phrase en groupe grammatical. La figure 1.6 montre l'arbre d'analyse syntaxique associé à la phrase " Jeff mange la pomme". Cette représentation est souvent employée par les *analyseurs morphologiques* pour réaliser un étiquetage morpho-syntaxique de chaque mot d'un document ou d'un corpus.

2. **Modèle de Markov caché**
   Les modèles probabilistes sont une famille de modèle qui permettent de prendre en compte la séquentialité au sein des textes contrairement aux modèles *sac de mots*. Ils reposent sur l'hypothèse de dépendances à horizon borné entre occurrences de vocables. Toutefois,

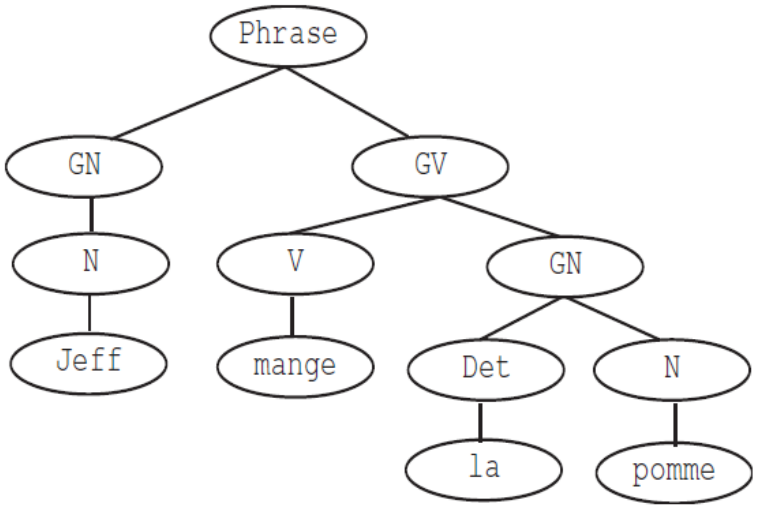

FIGURE 1.6 – Exemple d'un arbre syntaxique.

la probabilité d'occurrence d'un vocable n'est influencée que par les quelques vocables qui le précèdent, le cas le plus simple étant celui d'une dépendance d'ordre 1. Ce cas est connu par les *chaînes de Markov*. Les chaînes de Markov représentent les dépendances temporelles dans une séquences de variables $S_0, \ldots, S_t, \ldots$ où chaque variable ne dépend que de la variable précédente dans la séquence ($P(S_t/S_0 \ldots S_{t-1}) = P(S_t/S_{t-1})$). Ils sont parfaitement adapté pour la représentation de séquences, qu'elles soient de nature temporelle (signal acoustique, cours boursier ...) ou spatiale (chaîne d'ADN, séquence de caractères ou de mots ...) [CM02]. Les modèles de Markov cachés MMC (*Hidden Markov Models*)[4] représentent une généralisation des chaînes de Markov.

Formellement un MMC (d'ordre un) est un modèle génératif de séquence défini par un ensemble d'état, un alphabet discret de symboles, une matrice de probabilités de transitions entre états et une matrice de probabilités d'émission de chaque symbole de l'alphabet à partir de chaque état. Le système évolue aléatoirement d'un état à l'autre suivant les probabilités de transition en émettant des symboles de l'alphabet.

Grâce aux MMC plusieurs problèmes classiques peuvent être traités, par exemple, la quantification de la probabilité d'une observation ou d'une séquence quelconque ou l'identification de la suite d'états maximisant la probabilité d'une séquence donnée. La figure 1.7 montre une représentation sous forme graphique d'un MMC modélisant les 3-grammes d'un corpus d'apprentissage simple "aabaacaab".

3. **Transducteurs**

   Un transducteur (*transducer*) ou son cas particulier transducteur pondéré à états finis (weighted finite state transducer) (WFST) est un graphe qui représente un ensemble de séquences en entrée, et leur associe, selon un poids donnée, des séquences en sortie. Les transducteurs présentés ainsi sont une généralisation des automates à états finis. Ils offrent aussi une représentation naturelle et commune des MMC [MPR00]. Nous reviendrons sur

4. Appelé aussi automate de Markov à états cachés. Appelé ainsi parce que seuls les symboles émis sont observables, et non les transitions entre états qui sont internes au modèle.

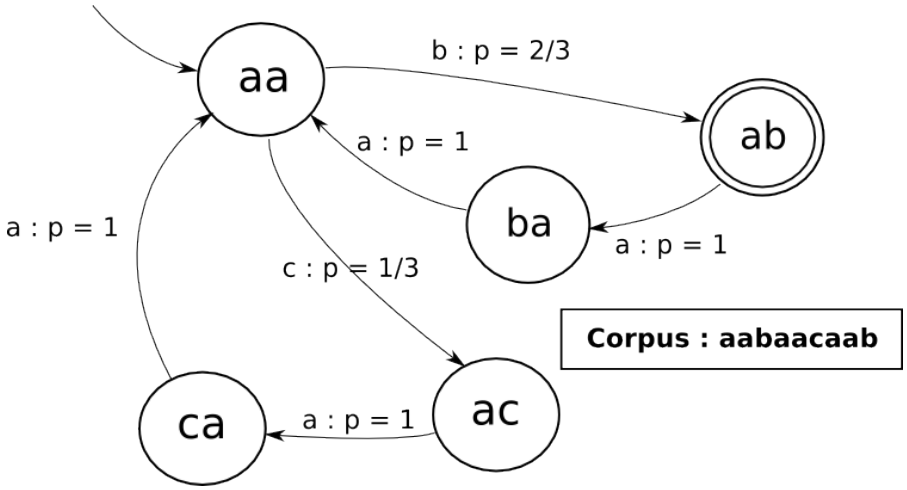

FIGURE 1.7 – Exemple d'un MMC modélisant les 3-grammes du corpus "aabaacaab".

la définition formelle des transducteurs dans la section 4.5.1.

L'utilisation des transducteurs a couvert un spectre d'application très large à l'instar de la reconnaissance de paroles, traduction automatique, reconnaissance de caractère optique (OCR), modèles de prononciation et dans quelques problématique liées aux données textuelles (*tokenization*, *stemming*) [CM09]. La figure 1.8 [5] montre un exemple d'un transducteur qui représente les différentes prononciations possibles du mot "data".

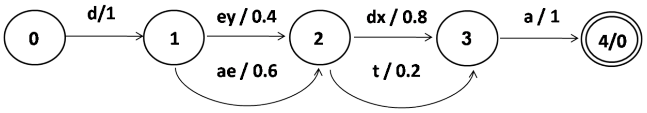

FIGURE 1.8 – Modèle de prononciation du mot "data" implémenté sous forme de transducteur.

Dans le cadre du traitement automatique de la langue naturelle, les transducteurs ont été adoptés dans les travaux de *Nehar* et al. [NZCG12] pour réaliser un *racineur* pour la langue arabe qui peut être intégré dans un système de T.C par exemple (voir 2.2.1).

Lorsque les transducteurs (ou les automates) sont adoptés pour représenter des documents textes certains noyaux (fonction de similarité) ont été développé permettant le calcul de similarité entre deux transducteurs (ou deux automates) de façon efficace [CHM+04]. Ces noyaux sont connus par les *noyaux rationnels* (voir la définition 4.5.2.1).

1.2.4 Prétraitement des données

Le principal enjeu de la catégorisation de textes réside dans la recherche des termes ou des descripteurs les plus pertinents pour le problème à traiter [AE99]. Différentes méthodes sont proposées pour le choix des termes à extraire, leurs normalisation, leurs pondération et éventuellement de réduction de l'ensemble de ces descripteurs.

5. http ://www.openfst.org/twiki/pub/FST/FstSltTutorial

1. **Extraction des termes**
   Dans cette phase on cherche :
   - à segmenter les documents en unités "élémentaires" désignées par "*tokens*". Cette phase utilise les signes de la ponctuation et les listes des séparateurs pour en extraire les termes d'un document.
   - à supprimer les mots outils appelés aussi mots vides (*StopList* ou encore *Common Words Removal*) comme les mots grammaticaux (les pronoms, les articles), les verbes introductifs, les verbes d'état ... qui sont des mots trop fréquents mais non utiles.

   Cette phase permet de réduire considérablement l'espace total de représentation. Elle dépend de la langue où certaines langues qui ne séparent pas les mots par des espaces posent des difficultés supplémentaires (comme le japonais et le chinois), et les langues qui s'écrivent de droite à gauche (l'Arabe et l'Hébreu) avec certains éléments qui s'écrivent de gauche à droite (comme les nombres). Plus de détails concernant les difficultés liées à la langue arabe seront exposées dans la section 2.1.7.

2. **Normalisation linguistique**
   C'est un processus morphologique permettant de regrouper les variantes d'un mot en ramenant plusieurs éléments à une même forme dans le but de réduire l'espace de représentation des documents et faire ressortir les similarités entre les mots. Ce processus morphologique peut être réalisé par la lemmatisation ou la radicalisation des mots.

   (a) **Radicalisation (*Stemming*) :** La radicalisation ou la dé-suffixation consiste à rechercher les racines lexicales grâce à l'étude morphologique des mots. Elle se base sur un dictionnaire d'affixes (suffixe et préfixe) qui permet d'extraire le radical d'un mot en supprimant tous ses affixes.
   Pour la recherche des racines lexicales, plusieurs algorithmes ont été proposés, le plus célèbre est l'algorithme de *Porter* [Por80] qui a été transposé de la langue anglaise vers plusieurs langues européennes.

   (b) **Lemmatisation :** La lemmatisation d'une forme d'un mot consiste à reprendre sa forme canonique : pour un verbe c'est son infinitif ; le masculin singulier pour les autres mots. Ainsi plusieurs formes variées selon leur genre, leur nombre,... seront réduit à une seule forme : leur lemme. L'analyse syntaxique est parfois insuffisante ; car de nombreux mots de même graphie peuvent provenir de différents lemmes. Donc en plus d'une analyse syntaxique, une analyse morpho-syntaxique est parfois nécessaire pour résoudre les ambiguïtés.

Stemming Versus Lemmatisation Les algorithmes de stemming sont beaucoup plus simplistes et mécaniques car ils utilisent des règles simples, propres à une langue, de flexion des mots pour trouver la base commune. Ils sont donc plus rapides, mais leurs précisions et leurs qualités sont naturellement inférieures [Gué06]. Alors que la lemmatisation repose sur un lexique et demande une analyse plus complexe puisqu'elle repose sur des contraintes linguistiques beaucoup plus fortes.

Remarque

Dans la littérature, on confond souvent la radicalisation (stemming) et la lemmatisation. Dans la suite de ce mémoire on considère toute manière de réduire ou de grouper les formes d'un mot dans une même classe comme un processus de stemming. Nous utiliserons, sauf indication, le terme stemming (stem) indifféremment pour designer la radicalisation (radical) ou la lemmatisation (lemme) d'un mot.

3. **Pondération des mots normalisés**

La pondération est l'une des fonctions fondamentales de la T.C. Elle repose sur l'idée que le poids d'un terme dans un document traduit son importance pour ce document. Dans la phase de prétraitement et avec la représentation en *sac de mots*, chaque document est représenté par un vecteur de n termes pondérés. Les poids attribués aux termes peuvent être tout simplement binaires comme ils peuvent représenter le nombre d'occurrences du mot dans le document.

D'une façon générale l'association d'un poids à un terme obéit à l'un des schémas suivants :

– pondération locale ;
– pondération globale ;
– ou la combinaison des deux.

La première quantifie la représentativité locale d'un terme dans le document, et la deuxième quantifie la représentativité du terme vis-à-vis de toute la collection.

(a) **Fréquence du terme (*TF*)**

la Fréquence du terme TF (Term Frequency) est une mesure qui indique l'importance du terme dans le document. Elle est proportionnelle à la fréquence du terme dans ce même document : plus ce terme apparaît plus il est représentatif. TF (ou le poids W_{ji}) du terme t_i dans le document d_j est donné par :

$$TF(t_i, d_j) = \sharp(t_i, d_j)$$

$\sharp(t_i, d_j)$ dénote la fréquence ou le nombre de fois où le terme t_i apparaît dans d_j.

(b) **Fréquence inverse de document (*IDF*)**

IDF (Inverse of Document Frequency) calcule l'importance d'un terme dans toute la collection. Elle traduit l'impact d'un terme selon son nombre d'apparition dans la base documentaire afin de tenir compte de la spécificité d'un terme pour un document.

$$IDF(t_i, T_r) = \log(\frac{/T_r/}{\sharp(t_i)})$$

$|T_r|$ représente la taille de la collection d'apprentissage et $\sharp(t_i)$ est le nombre de documents où t_i apparaît.

(c) **Term Frequency-Inverted Document Frequency (*TF-IDF*)**

La pondération locale-globale est utilisée pour décoder si un terme est représentatif

pour un document (TF) : donner plus d'importance aux termes qui apparaissent souvent à l'intérieur d'un même document , mais aussi s'il est discriminant pour ce document dans l'ensemble de la collection ; (IDF) : donner moins de poids aux termes qui appartiennent à plusieurs documents, pour refléter le fait que ces mots ont un faible pouvoir de discrimination. La fonction de pondération *TF-IDF* a démontré une bonne efficacité dans les tâches de catégorisation de textes [Seb99]. Elle est calculée comme suit :

$$W_{(j,i)} = TF\text{-}IDF(t_i, d_j) = \sharp(t_i, d_j).\log(\frac{/T_r/}{\sharp(t_i)})$$

Le schéma de pondération *TF-IDF* réalise un équilibre entre deux objectifs contradictoires bien connu par la dualité de l'*exhaustivité* et *spécificité*. Le premier, tend à ce que la description du document soit la plus complète possible, alors que le deuxième consiste en une meilleure différenciation entre les descriptions des documents pour mieux les distinguer. Une bonne méthode de pondération doit établir un compromis entre ces deux aspects [GJZ03].

Le Tableau [Tab 1.3] reprend d'autres fonctions qui reviennent souvent dans la littérature [Seb02],[LK02],[BSAS94] :

Fonction	Formule
log TF	log($TF\ (t_i,d_j)\ +\ 1$)
log *TF-IDF*	log($TF(t_i,d_j)+1) \cdot IDF(t_i,T_r)$
ITF Inverse Text Frequency	[1- (1 /($TF\ (t_i,d_j\ )\ +\ 1$))]

TABLE 1.3 – Quelques fonctions usuelles de pondération.

L'idée qu'à partir d'une certaine fréquence, les termes ont approximativement la même importance, a donné lieu à la pondération ITF [LK02]. cette dernière permet de réduire l'influence des termes trop fréquents. Par conséquent, le poids accordé par l'ITF croît très rapidement pour les fréquences faibles et moyennes et se stabilise pour les fréquences élevées.

4. **Réduction de dimension**

Bien que l'élimination des mots vides et la normalisation permettent de réduire l'espace de représentation d'une manière significative [6], ce dernier conservera toujours une dimension importante. En effet avec la représentation en *sac de mots*, chacun des mots d'un corpus est un descripteur potentiel, or pour un corpus de taille raisonnable, ce nombre peut être de plusieurs dizaines de milliers.

6. C'est la raison pour la quelle la normalisation linguistique des mots peut être perçue comme une manière de réduction de dimension.

On retient, généralement pour les traitements ultérieurs, un pourcentage ou un nombre de termes (triés par ordre décroissement) pondérés selon un schéma de pondération donné ou une fonction statistiques.

La réduction de dimension, permet aussi de limiter le bruit ou encore d'éviter le sur-apprentissage. Dans ce dernier cas, la solution passe par la limitation du nombre de descripteurs en fonction du nombre d'exemples dans l'échantillon d'apprentissage [FB91] mais comme on ne dispose que d'un nombre limité d'exemples d'apprentissage, on tend à réduire le nombre des termes utilisés pour éviter ce problème.

Les techniques utilisées pour la réduction de dimension sont issues de la théorie de l'information et de l'algèbre linéaire. Elles peuvent être classés selon le résultats de la sélection [Seb99] :

(a) **Extraction d'attributs "feature extraction"**
L'extraction d'attribut vise à proposer un sous ensemble T_0 (avec $|T_0| \ll |T|$) de variables artificielles crées à partir d'attributs de départ. Ces variables synthétiques sont créées par une combinaison linéaire des descripteurs en faisant soit des regroupements, soit des transformations qui visent à éliminer les problèmes liés aux synonymies, polysémie et homonymies.

L'analyse par sémantique latente "LSA" [DDL$^+$90] et le regroupement des termes (*Term clustering*) constituent les principales méthodes dans cette approche.

(b) **Sélection d'attributs " Feature selection "**
Dans cette approche on cherche un nouvel ensemble T_0 (avec $|T_0| \ll |T|$) de termes réduits porteurs de plus d'information. Les termes moins discriminants seront supprimés : par exemple les mots les plus fréquents (présents partout) puisqu'ils n'apportent pas d'information. De même, les mots très rares seront supprimés, car leurs faibles fréquences ne permettent pas de construire des règles stables (pour éviter d'avoir un modèle spécifique au corpus). Les méthodes de sélection de descripteurs fournissent en général, une liste de descripteurs triés par score, ce qui nécessite généralement le choix d'un seuil (ou un nombre) à partir duquel les termes importants seront retenus.

Pour le calcul de ces scores plusieurs statistiques ont été développées. Le tableau [Table 1.4] reprend celles les plus usuelles de ce domaine [YP97], [Seb99], [Seb02].
L'information mutuelle (*I.M*) permet de quantifier le degré de dépendance statistique entre un terme et une classe. L'*I.M* d'un terme est jugée élevée pour les termes apparaissant souvent dans la catégorie et moins élevée pour les termes apparaissant en dehors de la catégorie. En effet, lorsque le terme t_k et la classe c_i sont considérés indépendants ($P(c_i, t_k) = P(c_i)P(t_k)$), l'*I.M* dans ce cas est nulle (la

Fonction	Notation	Formule
L'information mutuelle	$\mathrm{MI}(t_k, c_i)$	$\log(P(t_k, c_i)/P(t_k).P(c_i))$
Gain d'information	$G.I(t_k, c_i)$	$\sum_{c\in(c_i,\overline{c_i})} \sum_{t\in(t_i,\overline{t_i})} P(t, c).(\log(P(t, c)/P(t_k).P(c_i)))$
Chi-square	$\chi^2(t_k, c_i)$	$\frac{\lvert T_r\rvert[P(t_k, c_i).P(\overline{t_k}, \overline{c_i}) - P(t_k, \overline{c_i})P(\overline{t_k}, c_i)]^2}{P(t_k).P(\overline{t_k}).P(c_i).P(\overline{c_i})}$
NGL Coefficient	$NGL(t_k, c_i)$	$\frac{\sqrt{(\lvert T_r\rvert)}[P(t_k, c_i).P(\overline{t_k}, \overline{c_i}) - P(t_k, \overline{c_i})P(\overline{t_k}, c_i)]^2}{\sqrt{P(t_k).P(\overline{t_k}).P(c_i).P(\overline{c_i})}}$
Odds Ratio	$OR(t_k, c_i)$	$\frac{P(t_k/c_i).(1 - P(t_k\overline{c_i}))}{(1 - P(t_k/c_i))P(t_k/\overline{c_i})}$

TABLE 1.4 – Quelques fonctions utilisées pour la sélection d'attributs.

présence de t_k dans un document n'apporte aucune information sur le degré d'appartenance du document à la catégorie c_i).

Le gain d'information (*G.I*) ne diffère de l'information mutuelle que par la prise en compte de l'information apportée par l'absence d'un terme pour une catégorie et non uniquement à sa présence : l'absence d'un terme précis dans certains documents peut aussi être liée à une classe. Le *G.I* peut être vue comme l'espérance de l'*I.M* et est souvent mis en pratique dans les arbres de décisions.

Chi-deux (χ^2) est une technique purement statistique, tandis que les deux premières sont issues de la théorie d'information. Comme l'*I.M* et *G.I*, il permet de déterminer le degré d'indépendance entre un terme et une classe. Une valeur nulle indique que le terme et la classe sont indépendants.

Mais la question importante est de savoir déterminer à partir de quelle valeur on élimine ou on conserve un terme ; c.à.d. définir le seuil où les termes seront considérés comme dépendants ou indépendants vis-à-vis une classe.
Cela dépend généralement du modèle ; les *SVM* par exemple sont capables de manipuler des vecteurs de grandes dimensions (de l'ordre de 9.962 termes distincts) [Joa98] à l'inverse des réseaux de neurones. Malgré cela, une réduction de l'espace des entrées (à 300 entrées) donne

des meilleurs résultats [DPHS98] pour le même corpus. Les résultats dépendent de la façon dont les différents algorithmes gèrent les corrélations entre les descripteurs. Mais généralement la précision de la classification s'accroît en fonction d'un nombre de descripteurs réduit et bien choisi [Str00]. Néanmoins le nombre de termes doit être réaliste concernant un domaine tel que la T.C [KS96] [YP97].
Aussi les métriques employées pour la sélection des termes jouent un rôle important. Selon Yang [YP97] les méthodes les plus efficaces et les plus souvent utilisées pour *Rocchio* et les *K-ppv* sont l'*I.M* et le "χ^2".

Quant au choix de la méthode de sélection, il a été constaté que la sélection d'attributs est meilleure pour éliminer les attributs inutiles ou erronés "*noisy*", tandis que l'extraction d'attributs est un choix à considérer pour la réduction du nombre d'attributs redondants [Réh11].

1.2.5 Construction du classifieur

Les algorithmes d'apprentissage supervisé tentent de trouver un modèle, une fonction mathématique, qui explique le lien entre les données d'entrée et les classes de sortie fouinées comme base d'apprentissage. Ce type d'apprentissage essai de déduire un ensemble de règles lui permettant une telle décision. Cette méthode de raisonnement est appelée inductive (ou induction supervisée), car on induit de la connaissance (le modèle) à partir des données d'entrée (l'échantillon de documents et leurs classes). Grâce à ce modèle, on peut alors déduire ou prédire les classes des nouvelles données.
La construction inductive d'un classifieur pour une catégorie $c_i \in C$, consiste habituellement en la définition d'une fonction f telle que, pour un document d_j, elle retourne une valeur indiquant l'état de catégorisation (**CSV** : Categorization Status Value) qui représente le fait que $d_j \in c_i$ ou non.

On parle d'une construction automatique " *hard* " et d'une construction semi-automatique " *ranking* " selon la valeur envoyée par la fonction f. La définition de f pour un classifieur de type " *ranking* " est $f : D \rightarrow [0;1]$ qui retourne, pour une classe $c_i \in C$, une valeur comprise entre 0 et 1 pour chaque document à classer. Cette valeur est ensuite interprétée selon la méthode d'apprentissage utilisée : pour le classifieur Naïve Bayes, c'est une estimation de la probabilité que le document appartient à la classe c_i et pour la méthode de Rocchio, c'est la proximité du document à classer au profil " prototypique " de la classe c_i dans l'espace de représentation.

La définition de f pour un classifieur de type " hard " est $f : D \rightarrow \{V;F\}$ qui retourne, pour chaque document à classifier, une valeur parmi deux valeurs possibles ; $V(vrai)$ si le document appartient à la classe c_i ou $F(faux)$ sinon.
Toutefois il est possible de transformer une construction semi-automatique en une construction automatique en introduisant un paramètre seuil (*threshold*) T_i tel que $CSVi(d_j) \geq T_i$ est

interprété comme *vrai* et *faux* pour l'autre cas.

Dans la suite de cette section nous nous contenterons de présenter les méthodes les plus utilisées dans la littérature et comment ces méthodes sont utilisées dans le cadre de la classification de textes.

• **Rocchio**

Le classificateur de Rocchio [Roc71] est l'un des plus simples et des plus ancien algorithmes d'apprentissage supervisé [Mit97]. Il se distingue par sa rapidité d'apprentissage et de classification grâce au fait qu'il nécessite très peu d'espace mémoire.

Rocchio est un algorithme de type orienté-profil " *Profile-based* " ; il calcul un profil "prototypique" pour chaque classe. Les profils sont des listes de termes pondérés, dont la présence et l'absence discriminent au mieux ces classes. Le profil de la catégorie $c_i = (W_{i1}, W_{i2}, \ldots, W_{in})$ est donné par :

$$W_{ik} = \beta . \sum_{d_j \in POS_i} \frac{W_{jk}}{|POS_i|} - \gamma . \sum_{d_j \in NEG_i} \frac{W_{jk}}{|NEG_i|}$$

W_{jk} représente le poids du terme t_k dans le document d_j, $|POS_i|$ est la cardinalité de l'ensemble des exemples positifs de la catégorie c_i, et $|NEG_i|$ est la cardinalité de l'ensemble des exemples négatifs.

β et γ sont les paramètres utilisés pour contrôler l'impact des exemples respectivement positifs et négatifs. Généralement le rôle des exemples positifs est accentué par rapport aux exemples négatifs en attribuant à β une valeur élevée. Pour le cas $(\beta = 1, \gamma = 0)$, les profils des classes coïncident avec les barycentres de leurs exemples positifs.

Le classification d'un nouveau document s'opère en calculant la distance entre la représentation vectorielle de ce document et le profil de chacune des classes ; le document est assigné à la classe la plus proche.

Rocchio représente une caractéristique intéressante : il est peu sensible aux bruits ; même avec 50% des exemples bruités, ses performances sont presque inchangées avec des exemples moins bruités [Réh11]. Cependant Rocchio devrait donc être peu adaptée quand la séparation des classes n'est pas linéaire ; due à sa nature qui implémente une règle de décision qui dessine des séparations linéaires (hyperplans) dans l'espace de représentation des textes [Jal03].

• **Naïve Bayes**

Le modèle *N.B* "*Naïve Bayes*" ou simple Bayes est un modèle probabiliste, très rapide et réalise un bon compromis vitesse-qualité [CM02]. Dans ce modèle, la fonction $CSV_i(d_j)$ est interprétée en termes de la probabilité $P(c_i/\overrightarrow{d_j})$ i.e la probabilité d'appartenance à une classe c_i pour un

document représenté par le vecteur $\overrightarrow{d_j} =< w_{1j}, \ldots, w_{nj} >$. Il calcule cette probabilité pour toutes les classes en utilisant le théorème de Bayes, comme suit :

$$P(c_i/\overrightarrow{d_j}) = P(c_i).P(\overrightarrow{d_j}/c_i)/P(\overrightarrow{d_j}) \tag{1.2}$$

Un document candidat est assigné à la classe qui maxime cette probabilité.
Les probabilités $P(c_i)$ et $P(\overrightarrow{d_j}/c_i)$ sont estimées à partir d'exemples d'apprentissage :

$$P(c_i) = (|docs_i|)/(|Tr|) \tag{1.3}$$

$$P(\overrightarrow{d_j}/c_i) = P(< w_{1j}, \ldots, w_{nj} >)/c_i \tag{1.4}$$

$|docs_i|$ représente le nombre de documents de la catégorie c_i ; et $|Tr|$ est la taille de la collection d'apprentissage.

Quant à la probabilité $P(\overrightarrow{d_j})$ de (1.2) ; elle est constante pour un document donnée ; elle est donc ignoré puisqu'elle n'a aucun effet sur la comparaison.

Le calcule de (1.4) pose des difficultés. Pour le simplifier , il est courant de faire l'hypothèse que tous les termes du document sont indépendants, d'où la qualification "*Naïve*" ou "*Simple*" pour ce modèle. Bien que cette hypothèse n'est pas réaliste, elle donne des bonnes résultats expérimentaux en T.C [Seb99]. Sous cette hypothèse (1.4) devient :

$$P(\overrightarrow{d_j}/c_i) = \prod_{k=1}^{n} P(w_{kj}/c_i) \tag{1.5}$$

N.B présente d'autres modèles comme le modèle multinomial et le modèle multivarié de Bernoulli [MN98]. Les réseaux Bayésiens constituent une autre variante de ce modèle. Dans ce dernier cas les variables ne sont pas considérés toutes indépendantes ; ils autorisent certaines à être liées. Cela n'augmente pas les résultats de façon significative, par contre il alourdit considérablement les calculs [DPHS98].

- **Les k plus proches voisins " K-ppv "**

Classé comme un algorithme basé sur les exemples "*Based-examples*", cet algorithme n'induit aucun modèle à partir des exemples, ce qui implique un temps d'apprentissage nul.

L'idée de base est d'inférer la classe d'un nouvel exemple en choisissant la classe majoritairement représentée par ses (K) plus proches voisins en fonction leurs distances. Pour qu'une décision d'affectation, d'un document d_j dans une catégoriser c_i doit être prise ; l'algorithme doit prendre en considération si les documents les plus similaire sont classifiés dans cette catégorie.

Donc, il utilise deux paramètres : le nombre (K) et la *fonction de similarité* pour comparer le nouveau cas aux cas déjà classés ; des résultats très différents découlent de leurs choix [Jal03].

Généralement le paramètre (K) est déterminé expérimentalement. Concernant les fonctions de similarité utilisées pour le calcul de la distance on peut citer : *Jaccard* , *Cosinus* , *Dice* , *produit scalaire* , *distance euclidienne*, ...

Bien que la fonction de similarité soit coûteuse puisque la méthode a besoin de comparer le nouveau cas à tous les exemples classés dans une catégorie donnée, l'algorithme est d'une complexité linéaire [YL99].

Des versions différentes ont été proposées pour enrichir et atténuer les inconvénients majeurs de la méthode de base. Parmi ces versions la variante *Category-Based Search*" [TI95] qui consiste à représenter tous les documents classés dans une catégorie par un cas unique (le représentant). On cherche donc le représentant le plus proche du texte à classifier. La comparaison se fait dans ce cas, uniquement avec les représentants des catégories et non avec tous les documents deux à deux. "*cluster-based search*" [SM84] constitue une autre variante. Elle peut être considéré comme une combinaison d'algorithmes orienté-profil " *oriented-profile*" et ceux basés sur les exemples "*Based-examples*". Cette variante utilise un algorithme de classification non supervisée qui regroupe les documents dans des catégories non prédéfinies et fait ressortir un représentant de chacune de ces catégories. La classification d'un nouveau document, se fait ainsi, par rapport aux représentants de chaque classe automatiquement découverte et non par rapport aux classes prédéfinies. L'intérêt réside dans le fait qu'un document peut être classifié dans plusieurs catégories ; puisque les documents classifiés automatiquement dans un cluster n'ont pas forcément la même catégorie de départ.

- **Les arbres de décisions**

L'expressivité et la lisibilité ont fait des arbres de décision un ensemble d'algorithmes (*CART*, *ID3*, *C4.5*, *CHAID*,... etc.) très utilisés depuis de nombreuses années dans le cadre de l'apprentissage supervisé [Mit97]. Ils peuvent traiter aussi bien des données représentées par des attributs quantitatifs, des attributs qualitatifs, ou des représentations composites.

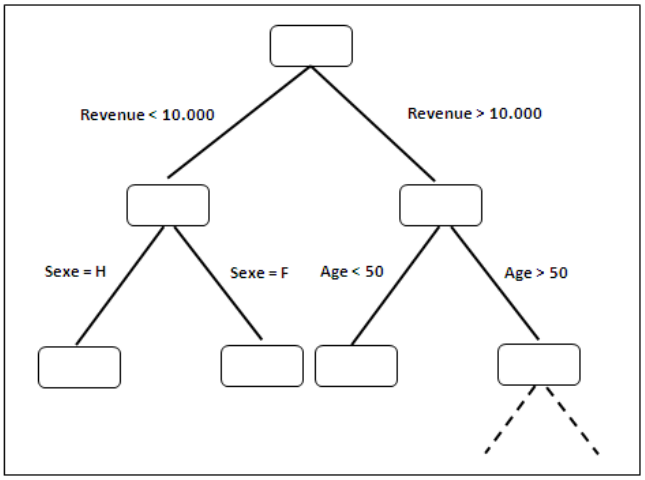

FIGURE 1.9 – Exemple d'un arbre de décision avec des attributs quantitatifs et qualitatifs.

Les arbres de décision produisent un processus décisionnel facilement compréhensible et exploitable par un humain. En effet, le résultat est un arbre où chaque nœud est une règle

de la forme *IF ... THEN* qui conditionne l'exploration d'un seul sous-arbre correspondant à la branche satisfaisant la condition. Cela dans un processus récursif jusqu'au nœud terminal (feuille) contenant la décision finale (les valeurs *Oui* ou *Non* dans le cas d'un arbre de décision représentant une catégorie donnée pour déterminer si un document sera affecté ou non à celle-là).

L'avantage majeur des arbres de décisions est qu'ils n'utilisent qu'un sous ensemble des attributs initiaux puisqu'ils n'exploitent dans chaque règle de décision (dans chaque nœud) qu'un seul attribut à la fois. En effet, le principe d'inférence utilisé dans la construction d'un arbre de décision suit un principe simple : plus on descend en profondeur dans l'arbre et plus la probabilité qu'un exemple soit d'une classe donnée est accentuée (les attributs les plus discriminants seront choisis en premier). Pour la détermination de ces attributs, plusieurs métriques ont été utilisées pour mesurer la qualité de la séparation réalisée par chaque nœud (dans quelle mesure cet attribut sépare bien les exemples). La méthode *ID3* utilise le *Gain d'information* qui se base sur le calcul de l'entropie.

Il existe d'autres extensions de l'algorithme *ID3* permettant d'utiliser des exemples où les valeurs des attributs avec des valeurs continues (par exemple lorsqu'on utilise *TF-IDF* au lieu des valeurs binaires). *C4.5* est une autre extension des arbres de décision où les conditions sont éliminées progressivement des décisions écrites sous forme $< IF \ldots THEN >$. On n'élimine une condition que si elle ne décroît pas la capacité de l'arbre à classer les exemples.

- **Les Séparateurs à vaste marge**

Les *SVM* ont été développés dans les années quatre-vingt-dix à partir des considérations théoriques de Vladimir Vapnik sur le développement d'une théorie statistique d'apprentissage : la théorie de Vapnik-Chervonenkis [Vap95] et la théorie de l'apprentissage de Valiant [Val84][7]. Depuis, elles ont connue leurs plein essor dans plusieurs tâches d'apprentissage : reconnaissance de caractères manuscrits, détection de visages, classification de protéines, classification de textes. L'intérêt d'une telle approche est incité par les fondements solides de la théorie statistique d'apprentissage de Vapnik-Chervonenkis expliquant la qualité des nombreux résultats expérimentaux obtenus [CST10].

C'est l'intérêt de ce travail où l'on cherche à les utiliser pour un système de classification de textes en langue arabe. Nous reviendrons sur ces méthodes dans le chapitre 3.

1.2.6 Évaluation

L'évaluation d'un classifieur est réalisée expérimentalement. Le paramètre d'efficience reposant sur les critères du temps de réponse et de l'espace utilisé (qui dépendent de la technologie utilisée et qui évolue dans le temps) pour le stockage d'information semble être moins importante lors de l'évaluation. Donc l'évaluation se contente de de mesurer l'efficacité d'un classifieur

7. Valiant Leslie (prix Turing en 2010) est connu par ses travaux sur la complexité de l'apprentissage automatique.

i.e sa capacité à prendre la bonne décision de catégorisation.

1. **Mesures d'évaluation**
   L'efficacité d'un système est évaluée avec les mesures conventionnelles utilisées dans la R.I qui ont été adaptées à la T.C. ces mesures permettent de caractériser un modèle ou un classifieur. Toutes les mesures usuelles se basent sur les tableaux de contingence [Table 1.5],[Table 1.6].

Catégorie c_i		**jugement de l'expert**	
		OUI	NON
Décision du classifieur	OUI	VP_i	FP_i
	NON	FN_i	VN_i

TABLE 1.5 – Table de contingence pour la catégorie c_i

Catégorie c_i		**jugement de l'expert**	
		OUI	NON
Décision du classifieur	OUI	$\sum_{i=1}^{m} VP_i$	$\sum_{i=1}^{m} FP_i$
	NON	$\sum_{i=1}^{m} FN_i$	$\sum_{i=1}^{m} VN_i$

TABLE 1.6 – Table globale de contingence

- VP_i : Vrai Positif; le nombre de documents correctement attribués à la classe c_i ;
- VN_i : Vrai Négatif; le nombre de documents correctement non attribués à la classe c_i ;
- FP_i : Faux Positif; le nombre de documents faussement attribués à la classe c_i ;
- FN_i : Faux Négatif; le nombre de documents faussement non attribués à la classe c_i ;

Les performances d'un classifieur sont souvent mesurées via la *Précision* (*Precision*) et le *Rappel* (*Recall*).

(a) ***Rappel* et *Précision***
Pour un système de classification, la *Précision* ou le taux de documents pertinents, mesure uniquement le degré de pertinence des documents classés comme instances positives. Elle est définie par le rapport entre le nombre de documents bien classés dans une catégorie et le nombre total de documents classés, par le système, dans la même catégorie. En utilisant la matrice de contingence on peut définir la *précision* par :

$$Pr_i = VP_i/(VP_i + FP_i)$$

Quant au *Rappel*, il mesure la complétude du système. Il exprime le taux des documents pertinents extraits par le système par rapport au nombre total de documents pertinents. Il s'agit du nombre de documents correctement classés dans une catégorie sur le nombre total de documents appartenant à cette catégorie. Formellement, nous avons :

$$Re_i = VP_i/(VP_i + FN_i)$$

Pour obtenir l'estimation relative à l'ensemble des catégories, deux méthodes peuvent être adoptées :

i. **Micro-moyenne** : obtenue par la sommation globale des décisions individuelles :

$$Pr^u = VP/(VP + FP) = \sum_{i=1}^{m} VP_i / \sum_{i=1}^{m} (VP_i + FP_i)$$

$$Re^u = VP/(VP + FN) = \sum_{i=1}^{m} VP_i / \sum_{i=1}^{m} (VP_i + FN_i)$$

Où m est le nombre de catégorie.

ii. **Macro-moyenne** : la *Précision* et le *Rappel* sont localement évalués sur chaque catégorie, ensuite globalement en moyennant les résultats de chaque catégorie.

$$Pr^M = \sum_{i=1}^{m} Pr_i/m$$

$$Re^M = \sum_{i=1}^{m} Re_i/m$$

Généralement la Micro-moyenne est retenu parce que les catégories doivent être considérer proportionnellement à leurs fréquences [Seb99].

D'autres méthodes sont employées pour calculer la moyenne du *Rappel* et de la Précision comme la méthode des *11 points de moyenne* (*11-point average precision*) [MRS08]. Dans cette méthode la moyenne est calculée à partir de la meilleure valeur dans chaque tranche de *Rappel* variant de 0 à 1 par incrément de 0.10 et par extrapolation. Une autre méthode dite *Breakeven point* consiste à trouver le couple (*Précision*, *Rappel*) où les deux mesures sont maximisées et leurs valeurs identiques.

(b) **Les mesures combinées**

Ni le *Rappel*, ni la *Précision* permettent, isolement, de caractériser correctement un système de classification ; l'amélioration de l'un va au détriment de l'autre [Seb99]. En pratique, on cherche un bon compromis entre le *Rappel* et la *Précision*. Cela dépend généralement de l'application ; par exemple le *Rappel* est beaucoup plus important que la *Précision* lorsqu'il s'agit de trouver les courriels qui ne sont pas des *Spam* ; il est très important de trouver tous les courriels, cependant il est moins grave que certains pourriels survivent au filtrage.

Or une combinaison de ces deux mesures doit être employée. Plusieurs indicateurs ont été créés, mais le plus usuel est la mesure *FScore* (*F-Mesure*) [MRS08]. Elle est définie par :

$$FScore = ((1 + \beta^2) * Precision * Rappel)/((\beta^2 * Precision) + Rappel)$$

Mais elle est souvent utilisée sous cette forme (paramètre $\beta^2 = 1$) :

$$F1 = (2 * Precision * Rappel)/(Precision + Rappel)$$

D'autres mesures sont employées notamment :

- **Exactitude** (*Accuracy*) ou *Pertinence* : $(VP_i + VN_i)/(VP_i + FP_i + FN_i + VN_i)$
- **Erreur** (*Error* ou *ErrorRate*) : $(FP_i + FN_i)/(VP_i + FP_i + FN_i + VN_i)$
- **Taux de chute**(*Fallout*) $= FP_i/(FP_i + VN_i)$
- **Silence** = 1 - Rappel $= FN_i/(VP_i + FN_i)$
- **Spécificité** $= FP_i/(FP_i + VN_i)$
- **Bruit** $= FP_i/(VP_i + FP_i)$

2. **Techniques de validation**

   Action par laquelle on tente d'estimer les performances futures d'un modèle qui vient d'être construit sur un jeu de données. Bien que la qualité réelle d'un modèle ne peut pas être estimée au vu de ses résultats sur les données qui ont servi à le construire, mais dans beaucoup de situation on ne dispose que d'un jeu de données restreint sur lequel le modèle est construit. Sur ces mêmes données le modèle est aussi validé à travers plusieurs techniques de simulation des conditions réelles d'utilisation.

   Le principe consiste à subdiviser les données disponibles en (K) sous-ensemble ou échantillon disjoints, puis on sélectionne un des (k) échantillons comme l'ensemble de validation et le reste comme l'ensemble d'apprentissage. On répète l'opération en sélectionnant un autre échantillon de validation parmi les (k-1) échantillons qui n'ont pas encore été utilisé pour la validation du modèle. L'opération est répétée (k) fois pour qu'en fin de compte chaque sous-échantillon ait été utilisé exactement une fois comme ensemble de validation. La performance moyenne est enfin calculée pour estimer l'erreur de prédiction.

   Parmi les techniques de validation on retrouve la *Validation Croisée* qui est souvent employée pour déterminer les meilleurs paramètres d'un modèle. Le *Leave One Out* est un cas particulier de *Validation Croisée* dans lequel chacun des ensembles d'apprentissage est constitué de l'ensemble complet des données dont on a retiré un seul élément.

3. **Collection de test**

   La communauté de la T.C a hérité de la communauté de la R.I la tradition ancienne[8] d'avoir recours à des démarche expérimentaux sur un ensemble de jeux de données ou des collections de test mises en œuvre pour fournir des éléments de comparaison entre les classifieurs. Cela en employant les mêmes mesures (*Rappel*, *Précision* ... etc.) sur le même corpus et portant sur les mêmes proportions des données d'apprentissage et de validation. Parmi ces collections on peut cité :

   - **La collection de *Reuters*** En 1987, la première collection concernant la T.C a vu le jour. Il s'agit de la collection *Reuters* [Lew92]. L'agence éponyme a proposé un corpus

8. Á titre d'exemple la collection de test *TREC* a été crée dès les années 60 dans le cadre du projet *Grandfield* par *Cleverdon*

initial, nommé *Reuters-22173*, de dépêches en langue anglaise. Les documents ont été étiquetés manuellement selon le sujet. Ce corpus a été préparé essentiellement pour des tâches de catégorisation. Il est constitué de plusieurs sous-ensembles notamment "*ModApte*" qui est l'ensemble le plus utilisé [Jal03]. Depuis, plusieurs versions (versions de Yang, Apte et PARC) ont été diffusées . Ces versions se différencient entre elles par le nombre de documents et de catégories.

Ce corpus est souvent utilisé ; comme dans [DPHS98] et [Joa98] pour évaluer les performances des machines à vecteurs supports.

– **Le corpus *Ohsumed*** Ce corpus est une collection de documents qui constitue un sous ensemble de la base bibliographique médicale "*MEDLINE*". La collection contient 348.566 références provenant de 270 journaux médicaux couvrant les années de 1987 à 1991. Chaque document appartient à une ou plusieurs des 23 catégories de *MeSH* (*Medical Subject Headings*) correspondant aux maladies cardio-vasculaire.

1.3 Conclusion

Dans la première partie de ce chapitre nous avons brièvement rappelé l'apprentissage numérique et ses différents types. Nous avons notamment vu les principaux critères de minimisation du risque utilisé comme principe d'induction pour l'extraction des relations entre les données. Parmi ces principes nous avons exposé le principe de la minimisation du risque structurel introduit dans la théorie d'apprentissage développée par Vapnik et Chervenonkis. Ce principe fait appel à la VC-dimension pour mesurer la complexité d'un modèle et de son pouvoir de généralisation. Les Séparateurs à Vaste Marge (*SVM*), méthode de classification, trouvent leurs large succès dans l'utilisation de la MRS qui leurs assure un bon niveau de généralisation et les rend peu sensible à la haute dimension.

La première partie a été consacrée à la classification de textes. Un domaine au carrefour d'autres disciplines et un "*test bed*" pour plusieurs méthodes d'apprentissage automatique [Seb02]. Après une présentation de cette discipline et de ses différentes applications, nous avons formalisé la T.C dans l'approche de l'apprentissage supervisé.
Dans cette approche, la T.C est la tâche d'affectation d'une ou plusieurs étiquettes à un document en fonction de son contenu textuel. Pour réaliser cette tâche le texte candidat doit être pré-traité dans une première phase pour être soumis par la suite, à un algorithme de catégorisation pour détecter le ou les thèmes abordés dans ce texte.

Les problématiques liées aux données textuelles, à l'instar de la T.C, nécessitent une attention particulière quant au choix de la représentation ou du modèle à adopter pour ce type de données [AES10] afin de permettre à l'algorithme d'apprentissage de tirer le maximum d'information sur les données et leurs natures fouinées dans leurs structures. Le modèle de l'espace vectoriel est souvent employé pour ce genre de problématique ; toutefois d'autres modèles ont été adoptés, dans le même contexte, tel que les modèles de Markov cachés et les transducteurs. Le prétraitement consiste à faire face à la haute dimensionnalité qui est le problème majeur dans cette tâche parce que les attributs redondants et non pertinents dégradent souvent les performances d'un système de classification à la fois en termes de vitesse et de précision de classification. La réduction de dimension que ce soit par extraction ou sélection, présente une solution à ce problème. Encore la normalisation linguistique peut être perçue comme une manière de réduction. Dans ce dernier cas, la démarche consiste à ramener plusieurs forme d'un même mot à une seule ; soit par dé-suffixation ou lemmatisation. Une fois le nombre de termes est réduit, on cherche à leurs attribuer une valeur pour quantifier la représentativité ou le pouvoir discriminant d'un terme vis-à-vis un document et/ou vis-à-vis toute la collection.

La deuxième phase dans le processus de construction d'un système de catégorisation est le choix de la méthode d'apprentissage et de ses paramètres. Un panorama de méthodes existe (*Rocchio*, *K-ppv*, *N.B*, *SVM* ...) qui se distinguent par leurs fondements théoriques et le principe inductif suivi. La plupart de ces méthodes ont été longuement et largement expérimentées

dans le cadre de la T.C.

La phase finale, est consacrée quant à elle, à l'évaluation du système de classification construit. Les métriques usuelles (*Rappel*, *Précision*, *F1*) employées pour cette fin ont été abordées dans la dernière section, ainsi que de quelques exemples de corpus standards mis en œuvre pour des raisons de comparaison entre différents systèmes de classification ou entre plusieurs paramètres d'une même méthode.

Chapitre 2

Classification de textes en langue arabe : état d'art

Dans le chapitre suivant nous allons abordé les différents travaux réalisés pour la classification des textes en langue arabe en se basant sur les notions développées dans le présent chapitre et s'inspirant des différents résultats atteints dans le cadre général de la T.C.

Dans ce chapitre, nous nous intéressons aux travaux réalisés pour la classification de textes en langue arabe, une langue caractérisée par sa richesse et sa morphologie très complexe. Nous essayerons d'expliquer en premier lieu, cette complexité dans un contexte général lié au cadre de notre étude. Ainsi nous nous limitons aux aspects morphologiques ; l'aspect syntaxique est au delà de notre intérêt. Ensuite nous allons présenter, pour les textes en langue arabe, les différentes phases de la classification antérieurement évoquée dans le chapitre précédent ; à savoir la représentation et le prétraitement des textes, les méthodes d'apprentissage et l'évaluation des systèmes de classification, tout en essayant de souligner la spécificité de cette langue qui fait que certains résultats constatés pour d'autres langues ne s'appliquent pas à cette dernière.

2.1 Présentation de la langue arabe

La langue arabe est l'une des six langues officielles des nations unis, c'est la première langue de plus de 422 millions de personnes et environ 250 millions de personnes comme deuxième langue [1]. Elle est de la famille des langues sémitiques qui descend du proto-sémitique. C'est une langue ancienne exclusivement parlée qui n'a pas de trace écrite [Ek05]. Elle doit son expansion territoriale à la propagation de l'islam, à la diffusion du Coran et à la puissance militaire des Arabes à partir du VII^e siècle. Du fait, cette langue s'est répandue dans toute l'Afrique du nord et en Asie mineure.

1. wikipédia : Arabic language - wikipedia, the free encyclopedia. URL : http ://ar.wikipedia.org/wiki/لغة عربية

2.1.1 Les descendants de la langue arabe

Il y a trois formes de la langue arabe :

- *L'Arabe classique :* C'est la forme littéraire que l'on trouve dans le Coran. Elle est employée pour lire ou réciter des textes religieux islamiques.
- *L'Arabe Standard Moderne (ASM) :* C'est une forme un peu différenciée de l'Arabe classique, c'est la langue écrite de tous les pays arabophones. Il s'agit de la variété retenue comme langue officielle dans tous les pays arabes, et comme langue commune entre eux. Elle est également la langue employée dans la plupart des écrits administratifs, médiatiques, scientifiques, techniques, littéraires ... etc.
- *L'Arabe dialectal :* C'est la langue parlée au quotidien dans les pays arabes, elle varie d'un pays à l'autre, voire d'une région à l'autre dans un même pays. C'est un résultat d'une interférence linguistique avec les langues locales ou voisines, à l'issue d'un processus d'arabisation ou d'une influence culturelle quelconque due principalement à la colonisation, aux mouvements migratoires, au commerce ... etc.

2.1.2 La langue arabe sur le Net

Depuis 1995; l'année de l'apparition en ligne du premier journal en Arabe[2]; l'emploi de la langue arabe sur Internet n'a cessé de croître. À la fin de 2010 la langue arabe venait en 8^{eme} place avec 65365400 d'utilisateurs représentant un taux de croissance colossal de 2501.2 % par rapport à l'an 2000. Encore, selon des statistiques récentes[3] (fin 2011), ce nombre est passé à 86.077.806 utilisateurs ce qui représente un taux de pénétration d'Internet de 23.9 % de la population arabe. Le nombre de sites a été estimé en 2005 à 1.9 million de sites en langue arabe et ce nombre est estimé à doubler chaque année [ACS05].
Ces chiffres reflètent l'importance de la communauté qui utilise la langue arabe sur le web. Fournir des outils de qualité que ce soit pour les domaines du traitement automatique de la langue naturelle, recherche d'information, filtrage d'informations ou classification hiérarchique des news, devient un enjeu essentiel pour faire face à cette croissance.

2.1.3 Propriétés morphologiques de la langue arabe

La langue arabe est une langue flexionnelle, et non analytique [Abd04] qui s'écrit de droite à gauche. Son alphabet est constitué de 28 caractères :

ا ب ت ث ج ح خ د ذ ر ز س ش ص ض ط ظ ع غ ف ق ك ل م ن ه و ي

En plus de la lettre EL-HAMZA (أ) qui est considérée par certains linguistes comme une lettre [Duw07]. Les lettres (ا و ي) jouent le rôle de voyelles et le reste est considérée comme des consonnes. L'alphabet peut être étendu à 90 éléments par l'écriture de signes supplémentaires, voyelles et différent formes selon leurs positions dans un mot [TAS90]. Par exemple la lettre (غ) peut être écrite de plusieurs manières (tableau 2.1) selon qu'elle est isolée, au début, au

2. Le journal asharq alawsat, www.asharqalawsat.com
3. http ://www.internetworldstats.com/stats19.htm page consultée le 23 Août 2012.

milieu ou à la fin d'un mot. Aussi, dans certaines situations la combinaison de certaines lettres comme EL-YEH (ى) avec EL-HAMZA (أ) peut donner lieu à une nouvelle forme (يء)[4].

Début	Milieu	Fin	Séparé
غيمة	مغرب	بلغ	بلاغ
(Nuage)	(Ouest)	(Arrivé)	(Communiqué)

TABLE 2.1 – Représentation du caractère (غ) dans un mot.

Un autre caractère spécifique avec un usage fréquent appelé kashida ou Tatweel (اَلمدّة) "-" permet l'allongement du trait au milieu des mots. Il est employé pour des raisons esthétiques et n'a aucun effet sur le mot. La vocalisation (التشكيل) est une manière d'ajouter des voyelles courtes ou de doubler une lettre. Elle est réalisée par le placement des signes : (Sukun , Damma, Kasra, Fatha, Shadda) aux dessous ou au dessus des lettres. La voyelle "shadda " [5] peut générer deux mots par exemple "شَدَّ" et " شَدَدَ ".
La vocalisation est généralement employée pour fournir un guide phonétique puisque la langue arabe écrite ne fournit pas suffisamment d'information concernant la prononciation ; elle est employée pour spécifier le sens voulu des mots qui ont la même graphie (désambiguïsation). Le tableau 2.2 illustre un exemple de l'emploi des accentuations voyelles du mot (البرّ)

اَلبِرُّ	اَلبَرُّ	اَلبُرُّ
Bienfaisance	Terre	Blé

TABLE 2.2 – Exemple de l'emploi des accentuations voyelles.

Cependant l'écriture explicite de " shadda " et de la majorité des accentuations voyelles en arabe est de moins en moins fréquente surtout dans les domaines scientifiques et techniques [SRZ07] .

2.1.4 Catégories des mots arabes

Selon les parties du discours, trois grandes catégories peuvent être recensées [Dah87] :

1. **Particules ou lettre (الحروف) :** fixes et dénombrables. Cette catégorie comprend les lettres de l'alphabet connues par les lettres de construction (حروف المباني) qui s'unissent pour former des mots, et les lettres (= particules) de signification حروف المعاني, dont le sens n'est complet que si elles sont employées avec un nom ou un verbe. Ces dernières représentent les connections entre les mots à l'instar des prépositions (حروف الجر), des pronoms (الضمائر), adverbes (الظروف), conjonctions (حروف العطف), ...etc. On les appelle

4. Parfois ces changements de la forme peuvent être accompagnés d'un changement dans l'encodage de ces lettres.

5. L'équivalent en français de la répétition de lettre : tt, mm,....

aussi mots outils, mots grammaticaux ou mots structuraux. On recense environ quatre-vingts particules de signification [Dah87]. Ces mots sont généralement supprimés dans les phases préliminaires de la classification.

2. **Noms (الأسماء) :** fixes ou obtenus par dérivation. Cette catégorie 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 noms en arabe ont deux genres, masculin (المذكر) et féminin (المؤنث) ; trois nombres : singulier (مذكر), duel (مثنى) et pluriel (جمع) ; et trois modes grammaticaux : nominatif (مرفوع), accusative (منصوب) et génitif (مجرور).

   Un nom est à la forme nominative quand il est sujet (مبتدأ) ; accusative quand il est l'objet d'un verbe (complément d'objet) (مفعول به) ; et génitif quand il est dépendant d'une préposition (اسم مجرور). La forme d'un nom peut changer selon son mode (nominatif, accusatif ou génitif) ; par exemple, le pluriel du mot (مسافر) (voyageur) peut avoir la forme (مسافرون) (voyageurs) dans le mode nominatif et la forme (مسافرين) dans le cas accusatif ou génitif. Le stemming assoupli peut traiter ces cas.
   Selon des critères morphologiques les noms sont subdivisés en deux sous classes : noms invariables (اسماء غير متصرفة) et noms variables (اسماء متصرفة). Les noms variable ont des formes différentes pour le : singulier, duel, pluriel, masculin, féminin, diminutive et relative. Autrement dit, un nom est dit variable s'il possède des formes fléchies. Certains d'entre eux sont des noms fixes ou non dérivés (ne se référant pas à une racine verbale) (اسم جامد) et d'autres dérivés (اسم مشتق). Il est à noter qu'il n'y a pas de capitalisation de lettres dans l'Arabe ; les noms ne sont pas détecter par le principe de capitalisation de la première lettre comme dans les langues indo-européennes.

3. **Verbes (الأفعال) :** cette catégorie regroupe toutes les unités lexicales référant à un état ou à une action passée, présente ou futur. Ici, le verbe joue un rôle flexionnel plus important que dans d'autres langues [SRZ07] . Le verbe s'accorde en personne, genre et nombre avec le sujet : (بحثنا, نبحث, بحثت, بحثتما, بحثتم, بحثتن ...)

2.1.5 Phénomène d'agglutination

Le phénomène d'agglutination est une caractéristique de la langue arabe. Ce phénomène se manifeste lorsque des prépositions précédant des mots comme " ك " (qui signifie comme) se collent avec ces derniers, et/ou certains pronoms comme "ها" (sa) sont rattachés aux mots vers la fin. Le terme " كبحثها " (comme sa recherche) est constitué de la préposition " ك " au début, le lemme " بحث" (rechercher) et le pronom " ها " à la fin. Dans ce cas, La séparation des mots (*tokenization*) ou l'extraction d'un lemme à partir d'un mot n'est pas toujours facile à réaliser sans risque d'erreur parce que une lettre comme "ك" peut s'employer comme préposition ou faire partie des lettres constituant un lemme comme dans le mot "كتاب" (livre). En réalité, un mot arabe peut avoir une forme plus compliquée comme " ليساعدونهم ", si tous les affixes s'attachent à sa forme standard (tableau 2.3).

Généralement, les préfixes et les suffixes dans les langues indo-européennes n'ajoutent au-

cune entité (comparable, incomparable) ; alors que dans la langue arabe, l'affixe peut s'ajouter à une entité, un nom ou un verbe (tableau 2.4) et modifier le sens : par exemple le préfixe peut être une préposition et le suffixe un pronom.
Les affixes sont classés en quatre catégories selon leur rôle syntaxique [SRZ07] :

- les antéfixes (sont généralement des prépositions),
- les préfixes (représentés par une seule lettre et indiquent la personne de la conjugaison des verbes au présent),
- les suffixes (sont les terminaisons de conjugaison des verbes et les signes du pluriel et du féminin pour les noms),
- les post-fixes (représentent des pronoms).

post-fixes	suffixe	Lemme	Préfixes	Antéfixe
هم	ون	ساعد	ي	ل
Pronom signifiant (eux)	Terminaison de conjugaison	Le verbe "aider"	Une lettre signifiant le temps et la personne de conjugaison	Préposition (pour que)

TABLE 2.3 – Forme agglutinée d'un mot arabe signifiant "pour qu'ils les aident" [SRZ07]

.

Sens	Suffixe	Infixe	Préfixe	Mot
Scientifique	ية			علمية
(Elle) Nous a appris	تنا			علمتنا
Scientifique	ه			علمية
Sa science	اء			علمه
Savants				علماء
Enseignement		ي	ت	تعليم
Sciences		و		علوم
Renseignement	ية	ا	است	استعلامية

TABLE 2.4 – Le mot "علم" et ses différents sens selon les affixes raccordés [Mot10].

2.1.6 Dérivation

La plupart des mots arabes (à l'exception de quelques noms propres et mots empruntés à d'autres langues) sont dérivés d'une racine. Une racine se compose habituellement de trois consonnes, rarement quatre ou cinq consonnes. La majorité des noms et des verbes sont dérivés d'un nombre réduit (environ 10.000) de racines verbales [Dar02]. La figure 2.1 énumère quelques mots dérivés de la racine tri-lettres" درس ".

A partir des racines, on peut générer des lemmes ou des dérivés aussi bien nominaux que verbaux par l'application des règles morphologiques appelées modèles morphologiques ou patrons (الأوزان). Ainsi on peut produire un stem et en lui attachant des affixes on peut produire

un mot [MMC+01]. Pour cette raison, un algorithme de stemming pour la langue arabe peut être basé sur une racine ou sur un stem.

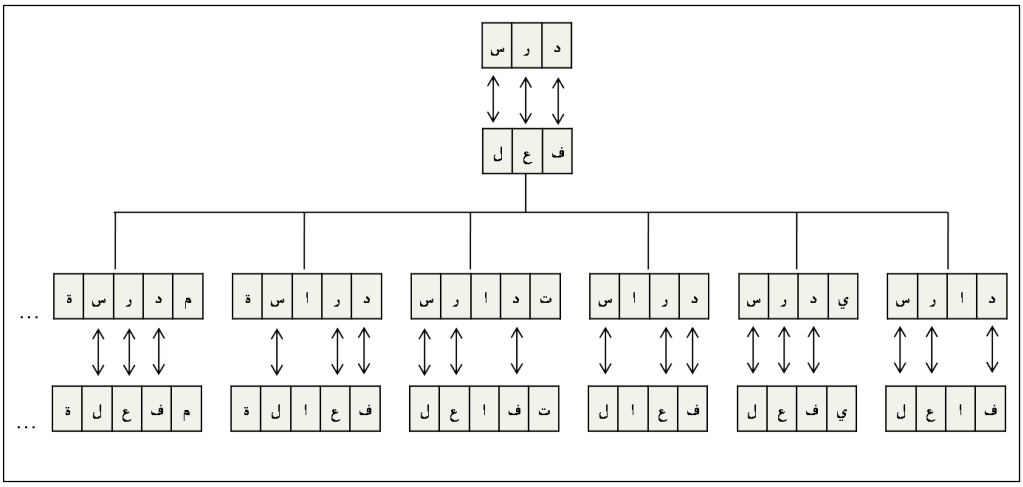

FIGURE 2.1 – Exemples de mots dérivés à partir de la racine " درس " .

Le tableau 2.5 montre quelques patrons pour des racines tri-lettres.

Longueur de patron	Exemples de patrons
4 lettres	مفعل / فعلة / فعال / فاعل
5 lettres	فعالة / افتعل / مفعول / تفاعل فواعل / مفاعل / تفعيل / فعولة فعاءل / فواعل / يفتعل / فاعلة
6 lettres	افتعال / مفاعلة / استفعل افعوعل / انفعال

TABLE 2.5 – Exemple accord verbes.

Parfois des lettres représentant les patrons de dérivation sont insérées à l'intérieur d'un mot (comme la lettre (ا) dans le patron (فاعل)) pour produire certaines formes (figure 2.1). Dans d'autres cas, au milieu du mot, certaines lettres sont supprimées ou substituées par d'autres lettres pour engendrer d'autres formes. C'est le cas des verbes irréguliers (الأفعال المعتلة) ou du pluriel irrégulier des noms (جمع التكسير) (le pluriel brisé). Par exemple, la forme plurielle du mot " قافلة" (caravane) est "قوافل" (caravanes).

2.1.7 Les défis de la morphologie de la langue arabe

L'ASM est la langue de la communication écrite, celle des médias officiels et de tous types de communication non spontanée dans les pays arabes. Elle relève certains défis, qui lui sont spécifiques, dans le cadre de la classification de textes, pour des raisons qu'on peut résumer en quelques points :

- Niveau élevé de sensibilité au contexte sur plusieurs dimensions notamment au niveau d'écriture ; variation orthographique et phénomène d'agglutination rendent la tâche de la segmentation des textes en arabes une tâche difficile pour la T.C.
- Le caractère d'infliction et dérivation qui caractérise cette langue, où des noms, des adjectifs, et des verbes peuvent être dérivés en nombre important d'une seule racine ; on peut générer jusqu'à 30 lemmes à partir d'une seule racine [Kad04]. Encore la majorité des dérivations sont réalisées sur des racines tri-lettres augmentant ainsi le degré de l'ambiguïté de cette langue et compliquant la tâche de l'extraction de la racine.
- L'ambiguïté où la langue arabe est considérée parmi les plus ambiguës des langues pratiquées [Abd04]. Cette ambiguïté est due ; en plus du niveau dérivationnels élevé ; à l'omission naturelle des accentuations voyelles dans l'Arabe écrit ou à leurs élimination par les outils de normalisation orthographique dans les phases de prétraitement ; et comme résultat de la non voyellation, les mots tendent à avoir un haut niveau d'ambiguïté : un mot peut avoir plusieurs significations, par exemple le mot non voyellé "جزر" qui a au moins deux significations : "جُزُر" (îles) et "جَزَر" (carottes).
  Cette ambiguïté peut pénaliser les performances de la T.C en langue arabe. Pour des mots s'écrivant par les même lettres et ayant des sens différents, la précision sera négativement affectée dans le sens où les documents contenant ces mots auront une chance d'être classés ensemble (la similarité augmente).
- Les règles régissant cette langue ont beaucoup d'exception. Le pluriels brisé, qui caractérise ces exceptions, est très courant : cette forme de pluriels n'obéit pas aux règles morphologique normales (ils ne sont pas reliés par de simples affixes). Elle échappe donc à la plus part des algorithmes de stemming. Aussi lors de la dérivation certaines lettres de la racine sont supprimées ou modifiées par d'autres lettres pour engendrer d'autres formes (le cas des verbes irréguliers). Ces changements des lettres à l'intérieur des mots sont observés dans plusieurs langues ; mais c'est un cas très fréquent en langue arabe [LBC02] compliquant encore la tâche de la normalisation linguistique.

En plus de ces difficultés intrinsèques à la langue arabe, il faut rajouter le manque constaté dans les outils de l'extraction des racines ou leurs faibles précisions (cf. page 43) et notamment le manque des corpus de test standards et libres (voir plus loin dans la section corpus pour la langue arabe). D'autres facteurs peuvent aussi être cités :

- L'emploi des synonymes qui sont très répandus, car la variété dans l'expression est souvent appréciée en tant qu'élément d'un bon modèle d'écriture par les rédacteurs ou les auteurs (littéraires, politiques ou scientifiques) arabes [XFW02].
- La concaténation erronée de quelques *mots outils* notamment (ما) et (لا) avec leurs successeurs comme par exemple (مازال , مايرام , لايزال , لاشكّ , لابدّ). Ce problème s'étend à d'autres combinaisons aléatoires et assez fréquentes de certains mots [Buc04] comme par exemple la forme (فقدتم) qui peut s'agir des verbes (فَقَدتُم), (ف / قُدتُم) ou (فَقَد / تَمَّ).

2.2 Représentation et prétraitement des textes en langue arabe pour la T.C

En plus des traitements communs à d'autres langues, à savoir la conversion du texte initial (généralement vers le codage UTF-8) et la suppression des nombres, des signes de ponctuation et des mots outils ; les textes en langue arabe nécessitent une phase de normalisation orthographique des lettres qui consiste à supprimer les voyelles courtes et au remplacement de certains caractères comme (أ ، إ ، آ) par (ا) et (ة) finale par (ه)

2.2.1 Normalisation linguistique

Les algorithmes de *stemming* ou les *stemmers* ont été développés pour plusieurs langues. L'efficacité d'un algorithme de stemming dépend de la langue. Pour les langues fortement fléchis (*highly inflected languages*) et quand les textes ne sont pas longs [LBC02] cette efficacité est nettement améliorée. Pour la langue arabe on peut distinguer , selon leurs natures ou leurs manières de procéder quant à la réduction des variantes des mots, les approches de stemming suivantes [LBC02] [SFH06] :

- Construction manuelle de dictionnaires : le stemming se base sur un dictionnaire de racines manuellement construit comme dans [AKE94] .
- Stemming assoupli ou léger (light stemming) : qui fait la suppression d'un ensemble restreint d'affixes (préfixes et suffixes) sans autant traiter les infixes.
- Extraction de la racine (*Root-Based stemmer*) : pour déterminer le lemme ou la racine d'un mot, la plus part de ces algorithmes emploient une liste de patrons qui représente les formes des mots. Ils procèdent, après une dé-suffixation, à une analyse morphologique du reste du mot pour tenter de trouver la forme convenable parmi la liste de patrons ; c'est pourquoi ces techniques sont considérées comme étant basées sur un dictionnaire (de patrons). Toutefois d'autres technique comme celle du *Shalabi et al.* [AKS03] ne se base sur aucun dictionnaire de ce genre pour extraire la racine.
- Stemming statistique : se base sur la représentation dite de "N-grammes" comme dans [SFH06].

La première technique où l'on construit manuellement le dictionnaire des racines est rarement employée pour des considérations pratiques. Pour les autres techniques nous allons les détailler en évoquant le contexte de leurs applications et cela toujours dans le cadre de la classification de textes en langue arabe.

Stemming assoupli "*light-stemming*"

Inspiré de l'algorithme de Porter, le stemming assoupli opère une légère dé-suffixation sur les mots. Pour ce faire, une liste d'affixes (à une, à deux et à trois lettres) est établie. La décision de tronquer un préfixe ou un suffixe d'un mot est faite selon de simples règles comme la longueur de mots ; par exemple, on ne peut pas tronquer un préfixe à trois lettres d'un mot

de longueur quatre.
Bien que le stemming assoupli échoue, dans certains cas (comme le cas du pluriel brisé), à regrouper correctement plusieurs formes d'un mot dans la même classe, il représente une approche légère dans sa mise en œuvre.
Plusieurs algorithmes de stemming assoupli ont été développés comme celui de *Chen et al.* [CG02], ou de *Larkey et al.* [LBC02]. Ce dernier a expérimenté [6] différentes versions de stemming assoupli (light1, light2, light3 et light8) (tableau 2.6) sans ou avec suppression des mots outils, en plus du lemmatiseur de *Khoja* (cf. paragraphe suivant).
Le stemmer lights8-s (light8 avec suppression des mots outils) est un autre stemmer assoupli basé sur ce dernier avec quelques ajouts dans la liste des affixes sont largement repris dans d'autres travaux (comme dans [Mes07] par exemple).

-	Suppression préfixe	Suppression suffixe
Light1	ال ، ول ، بل ، كل، فل	-
Light2	ال ، ول ، بل ، كل، فل ، و	-
Light3	"	ه ، ة
Light8	"	ها ، ان ، ات ، ون ، ين ، يه ، ه ، ت ، ي

TABLE 2.6 – Différents stemming assoupli expérimentés par *Larkey et al.* [LBC02].

Extraction de la racine

Les algorithmes de l'extraction des racines (*Root-Based stemmer*) ou les lemmatiseurs réduisent plusieurs formes à leur lemme. Ils constituent un procédé pour réduire considérablement l'espace de représentation.
Une première approche intuitive, pour la langue arabe, consiste à utiliser les racines en tri-lettres comme descripteurs. Le lemmatiseur de *Khoja et al.* [KG99], parmi les plus cités dans la littérature, applique cette méthode. Il essaye de trouver les lemmes des mots en éliminant d'abord les préfixes et les suffixes les plus longs (en s'assurant qu'il ne supprime pas une partie de la racine). Ensuite, il procède à une analyse morphologique qui compare la forme dépourvue d'affixes avec une liste de patrons de la même longueur pour déterminer le lemme recherché [LC01] [EBR+04b]
L'extraction des lemmes n'est pas une tâche facile, spécifiquement pour la langue arabe où les règles à appliquer dans cette opération sont trop complexes. Pour contourner ce problème *Shalabi et al.* [AKS03] a développé une technique statistique de mise en œuvre simple pour extraire le lemme en tri-lettres. Cette technique consiste, pour un mot qu'on cherche à extraire le lemme, à multiplier le rang de chaque lettre constituant ce mot par le poids prédéfini de cette lettre, ensuite les trois lettres correspondantes aux minimales des produits (entre le poids et le rang des lettres) seront retenues comme le lemme du mot en question.
Le rang d'une lettre dépend de la longueur du mot et du nombre de lettres s'il est pair ou impair comme illustré dans le tableau 2.7. Quant aux poids attribués aux 28 lettres de l'alphabet

6. Sur le corpus TREC-2001 pour l'évaluation d'un système R.I

Position de la lettre (à partir de droite)	Rang (si la longueur du mot est paire)	Rang (si la longueur du mot est impaire)
1	N	N
2	N-1	N-1
3	N-2	N-2
...	...	...
[N/2]	N/2 + 1	[N/2]
[N/2]+1	N / 2 + 1 - 0.5	[N/2] + 1 -1.5
[N/2]+2	N / 2 + 2 - 0.5	[N/2] + 2-1.5
[N/2]+3	N / 2 + 3 - 0.5	[N/2] + 3-1.5
...	...	...
[N/2]	N - 0.5	N - 1.5

TABLE 2.7 – Calcul du Rang des lettres [AKS03].

arabe, ils sont figés : par exemple le poids des lettres (ا ، ة) est 5 ; pour (م ، ن) le poids est 2,...). Le tableau 2.8 donne un exemple d'extraction de la racine du mot "المحافظة".

Mot	المحافظة							
lettre	ة	ظ	ف	ا	ح	م	ل	ا
Poids	5	0	0	5	0	2	1	5
Rang	7.5	6.5	5.5	4.5	5	6	7	8
Produit	37.5	0	0	22.5	0	12	7	40
Racine	حفظ							

TABLE 2.8 – Exemple d'extraction d'une racine tri-lettres [AKS03].

Selon les auteurs [AKS03], la technique est efficace à 90% puisque la plupart des racines de la langue arabe sont en tri-lettres. En plus la plus part des mots utilisés dans les textes en arabe sont tri-lettres.
Cette technique a été expérimentée par *Duwairi* [Duw07] dans un système de classification.

Un autre lemmatiseur a été introduit par *El-kourdi et al.* [EBR+04b] relevant le défi des noms en pluriel irrégulier, et des verbes irréguliers.Selon les auteurs, le lemmatiseur extrait correctement la racine à 97% d'efficacité pour des documents Web. Ce lemmatiseur a été utilisé par [EBR04a] avec l'algorithme " *Naïve Bayes* " comme méthode d'apprentissage sur une collection de 1500 documents Web et 5 catégories. L'expérience avait, entre autre, comme objectif l'évaluation de l'effet du stemming sur les performances d'un système de classification de textes en langue arabe.

Toujours dans cette optique, et pour contourner les difficultés algorithmiques quant au développement d'un racineur efficace, une approche différente invoquant une représentation basée sur les transducteurs a été adopté par *Nehar et al.* [NZCG12]. La figure 2.2 montre un exemple d'un transducteur représentant les patrons des verbes en langue arabe.

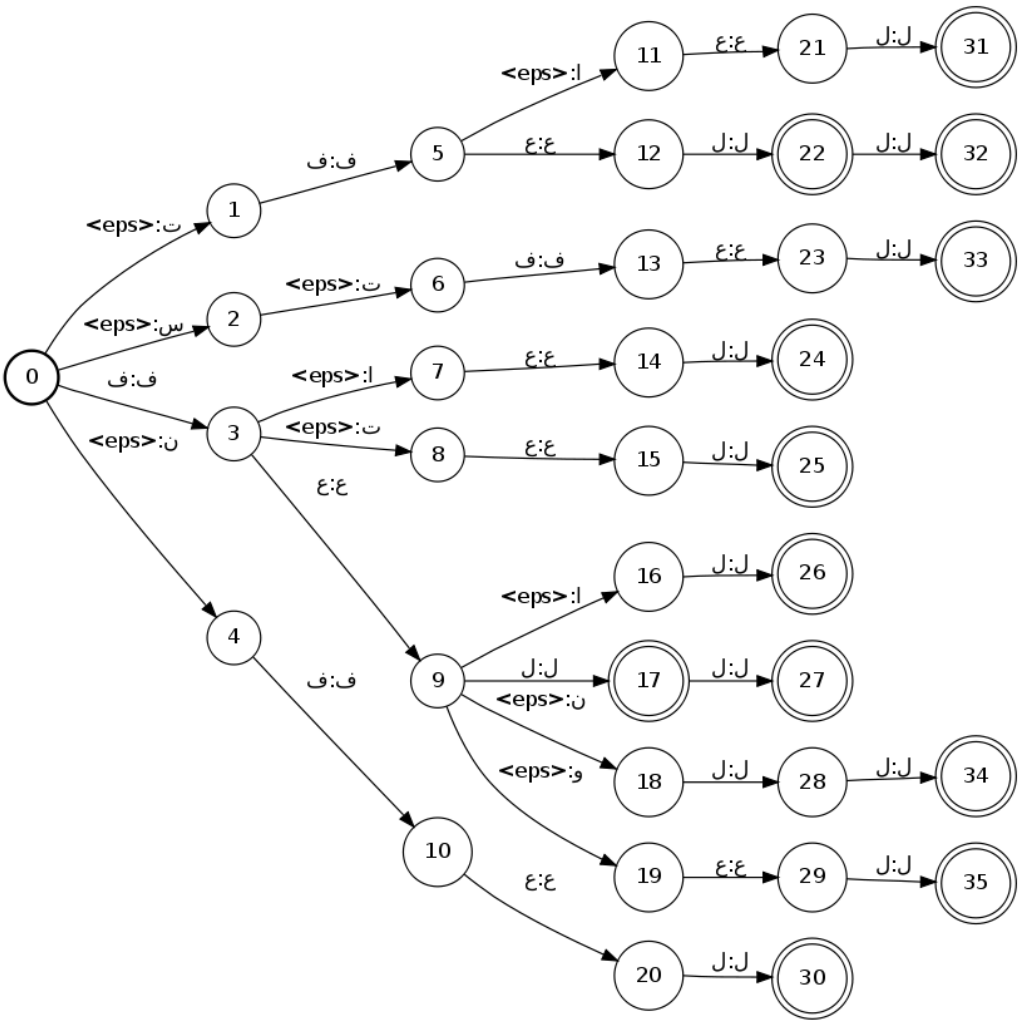

FIGURE 2.2 – Exemple d'un transducteur représentant les patrons des verbes en langue arabe.

Ainsi pour extraire la racine du verbe "تدارس" par exemple ; le verbe va être soumis à ce transducteur qui va retourner "درس" comme séquence de sortie (le chemin le plus haut de l'exemple précédent).

Loin du contexte du TAL où l'exactitude est vitale, la plus part des algorithmes de lemmatisation sont désignés pour résoudre des problématiques liées aux textes (recherche d'information, classification de textes, ...). Même dans ce contexte, " *le taux d'exactitude montre que le meilleur algorithme*[7] *échoue à la réalisation de plus de 75% d'exactitude* " [SA08]
En plus du problème de l'exactitude évoqué, l'emploi des lemmes comme descripteurs peut conduire à la perte du sens des mots originaux. En effet les lemmes sont trop généraux et très abstraits et ne sont pas spécifiques au sens que peuvent représenter les mots originaux. Par exemple si on représente le mot (مذهب) (direction mais employé pour désigner une école de pensée) par son lemme (ذهب) (le verbe aller) ; le sens des mots originaux est perdu.

7. C'est le lemmatiseur de *Khoja et al.* dans le contexte de l'étude évoquée.

Les N-grammes

La représentation en N-grammes offre une alternative intéressante. En effet cette technique se base sur l'idée que deux mots (ou deux textes) se ressemblent le plus s'ils ont en commun le plus de séquences de lettres. Elle consiste à segmenter un mot en un ensemble de séquence de N caractères consécutifs comme par exemple le tri-grammes du mot " مستشرقون " qui est l'ensemble : (مست , ستش , تشر , شرق , رقو , قون) ensuite évaluer la similarité entre deux mots à travers le nombre des N-grammes en commun. Pour les deux mots "سياسيا" et "سياسة" la similarité basée sur les 2-grammes peut être calculée par [SFH06] :

$$Similarite = \frac{2 * c}{A + B} = \frac{2 * 3}{3 + 4}$$

où c est le nombre de 2-grammes communs entre les deux mots (ici c = card { "سي" , "يا", " اس " }). A et B représentent le nombre des 2-grammes uniques pour respectivement le premier et le deuxième mot (A = card {"سي", "يا", اس} et b = card { "سي", "يا", اس , سة }).

L'intérêt d'une telle technique est qu'elle est indépendante de la langue et donne généralement de bonnes performances notamment pour le cas des textes avec des erreurs typographiques [Khr06].

Parmi les études qui exploitent cette technique est celle de *Sawaf et al.* [SZN01] qui a utilisé la métrique d'entropie maximale comme mesure de similarité pour classer des articles de journaux d'une taille de 33k et d'environ 7 M de mots.
La même technique (N-grammes) a été reprise par *Khreisat* dans [Khr06] en utilisant un algorithme basé sur la distance (*distance-based algorithm*) dans une étude comparative entre deux mesures de distance : " *Dice* " et "*Manhattan*".

2.2.2 Pondération et réduction de dimension

Parmi les schémas de pondération les plus utilisés dans la littérature de la T.C (*TF*, *Booléen*, *TF-IDF* , *TF-IDF normalisée*, ...). C'est *TF-IDF normalisée*, selon *Gharib et al.* [GHF09], qui est le schéma de pondération le plus convenable pour la T.C des textes en langue arabe. *Syiam et al.* [SFH06] a confirmé ce résultat avec le classifieur *Rocchio*.

Concernant la réduction de dimension rappelons que c'est une technique employée pour limiter le nombre de termes utilisés pour représenter les documents aux termes les plus représentatifs (voir Sec. 4). Elle peut être réaliser par la sélection ou l'extraction d'attributs.

1. **Sélection d'attributs**

   Concernant la réduction de dimension, *Thabtah et al.* [TEZH09] a conclu que la sélection d'attributs améliore les performances de la classification en conséquence d'élimination des termes rares qui créent des confusions entre les catégories. Dans leur étude *Thabtah et al.* a atteint le maximum de performance pour 800 termes (en *Précision*). Pour le *Rappel* le maximum a été atteint pour 50 termes.

   Toujours sur l'impact de la sélection d'attributs, une recherche plus approfondie a été menée dans [EBR04a] pour justifier les erreurs de classification. La démarche consistait à construite la matrice de confusion entre les catégories (Tableau 2.9) :

Catégorie	santé	économie	culture	science	sport
santé	–	13%	4%	3.7%	21.3%
économie	4.6%	–	5.3%	4.6%	12%
culture	2.3%	10%	–	0.7%	30%
science	13.3%	5.3%	2.3%	–	20%
sport	2.0%	1.3%	3.6%	1.3%	–

TABLE 2.9 – Exemple d'une matrice de confusion [EBR04a].

Les classes avec des erreurs de classification contiennent un nombre important de termes qui sont aussi représentatifs pour d'autres catégories (ces catégories contiennent beaucoup de termes qui ont une fréquence élevée dans d'autres catégories) comme illustré par l'étude (réalisée sans sélection d'attributs) qui montre que 30% d'erreurs de classification des documents de la classe " *Culture* " sont mal classés dans la classe " *Sport* ". Ces derniers (les documents mal classés) ont des termes qui sont fréquents dans la classe " *Sport* " notamment les termes : " جاءيزة " (*prime*), "بطل" (*champion*) et "تسجيل" (*marquer*). La sélection d'attributs présente une solution pour réduire l'influence de ces termes dans le processus de prédiction du classifieur. *Al-Harbi08 et al.* [AHAAT+08] s'est contenté de 30 termes sélectionnés par la métrique χ^2 selon leurs importances pour représenter chaque catégorie. L'objet de l'étude était de réaliser une comparaison entre les arbres de décision (la version $C5.0$) et les SVM sur 7 corpus différents.

Les performances d'un classifieur ne dépendent pas uniquement du nombre de termes retenus pour représenter un document ou une catégorie, mais aussi de la métrique employée. *Syiam et al.* [SFH06] a réalisé une étude comparative entre 6 méthodes de sélection d'attributs (DF : Document Frequency, IG : Information Gain, χ^2 , OR : Odds Ratio, $NGLCoefficient$ et GSS). Utilisées séparément, ces différentes métriques ont donné des résultats proches. L'hybridation entre le DF ($DF(< 2)$) et $IG : DF$ pour éliminer les termes rares et IG pour sélectionner les termes les plus informatifs réalise les meilleures performances pour un système de classification.

Cette hybridation est aussi motivée pour résoudre le problème des documents vides. Ce problème peut se manifester quand la sélection globale d'attributs est employée et la fonction utilisée sélectionne les termes rares. Dans ce cas il se peut que certain document

peuvent ne contenir aucun élément de la liste globale des termes.

2. **Extraction d'attributs**
   Pour la réduction de dimension basée sur l'extraction d'attributs *El-khoribi et al.* [EKI06], après un stemming assoupli, a procédé au regroupement des mots similaires (qui représentent le même concept) dans le même segment (*cluster*) par l'algorithme $k-means$. Le modèle de Markov Caché (*HMMs Hidden Markov Models*) a été employé pour réaliser un système de classification de textes où les données (*Saudi Fiqhi Fatwa encyclopedia*) étaient implémentés sous forme d'un système questions/réponses. Selon *El-khoribi et al.* [EKI06], l'extraction d'attributs améliore les performances d'un système de classification puisque elle élimine les termes redondants.

2.3 Les classifieurs pour les textes en langue arabe

En plus des *SVM*, Différents algorithmes ont été adoptés pour les besoins de la construction d'un système de classification ou pour des raisons de comparaison.

Nous allons d'abord, présenter les travaux réalisés sur la base des méthodes classiques (*N.B*, *Rocchio*, *K-ppv*, *algorithmes basés sur les mesures de distance*) ; ensuite nous présenterons les travaux s'inscrivants dans le cadre de l'utilisation des *SVM* comme méthode de classification.

2.3.1 Les classifieurs classiques

l'étude de *Sawaf et al.* réalisée en 2001 [SZN01] compte parmi les premiers travaux dans le domaine de la T.C pour les textes en langue arabe. *Sawaf* a adopté les 3-grammes comme méthode de représentation de textes. Pour la classification il a utilisé la métrique d'entropie maximale comme mesure de similarité pour classer des articles de journaux d'une taille de 33k et d'environ 7M de mots. Les performances en termes de *F1* obtenues n'étaient pas satisfaisantes sur les deux expériences portées successivement sur 34 et 10 catégories prédéfinies. Mais ces performances ont été considérées comme prometteuses pour une approche d'une mise en œuvre simple.

Concernant les algorithmes basés sur les mesures de distance, *Khreisat* [Khr06] a réalisé une étude comparative entre deux mesures de similarité : " *Dice* " et " *Manhattan* " où l'expérimentation à montré que cette dernière est moins performante que la distance " *Dice* ". D'un autre coté l'étude confirme la spécificité de la langue arabe, puisque la distance " *Manhattan* " qui affiche de très bonnes performances pour la langue anglaise, ne dépasse pas (en moyenne) 56% en terme de *Rappel* et 66% de *Précision*. Malheureusement cette étude a été menée sur un corpus limité à 4 catégories, et d'une taille variant de 1 Ko à 4 Ko pour les catégories " Météo " et " Technologie " et de 15 ko à 18 ko pour les catégories " Sport " et " Economie ".

Duwairi [Duw07] a réalisé une étude comparative entre un algorithme basé sur la distance " *Dice* " et les méthodes classiques de la classification (*N.B*, *K-ppv*). La démarche adoptée

consiste à construire un vecteur d'attributs pour chaque catégorie, ensuite comparer par la distance " *Dice* " chaque texte à classifier avec tous les vecteurs représentant les catégories. La catégorie la plus similaire dans le sens de cette distance sera choisie comme catégorie de destination pour le texte à classifier. Les résultats mesurés en *Précision*, *Rappel*, *Taux de chute*, et *Erreur* montrent que le *N.B* et le *K-ppv* sont plus performants que l'algorithme basé sur la distance " *Dice* ". Notant que dans cette étude l'espace de représentation n'était pas normalisé : le vecteur des attributs définit pour le *N.B* a pris comme poids la pondération *TF-IDF*, et aucune pondération n'a été employée pour les autres.

La méthode de *N.B* a été retenue par *El-kourdi at al.* [EBR04a] pour analyser l'effet sur les performances des paramètres suivant :

- Stemming : sans stemming, ou avec le stemmers d'*El-kourdi et al.* [EBR04a] ,
- Réduction de dimension : 50, 100, 500, 100, 2000 ou tous les termes,
- Différentes techniques de validation : *leave one out*, 2/3 Vs 1/3, 50 Vs 50, 1/3 Vs 2/3 .

L'étude a utilisé *TF-IDF* pour pondérer les termes sur une collection de test comportant 1500 documents divisés en 5 catégories. Les performances avaient une corrélation croissante avec la taille des données d'apprentissage et la dimension de l'espace de représentation (tous les termes, 2000 termes, 1000 termes ...). Mais les modestes performances réalisées, reviennent selon cette étude, à l'algorithme (*N.B*) adopté qui a réalisé des résultats similaires pour d'autres langues. Par conséquent, les *SVM* et *AdaBoost*[8] sont l'avenir de la classification des textes en langue arabe puisque ces algorithmes affichent de bonnes performances pour d'autres langues.

Kanaan et al. [Kan05] ont aussi employé le classifieur *N.B* pour classifier 600 documents distribués uniformément sur 7 catégories. L'exactitude réalisée avait une moyenne relativement faible (57.19%) avec un grand écart enregistré entre les catégories (entre 41% à 100%).

El-halees [EH08] a exploité les techniques du POS : *partie du discours* " du traitement automatique de la langue naturelle *TALN* " pour extraire uniquement les noms et les noms propres sur un corpus d'une taille de 609 Ko de données d'apprentissage constituées d'un ensemble d'articles tiré du site " Aljazeera.net ". Le processus de catégorisation s'est basé sur la métrique " *entropie maximale* " pour classifier un texte dans la catégorie la plus similaire. Les résultats ont enregistré une *F1* de 80.41% pour l'expérimentation basé sur le POS contre uniquement 71.24% pour l'expérimentation sans les techniques du TAL. " *ArabCat system* " le système ainsi nommé dans cette étude,a fait aussi l'objet d'une comparaison par rapport aux systèmes développés dans des travaux antérieurs (El-Kourdi et Sawaf) et notamment avec le système de classification "*Seraj*" de la société *Sakhr* (qui, selon cette étude, a réalisé uniquement une *F1* 57.68%). Ce dernier système bien qu'il soit parmi les premiers systèmes de catégorisation de textes en langue arabe ne fournit aucune documentation sur ses aspects théoriques liés à l'approche de classification adoptée [EH08] [EBR04a].

8. Adaptative boosting est un des algorithmes les plus utilisés en boosting qui est un domaine de l'apprentissage automatique.

Concernant *Rocchio*, il a été adopté dans quelques travaux pour des raisons de comparaison. *Rocchio* semble généralement, donner de bons résultats et avec un temps de traitement nettement inférieur par rapport aux autres classifieurs. Et cela même avec un nombre limité de descripteurs [GHF09].

2.3.2 Les SVM

Parmi les premiers travaux où les *SVM* ont été utilisés, *Mesleh* [Mes07] en 2007 a réalisé plusieurs expériences basé sur la mesure *Chi square* (χ^2) pour déterminer le meilleur nombre de termes à sélectionner dans la phase de réduction de dimension. Les meilleurs résultats ont été atteints (*F1* = 88,11%) pour une sélection d'attributs à 162 termes pour chaque catégorie. L'augmentation de ce nombre ralentit l'apprentissage sans autant améliorer les performance de la classification. Pour le stemming il conclu que le stemming assoupli n'améliore pas les performances (*F1* diminue à 87,1% avec le light-stemmers de Larkey). Il a aussi montré que les *SVM* surpassent les autres algorithmes de classification *k-ppv* et *N.B*.

Une étude plus générale réalisée par *Gharib et al.* [GHF09] ou il a proposé un système intelligent pour la classification des textes en langue arabe. L'étude tentait de comparer les quatre classifieurs conventionnels (*Rocchio*, *K-ppv*, *N.B*, *SVM*) avec différents paramètres : différentes techniques de stemming, différentes pondérations de termes et différents nombres de termes sélectionnés pour la réduction de dimension (entre 2500 à 5000 termes). L'étude n'a pas ressorti l'effet de ces différents paramètres sur les performances de ces algorithmes, à l'exception de celui du nombre de termes sélectionnés, où les résultats ont montré que le classifieur *Rocchio* est très performant et s'approche de ceux réalisés par les *SVM* lorsque le nombre de termes sélectionnés est petit, tandis que les *SVM* surpassent les autres classifieurs pour les grandes dimensions : les performances ont atteint une très grande précision (plus de 90%) pour une dimension de 4500 termes sélectionnés en fonction de *TF-IDF*. En contre partie l'étude a mis l'accent sur le temps d'apprentissage considérable concernant les *SVM* par rapport aux autres algorithmes.

Dans l'étude menée par *Alsaleem* [Als11] l'objectif était, de comparer sur la base des trois mesures d'évaluation choisies (*F1*, *Rappel* et *Précision*) les résultats obtenus par les deux approches : *SVM* et N.B sur une collection d'une taille importante qui est la collection SPN (Saudi News Papers) constituée de 5121 documents et 7 catégories. Les résultats ont montré que les *SVM* (*Précision* = 0.779, *Rappel* = 0.778, *F1* = 0.778) devancent l'algorithme de *N.B* (*Précision* = 0.741, *Rappel* = 0.740, *F1* = 0.740).

Les tableaux 2.10 et 2.11 résument les différentes techniques employées dans les différentes phases de la classification, les collections de test ainsi que les résultats et les conclusions tirés à partir des principaux travaux réalisés pour la classification de textes en langue arabe.

Début	Milieu	Fin	Séparé
(Nuage)	(Ouest)	(Arrivé)	(Communiqué)
(Nuage)	(Ouest)	(Arrivé)	(Communiqué)

TABLE 2.10 – État récapitulatif des différentes études réalisées sur la base des classifieurs classiques.

ceci pour le saut de page

Début	Milieu	Fin	Séparé
(Nuage)	(Ouest)	(Arrivé)	(Communiqué)

TABLE 2.11 – État récapitulatif des différentes études réalisées sur la base des *SVM*.

2.4 Corpus pour la langue arabe

Le problème de la classification de textes en langue arabe est principalement l'absence des corpus standards faisant l'objet d'un repère pour évaluer les différentes approches adoptées pour les différentes phases (prétraitement, réduction de dimension, méthodes d'apprentissage ...)qui rentrent dans la construction d'un système de classification. En effet, la taille et la nature des collections de test influence considérablement les résultats. *Dina et al.* [SWDH09] en 2009, ont tiré des conclusions contradictoires (que ce soit pour l'effet du stemming, ou la métrique employée pour la réduction de dimension,...) sur deux collections différentes : La première collection Alj-News (*Aljazeera News Arabic Data set*) constituée de 1500 articles uniformément distribués sur 5 catégories et la deuxième collection (Alj-Mgz) (*Al-jazirah Magazine Arabic Data set*) comporte 4470 articles dont leurs distribution sur les catégories n'est pas uniforme.

Concernant les corpus, les premières tentatives pour la construction d'un corpus standard ont été initiées dans le domaine du TAL essentiellement pour l'étude de la langue arabe (l'analyse morphologique), ensuite étendues pour l'évaluation des systèmes RI comme dans *Al-Kharashi et al.* [AKE94]. En 2001 *Goweder* a produit un corpus de 42591 articles de l'année 1998 du journal *Al-Hayat*. La collection *Zad* produite par *Darwish et al.* [DDJ+01] à partir d'articles fournis par "Al-Areeb Electronic Publishers LLC" est constituée de 4 000 documents écrits dans l'Arabe classique.

Aucune de ces collections n'a fait l'objet d'une commune utilisation ; aussi la plupart de ces collections présentaient l'inconvénient d'être d'une petite taille, spécialisée et régionale [ACS05], et c'est jusqu'à 2005, que seul deux corpus étaient communément disponibles pour les besoin de la recherche :

- **La collection de LDC** (Linguistic Data Consortium) qui fournit deux corpus " *Arabic Gigaword* " et " *Arabic NEWSWIRE* " compilés à partir de différentes sources incluant "Agence France Presse", "Al Hayat News Agency", "Al Nahar News Agency" et "Xinhua News Agency".
- **La collection Al-Hayat de ELRDA** (European Language Ressource Distribution Agency) comportant plus de 42.000 articles rassemblés depuis 1994.

Ces dernières années les efforts de construction d'un corpus se sont intensifié. En 2005 *Abdelali et al.* [ACS05] ont entamé une démarche pour construire un corpus d'une taille importante en collectant des articles publiés en 2002 de différents journaux en ligne d'une large région géographique : 11 pays Arabes (Egypte, Kuwait, Oman, Algérie, Arabie Saoudite, Liban... etc) totalisant un nombre de 102134 documents d'une taille de 1068744 ko. L'auteur a essayé de prouver la représentativité [9] et la complétude [10] du corpus en question.

En 2010 *Motaz* [SA10] a proposé un corpus constitué de trois collections (*BBC Arabic corpus*, *CNN Arabic corpus*, *Open Source Arabic Corpus (OSAC)*). Ce corpus représente l'intérêt d'être d'une taille importante et de libre utilisation. [11]. Cependant ce corpus nécessite une phase de

9. Quand la taille des données du corpus augmente la proportion des mots rares diminue
10. l'ajout des données après une certaine taille du corpus n'augmente pas leurs contribution dans ce corpus
11. Téléchargeable depuis : http ://sourceforge.net/projects/ar-text-mining/files/Arabic-Corpora
D'autres corpus libres sont publiés en ligne à l'adresse http ://aracorpus.e3rab.com/ consulté le 23/08/2012.

nettoyage assez importante.

Il existe d'autres corpus compilés par certains auteurs comme celui de *Al-Thubaity* qui fourni le corpus SPA (Saudi Press Agency) [AAA+08]. Ce corpus est d'une taille moyenne (1520 documents divisés sur 6 catégories) supérieure à la plus part des collections sur les quelles les travaux de la T.C en langue ont été réalisés.

2.5 Conclusion

Ce chapitre a été consacré aux différents travaux réalisés pour la classification des textes en langue arabe où nous avons présenté en premier lieu la langue arabe et ses défis pour tous système automatique traitant avec ses aspects morphologiques à l'instar de la normalisation linguistique qui rentre dans la phase des prétraitements nécessaires pour un système de classification de textes. La pondération et la réduction de dimension ont été évoquées par la suite.

Les résultats des différents algorithmes d'apprentissage adoptés par les travaux réalisés pour la classification de textes en arabe ont été détaillé dans leurs contextes. Le problème de l'absence des corpus standards a été passé en revue.

Toutefois, et à terme de ce chapitre nous pouvons dire que le développement d'un système d'apprentissage supervisé basé sur les *SVM* semble bien approprié pour la classification des textes en général de par la capacité des *SVM* à traiter un nombre de descripteurs très élevé, qui est le cas de la représentation des textes en *sac de mots*, tout en restant raisonnable dans le temps de traitement.

La classification des textes en langue arabe, n'a pas bénéficié d'une telle puissance théorique et empirique. Le nombre de travaux exploitant les *SVM* dans ce cadre en est un témoin. Cette pauvreté est due, peut être, à plusieurs raisons, citons par exemple l'absence d'un algorithme de stemming qui reste encore loin d'un consensus général comme c'est le cas de l'algorithme de Porter pour la langue anglaise. Ce manque vient s'ajouter aux spécificités de la langue arabe qui rendent plusieurs méthodes connues dans le domaine de la recherche d'information et de la classification de textes insatisfaisante [ACS05]. De plus les corpus sur les quels ces résultats ont été réalisés où le nombre de documents était d'une taille moyenne [GHF09] limite l'intérêt de ces expérimentations.

Néanmoins, quelques travaux s'intéressant au sujet ont vu le jour. Ces travaux réalisent de bonnes performances en utilisant les *SVM* par rapports aux méthodes conventionnelles [Mes07], [Als11] et pratiquement avec différentes approches de mise en œuvre : avec ou sans stemming, en employant la réduction de l'espace de représentation ou sans celui-là ...

Exploiter cette méthode est l'intérêt de ce mémoire où l'on cherche au minimum à adopté une approche différente.

Chapitre 3

Noyaux et méthodes à noyaux

On vise à concevoir un système de classification pour la langue arabe. ce système est basé sur les méthodes à noyau. Le choix des *SVM* comme méthode semble très bien justifier pour la construction des classifieurs non seulement par ses fondement théorique expliquant la qualité de leurs résultats empiriques mais aussi dans la possibilité du choix de la fonction noyaux appropriées pour le type de données. Dans la première section de ce chapitre nous allons décrire les *SVM*, la méthode qui a donné naissance aux méthodes à noyaux. Nous consacrerons la deuxième section aux noyaux dans le but d'explorer différents noyaux définis pour les données textuelles. la dernière section nous permettra de donner un ébauche sur la solution à concevoir pour notre système de classification.

Les méthodes à noyaux (*kernel machines*) sont devenues ces dernières années largement répandues dans la communauté de l'apprentissage. L'intérêt croissant pour ces méthodes réside dans la possibilité d'avoir, grâce à eux, le meilleur de deux mondes : utiliser des techniques simples et rigoureusement garanties (les algorithmes linéaires) et traiter des problèmes non linéaires [DMS+11]. En effet, les méthodes à noyaux permettent de trouver des fonctions de décision non linéaires, tout en s'appuyant fondamentalement sur des méthodes linéaires. Une fonction noyau correspond à un produit scalaire dans un espace de grande dimension où il n'est pas nécessaire de manipuler explicitement. Dans cet espace, les méthodes linéaires peuvent être appliquées pour y trouver des régularités non linéaires.

Les méthodes à noyaux comportent une large gamme de méthodes qui traitent différentes tâches d'apprentissage [CST10] : Support Vector Clustering pour l'apprentissage non supervisé, l'Analyse en Composantes Principales à noyau pour l'extraction d'attributs, les *SVM* pour l'apprentissage supervisé ....

Toutefois, il faut distinguer les méthodes à noyaux des fonctions noyaux. Les méthodes à

noyaux déterminent la manière dont la solution est recherchée dans la tâche d'apprentissage correspondante alors que les fonctions noyaux encapsulent la notion de proximité ou de similitudes entre les données [Hof00]. Les noyaux peuvent être définis sur des espaces d'entrée quelconques. Cette caractéristique des noyaux permet d'appliquer des algorithmes numériques (tels que les *SVM*) sur des données complexes non nécessairement numériques. En effet, beaucoup de problèmes dans lesquels la structure des données porte une information essentielle tels que les textes en langage naturel, documents XML, séquences biologiques, analyse de scènes (images), analyse des réseaux sociaux, ... pour les quels les fonctions noyaux offrent une possibilité de coder l'information structurelle, ou à défaut de comparer, ou calculer une mesure de similarité, entre deux structure [Suj07].

3.1 Les machines à vecteurs de support (SVM)

Les *SVM* ; les Machines à Vecteurs de Support ou encore Séparateurs à Vaste Marge (*Support Vector Machine*) sont les meilleures méthode connue dans la classe des méthodes à noyaux [CST10]. Cette classe est destinée à résoudre des problèmes de discrimination : décider à quelle classe appartient un échantillon, ou de régression : prédire la valeur d'une variable.

De manière simplifiée, le principe de base des *SVM* consiste à ramener le problème de la discrimination à celui, linéaire, de la recherche d'un hyperplan de marge optimale qui, lorsque c'est possible, classe ou sépare correctement les données tout en étant le plus éloigné possible de toutes les observations. Dans la suite, nous allons présenter ce principe de base.

3.1.1 Discrimination linéaire et hyperplan séparateur

Dans le cadre de l'apprentissage supervisé, l'objectif est de prédire une variable y à partir d'un échantillon statistique $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ de taille n et de loi inconnue. les données x_i sont des vecteurs de taille P ($x_i = x_i^1 \ldots x_i^p$) qui représentent l'ensemble des variables explicatives. Dans le cas d'un problème à deux classes la résolution d'un problème de discrimination (binaire) pour l'ensemble S passe par l'approximation d'une fonction de décision h qui à un vecteur d'entrée x fait correspondre une sortie $y \in \{-1, 1\}$.

$$h : X \to Y$$
$$x_i = (x_1^1 \ldots x_1^p) \longmapsto y_i = h(x_i)$$

Dans le cas où la séparation linéaire est possible, la fonction h recherchée s'exprime sous la forme linéaire $h(x) = w.x + b$. La surface de séparation est donc l'hyperplan : $w.x + b = 0$.
Les hyperplans ou les surfaces de décision $h(x) = 0$ sont appelés *hyperplans séparateurs*.

La figure [Fig. 3.1] montre un exemple linéairement séparable dans un espace à deux dimensions. Dans cet exemple on considère deux classes : la classe des exemples positifs (+) pour

$y > x$ et la classe des exemples négatifs (-) pour $y < x$. La droite d'équation $y = x$ est un séparateur linéaire.

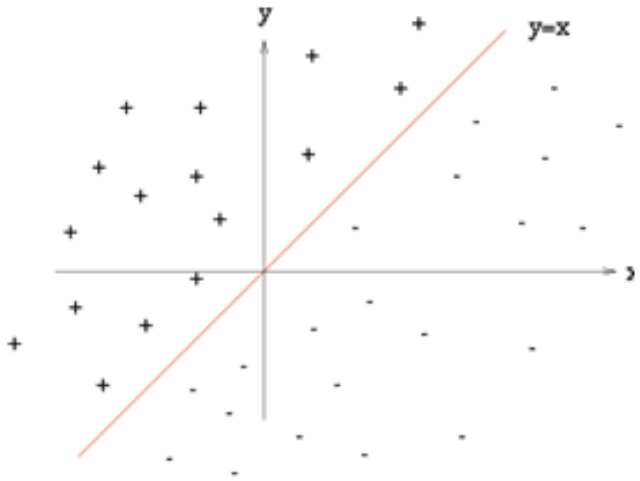

FIGURE 3.1 – Exemple d'un séparateur linéaire.

La fonction $h(x) = w.x + b$ est valide si et seulement si elle permet de classer correctement tous les points de X (*condition de séparabilité*) : $\forall i \quad y_i h(x_i) \geq 0$
on transforme l'hyperplan sous *forme canonique* c.à.d $min_i |wx_i + b| = 1$ (voir la figure [Fig. 3.2])

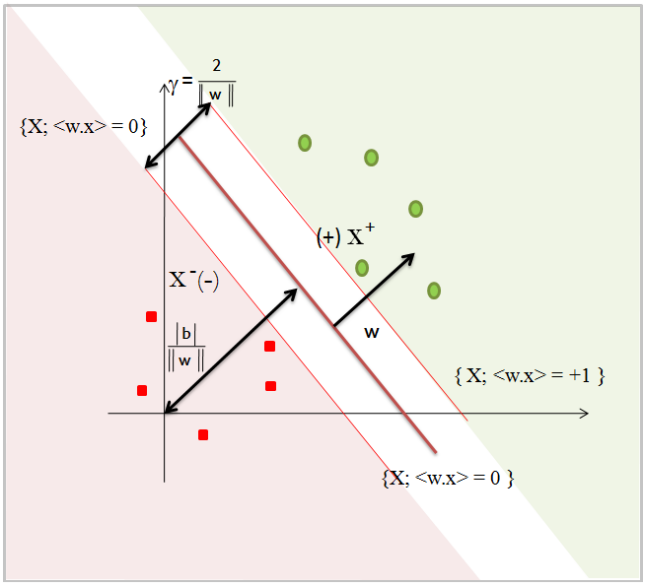

FIGURE 3.2 – Hyperplan séparateur sous forme canonique.

d'où :

$$\forall i \quad y_i h(x_i) \geq 1 \tag{3.1}$$

Le choix de l'hyperplan séparateur n'est pas évident. Il existe en effet, une infinité d'hyperplans séparateurs. La figure [Fig. 3.3] montre plusieurs séparateurs possibles dont les performances en apprentissage sont identiques (le risque empirique est le même) mais les performances

en généralisation peuvent être très différentes : $H3$ ne sépare pas les deux classes, $H1$ sépare les deux classes avec une petite marge et $H2$ est l'hyperplan séparateur optimal. La recherche du "meilleur" hyperplan peut être menée selon le principe de minimisation du risque empirique classique. Dans ce principe le meilleur hyperplan sera celui qui minimise le nombre d'exemples mal classés. Un autre critère peut être considéré dans la recherche du "meilleur" hyperplan ; le critère de confiance ou de robustesse de la fonction de décision obtenue. Un moyen de traduire la robustesse de cette fonction est de considérer la marge séparant les exemples de la classe '+' des exemples de la classes '-'. La marge est la distance entre la frontière de séparation et les échantillons les plus proches appelés *vecteurs supports* ou *exemples critiques*. Dans les *SVM*, la frontière de séparation est choisie comme celle qui maximise la marge.

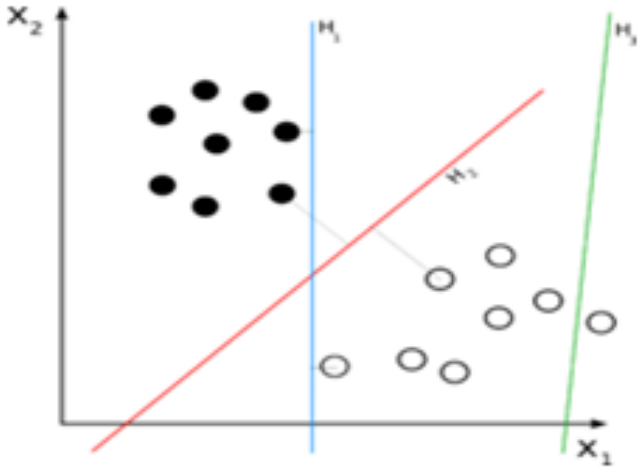

FIGURE 3.3 – Exemple de plusieurs hyperplans séparateurs.

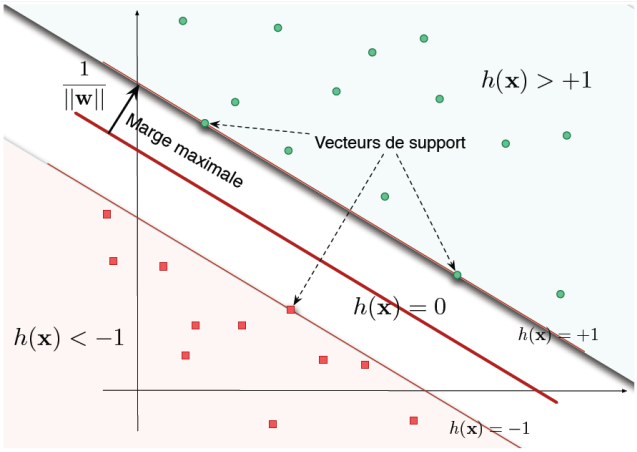

FIGURE 3.4 – Hyperplan optimal, marge maximale et vecteurs supports.

La théorie de Vapnik-Chervonenkis a montré qu'il existe un unique hyperplan optimal, défini comme l'hyperplan qui maximise la marge entre les vecteurs supports et l'hyperplan séparateur où la *capacité* des classes d'hyperplans séparateurs diminue lorsque leurs marge augmente [BGV92]. En effet, la capacité (ou la dimension de Vapnic-Chervenonkis notée D_{VC}) des *SVM* n'est plus dépendante de la dimension de l'espace mais de la valeur de la marge (voir sec. 3.1.2.2).

3.1.2 La recherche de l'hyperplan optimal

La distance d'un point x à l'hyperplan d'équation $h(x) = w.x + b$ égal à :

$$d(x) = \frac{|w.x + b|}{||w||}$$

L'hyperplan optimal est celui pour lequel la distance aux points les plus proches des deux classes (+) et (-) est maximale (Voir figure [Fig. 3.4]) . Donc cette distance vaut : $d(x) = 1/||w||$. La recherche d'une *SVM* devient un problème de minimisation de $||w||$ tous en s'assurant de la satisfaction de la condition (3.1). Dans le cas où l'ensemble d'apprentissage est linéairement séparable le minimum est unique (le domaine de minimisation est convexe) [DMS+11].

3.1.2.1 Formulation primale et duale

La recherche de l'hyperplan optimal revient alors au problème de minimisation sous contraintes suivantes :

$$\begin{cases} Maximiser_{w,b} & 1/2\|w\|^2 \\ sous & y_i(w.x_i + b) \geq 1 \quad i = 1 \dots n \end{cases} \tag{3.2}$$

cette écriture [1] est connue par la *formulation primale* du problème. Pour résoudre l'équation (3.2) on peut utiliser la fonction que l'on appelle Lagrangien. Le Lagrangien possède un unique *point selle* pour ce genre de problème. La construction du Lagrangien est donnée par :

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(w.x_i + b) - 1] \tag{3.3}$$

Le Lagrangien doit être maximisé par rapport à w et b, et minimisé par rapport à α_i (on cherche w^* , b^*, et α^* optimaux).

$$\alpha_i^* [y_i(w^*.x_i + b^*) - 1] = 0 \quad , i = 1 \dots n \tag{3.4}$$

Sous les conditions de *Kuhn-Tucker* on peut annuler les dérivées partielles du Lagrangien :

$$\frac{\partial L}{\partial w}(w, b, \alpha) = 0 \Leftrightarrow w = \sum_{i=1}^{n} \alpha_i.y_i.x_i \tag{3.5}$$

$$\frac{\partial L}{\partial b}(w, b, \alpha) = 0 \Leftrightarrow \sum_{i=1}^{n} \alpha_i.y_i = 0 \tag{3.6}$$

En substituant (3.5) et (3.6) dans (3.3), on élimine les variables primaires et on obtient la *forme duale* du problème d'optimisation, dans laquelle on cherche les multiplicateurs de Lagrange tels que :

1. les coefficient 1/2 et le carré de $||w||$ sont rajoutés pour simplifier le calcul du dérivée dans les étapes ultérieurs

$$\begin{cases} maximiser & L(\alpha) = & \sum_{i=1}^{n} \alpha_i - 1/2 \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j < x_i, x_j > & \\ Sous & & \alpha_i \geq 0 & i = 1 \ldots n \\ & & \sum_{i=1}^{n} \alpha_i y_i = 0 & i = 1 \ldots n \end{cases} \quad (3.7)$$

Le problème dual a des caractéristiques intéressantes : le nombre d'inconnues est n ($n << p$) indépendamment de la dimension de l'espace des caractéristiques. Pratiquement ce problème à des algorithmes efficaces pour maximiser une fonction quadratique sous contraintes.
Les valeurs de b^* peuvent être obtenues en utilisant n'importe quel exemple critique (x_k, y_k) dans l'équation :

$$b^* = y_k - w^t.x_k = y_k - \sum_{i=1}^{n_k} y_i \alpha_i x_i.x_k \quad (3.8)$$

Où n_k est le nombre d'exemples critiques ou les points supports. Il devient alors possible d'exprimer l'hyperplan solution par :

$$h^*(x) = (w^* x) + b* \quad = \sum_{i=1}^{n} \alpha_i^* y_i < x_i, x > + b^* \quad (3.9)$$

Pour une nouvelle observation x non apprise présentée au modèle, il suffit de regarder le signe de l'expression $\sum_{i=1}^{n} \alpha_i^* y_i < x_i, x > + b^*$

3.1.2.2 L'intérêt de l'approche SVM

Les premiers résultats intéressants découlent directement de l'une des conditions de Kuhn-Tucker (3.4) :

- seuls les points pour lesquels les contraintes du Lagrangien sont actives sont les points tels que $h(x_k) = \pm 1$; les points situés sur les hyperplans de marges maximales. Ce résultat est intéressant en niveau de la complexité puisque, d'une part seul un sous-ensemble restreint de points (les vecteurs supports) participent à la définition de l'hyperplan optimal. Ce sous-ensemble est constitué de n_k exemples qui est généralement bien inférieur à la taille des exemples d'apprentissage. D'autre part, le changement ou l'agrandissement de l'ensemble d'apprentissage a moins d'influence que dans d'autre modèles.
- l'hyperplan solution ne dépend que du produit scalaire entre le vecteur d'entrée et les vecteurs supports, cette remarque est l'origine de la deuxième innovation majeure des *SVM* : le passage, grâce à une fonction noyau, par un espace de redescription de dimension plus grande où des séparation non linéaire peuvent être appliquée (voir sec. 3.1.2.4).

3.1.2.3 Marge souple (Soft Margin)

L'hypothèse que S soit linéairement séparable conditionne beaucoup la résolution du problème (3.2). En effet, il suffit qu'une observation des deux classes viole cette contrainte pour que ce problème n'ait plus de solution. Pour tenter de résoudre ce problème, une idée simple consiste à relâcher les contraintes dans le but d'autoriser quelques erreurs de classification i.e

autoriser quelques exemples d'être à une distance plus faible de la marge correspondante (Voir la figure [Fig. 3.5]). L'hyperplan à marge maximale à été généralisé en introduisant les variables ressorts ou d'écarts (ξ_i)à la marge [Vap95].

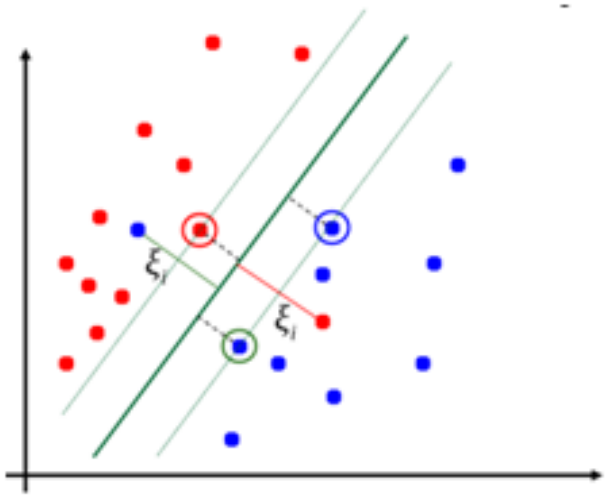

FIGURE 3.5 – Variables ressort.

Le problème (3.2) devient alors :

$$\begin{cases} Maximiser & 1/2\|w\|^2 + C\sum_{i=1}^{p}\xi_k \quad , C > 0 \\ sous & y_i(w.x_i + b) \geq 1 - \xi_k \quad , \xi_k \geq 0 \text{ pour } i = 1 \ldots n \end{cases} \tag{3.10}$$

Autrement dit, on cherche à maximiser la marge en s'autorisant pour chaque contrainte une erreur positive ξ_i, la plus petite possible. Le paramètre supplémentaire C est une constante qui permet de contrôler le compromis entre nombre d'erreurs de classement, et la largeur de la marge. Plus C est important moins d'erreurs sont autorisées. Le paramètre C doit être choisie par l'utilisateur après une recherche exhaustive dans l'espace des paramètres en utilisant la validation croisée par exemple.

En suivant la même démarche que celle de la marge stricte. La différence est la majoration des ξ_i par C.

$$\begin{cases} Maximiser & L(\alpha) = \sum_{i=1}^{n}\alpha_i - 1/2\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j < x_i, x_j > \\ sous & \sum_{i=1}^{n}\alpha_i y_i = 0 \\ & 0 \leq \xi_i \leq C \end{cases} \tag{3.11}$$

Si S est linéairement séparable et C est suffisamment grand, les problèmes (3.7) et (3.11) deviennent équivalents [Vap98].

3.1.2.4 Généralisation aux cas non linéairement séparable

Cette procédure de recherche d'hyperplan séparateur ne permet de résoudre que des problèmes de discrimination linéairement séparables. C'est une limitation sévère qui condamne à ne pouvoir résoudre que des problèmes simple et très particuliers. Pour d'autres problèmes où

les données de deux classes se chevauchent sévèrement, aucun hyperplan séparateur ne sera satisfaisant.
Comme nous l'avons mentionne précédemment seuls les produits scalaires $< x_i, x_j >$ sont nécessaires ; d'où l'idée clé des *SVM* de plonger les observations x_i dans un espace de dimension plus grande (T) à l'aide d'une fonction non-linéaire (fonction noyau), $\phi : R^N \to F$ choisie à priori [BGV92]. Dans ce nouvel espace il est probable qu'il existe un séparateur linéaire.

La figure [Fig. 3.6] montre un problème non linéairement séparable en coordonnées cartésiennes, par contre en coordonnées polaires le problème devient linéaire.

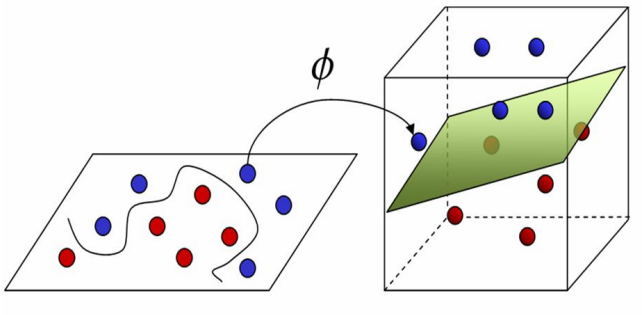

FIGURE 3.6 – Exemple d'un cas non linéairement séparable dans R^2 qui devient séparable dans R^3.

Dans le nouveau espace F, Tout ce qu'il reste à faire c'est de résoudre le problème (3.7) ou (3.11) en remplaçant $< x_i \cdot x_j >$ par $< \phi(x_i) \cdot \phi(x_j) >$. L'hyperplan séparateur obtenu dans l'espace F est appelé *hyperplan optimal généralisé*.

Pour illustrer la transformation dans un espace de dimension plus grande, on définit le noyau $k(x, y) =< \phi(x), \phi(y) >= < x, y >^2$ où la fonction de transformation est la suivante :

$$\phi : (X \equiv R^2) \to (R^3 \equiv T)$$

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \longmapsto \phi(X) \begin{pmatrix} \phi(x)_1 \\ \phi(x)_2 \\ \phi(x)_3 \end{pmatrix} = \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix} \tag{3.12}$$

Dans ce cas le produit scalaire s'écrit :

$$\begin{aligned} < \phi(x), \phi(x\prime) > &= x_1^2 x\prime_1^2 + 2x_1x_2x\prime_1x\prime_2 + x_2^2x\prime_2^2 \\ &= (x_1x\prime_1 + x_2x\prime_2)^2 \\ &= < x, x\prime >^2 \\ &= k(x, x\prime) \end{aligned} \tag{3.13}$$

La fonction noyau a l'avantage de ne pas nécessiter la connaissance de la transformation à appliquer pour le changement d'espace i.e le calcul du produit scalaire dans le nouveau espace ne nécessite pas l'évaluation explicite de ϕ.

3.1.2.5 L'astuce noyau

L'astuce consiste à utiliser une fonction noyau qui vérifie $k(x_i, x_j) =< \varphi(x_i), \varphi(x_j) >$. L'expression de l'hyperplan séparateur en fonction de la fonction noyau devient alors :

$$h(x) = \sum_{i=1}^{n} \alpha_i^* y_i K(x_i, x_j) + b$$

L'intérêt de la fonction noyau est double :

- Le calcul se fait dans l'espace d'origine, ceci est beaucoup moins coûteux qu'un produit scalaire en grande dimension.
- La transformation ϕ n'a pas besoin d'être connue expliciter ; seule la fonction noyau intervient dans les calculs ce qui permet d'envisager des transformations complexes et même un espaces de redescription de dimension infinie.

Toutefois la fonction K doit satisfaire certaines conditions dites conditions de *Mercer* (Voir détails sur les noyaux valides sec. 3.2.1). Sous ces conditions n'importe quelle fonction noyau symétrique, continue, semi-définie positive peut être exprimer comme un produit scalaire dans un espace de grande dimension. Définie positive signifie que pour tous les x_i possibles, la matrice de terme général $k(x_i, x_j)$ est une matrice définie positive c'est-à-dire quelle définit une matrice de produit scalaire.

Quelques noyaux usuels seront présentées dans la sous-section 3.2.3, ainsi que la construction des noyaux nouveaux. Quant au choix de la fonction noyau qui joue un rôle essentiel dans les performances des *SVM*, il sera évoqué dans la section 3.1.3.

3.1.3 SVM et nature des données

L'apprentissage en *SVM* comporte les étapes suivantes [2] :

- Choisir un noyau ;
- Sélectionner le paramètre C ;
- Calculer la matrice $D^{kk\prime}$ (dite matrice de Gram) où les indices $kk\prime$ parcourent toutes les paires d'exemples ;
- Minimiser le Lagrangien dual pour trouver les coefficient α^k ;
- Garder les exemples (entrée et classes) qui sont vecteurs supports et leurs coefficients α^k.

Pour le choix de la fonction noyau quelques noyaux génériques peuvent être employés (cf. sec. 3.2.3), toutefois les performances des *SVM* dépendent pleinement du choix d'un noyau approprié pour le type de données sous-jacent.

Quant au sélection d'hyper-paramètres le recours aux méthodes empiriques (validation croisée) est nécessaire pour faire le meilleur choix.

2. On se met dans le cas le plus général qui est le cas des *SVM* en marge souple

Pour trouver les coefficients α^k de la formule duale, l'algorithme SMO (*Sequential Minimal Optimization*) de Platt [Pla99] propose une méthode de décomposition du problème initial en plusieurs petits sous-problèmes où la résolution de chacun de ces derniers fournit une approximation toujours meilleure de l'optimum. Cet algorithme a été proposé pour palier aux insuffisances de la majorité des méthodes existante qui sont inadaptées aux problèmes de grande taille.

En ce qui nous concerne, rappelons que la classification des textes été parmi les premières applications des *SVM* [Joa98] [DPHS98]. Dans ce domaine, la représentation adoptée est généralement celle de *sac de mots* où chaque documents est codé par un vecteur d'attributs de très grande taille comprenant le lexique complet du corpus.

L'utilisation des *SVM* dans ce cas est alors justifiée par [Joa98] :

- La matrice des profils peut être extrêmement creuse : beaucoup de lignes n'auront alors aucun élément commun. les méthodes qui utilisent le même principe inductive que les *SVM*, sont beaucoup plus performants avec ce type d'instances "creuses" [KWA97].
- Les *SVM* ont le potentiel pour gérer un espace d'entrée très élevé, ce qui permet de faire face à l'effet du sur-apprentissage. Cette "haute-dimensionnalité" ne pose pas un problème pour les *SVM* ce qui n'est pas le cas pour d'autres techniques comme par exemple le classifieur Naïve Bayes.
- La majorité des problèmes de classification de texte sont linéairement séparable.

Les *SVM* ont été étalés à d'autres domaines. Elles ont été exploités dans le domaine médical pour, par exemple, réaliser des diagnostics médicaux : la détection du cancer du sein, l'évaluation du risque d'accidents cardio-vasculaires. Pour le traitement d'images, les *SVM* ont fait des progrès dans la classification des images satellitaires, reconnaissance de caractères manuscrits, reconnaissance de visages[VF99].

Dans la plus part de ces applications les données étaient de type informations globales (sexe, age, bilan sanguin, ... pour un patient ; vecteur de mots et leurs fréquences pour les données de type texte). Cependant certains domaines comme en biologie où l'on cherche à réaliser une classification de protéines, à étudier des séquences pour en extraire des structure secondaire (Intron/exon) du génome (chaînes d'acides aminées) ou à identifier les régions codantes de l'ADN génomique ; nécessitent des informations non plus générale mais plutôt structurelles.

Le choix de la fonction noyau appropriée pour un domaine est essentielle puisque la fonction noyau peut être vue comme un moyen de coder de l'information à priori [DMS+11]. Bien choisi, le noyau permet de matérialiser une notion de "proximité" adaptée au problème de discrimination et à sa structure de données. Dans la section suivante nous allons présenté quelques concepts liées aux noyaux, quelques noyaux usuels utilisés dans un cadre général, ensuite nous nous intéresserons en détails aux noyaux de chaînes et de séquences.

3.2 Noyaux

Un noyau est défini comme un produit scalaire dans un espace où sont projetés tous les objets de départ. Plus précisément, un noyau k(x , y) est une fonction symétrique à deux arguments x et y (x et y sont deux objets appartenant à l'ensemble de départ qui n'est pas nécessairement un espace vectoriel), qui détermine une mesure de similarité généralisée, avec des propriétés mathématiques particulières qui garantissent que le solveur généraliste, auquel on va l'accoupler pour résoudre la tâche, va normalement bien fonctionner (convergence des algorithmes, unicité de la solution) [GY11].

Nous avons vu dans la section 3.1.3 que la fonction noyau reflète une mesure de similarité entre éléments de l'espace d'entrée. Nous avons vu aussi que le choix de la fonction noyau doit traduire aux mieux toute connaissance à priori sur le domaine étudié.

3.2.1 Définitions

Définition 3.2.1.1 ***Produit Scalaire :***
Soit $\mathcal{X}$ un espace vectoriel, l'application $k : \mathcal{X} \times \mathcal{X}$ est un produit scalaire dans $\mathcal{X}$, si k est une application symétrique bilinéaire et strictement positive.

- *Symétrique :* $\forall(\mathrm{x},\mathrm{y}) \in \mathcal{X}^2, k(\mathrm{x},\mathrm{y}) = k(\mathrm{y},\mathrm{x})$
- *Bilinéaire :* $\forall(\mathrm{x},\mathrm{y},\mathrm{z}) \in \mathcal{X}^3, k(\mathrm{x}+\mathrm{z},\mathrm{y}) = k(\mathrm{x},\mathrm{y}+\mathrm{z})$
  et $\forall\alpha \in \mathbb{R}, k(\alpha\mathrm{x},\mathrm{y}) = k(\mathrm{x},\alpha\mathrm{y}) = \alpha k(\mathrm{x},\mathrm{y})$
- *Strictement positive :* $\forall \mathrm{x} \in \mathcal{X} - \{0\}, k(\mathrm{x},\mathrm{x}) > 0$ $(k(\mathrm{x},\mathrm{x}) = 0 \Rightarrow \mathrm{x} = 0)$.

Dans la suite nous allons donner les définitions des fonctions noyaux, fonctions noyaux définies positives et fonctions noyaux valides [Aro50] :

Définition 3.2.1.2 ***Noyau :***
Un noyau k est une fonction telle que :
$\forall(\mathrm{x},\mathrm{y}) \in \mathcal{X}^2 \quad , k(\mathrm{x},\mathrm{y}) =< \Phi(\mathrm{x}), \Phi(\mathrm{y}) >$

avec $\mathcal{X}$ étant l'espace d'entrée ou l'espace de données et Φ est une fonction de transformation associant $\mathrm{x} \in \mathcal{X}$ à son image $\Phi(\mathrm{x}) \in \mathcal{H}$ ($\mathcal{H}$ est un espace de Hilbert) pour tout $\mathrm{x} \in \mathcal{X}$.

Définition 3.2.1.3 ***Matrice de Gram :***
La matrice de Gram est la matrice carrée G de taille $N \times N$ définie pour un ensemble de données $(\mathrm{x_i},\mathrm{y_i})_{i=1}^{N}$ tel que $G_{ij} = k(\mathrm{x_i},\mathrm{y_j})$.

La matrice de Gram (*kernel matrix*) est la source principale d'informations pour les méthodes à noyaux qui n'utilisent que ces information comme données d'entrée des algorithmes d'apprentissage.

Définition 3.2.1.4 ***Noyau semi-défini positif :***
Une fonction $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est dite semi-définie positive sur $\mathcal{X}$
si k est symétrique : $\forall(\mathrm{x},\mathrm{y}) \in \mathcal{X}^2, k(\mathrm{x},\mathrm{y}) = k(\mathrm{y},\mathrm{x})$ *et*

si $\forall N \in \mathbb{N}, (\mathrm{x}_1, \ldots, \mathrm{x}_N) \in \mathcal{X}^N, \forall (a_1, ..., a_N) \in \mathbb{R}^N$, *la matrice de Gram K associée à k (avec* $K_{ij} = k(\mathrm{x_i}, \mathrm{x_j})$*) est semi-définie positive :* $\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j K_{ij} \geq 0$

Théorème 3.2.1 ***Noyau valide :***
Une fonction symétrique continue ou à domaine fini $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *, est un noyau valide si et seulement si k est un noyau semi-défini positif.*

Une fonction de similarité bilinéaire symétrique pourra être utilisée comme noyau valide après s'être assuré qu'elle est semi-définie positive. Dans ce cas il existe un espace [3] dans lequel le noyau pourra être exprimé sous la forme d'un produit scalaire.

Certes que cette définition ne donne aucune indication sur la manière de construire une fonction noyau ni sur la transformation ϕ , mais ce qui intéressant c'est, aucune contrainte n'est imposée sur l'espace d'entrée $\mathcal{X}$. Ainsi un noyau peut être défini sur des données de type varié et complexe : vecteur, graphe, arbre, ... etc.

3.2.2 Définitions de nouveaux noyaux valides

La définition d'un noyau peut s'avérer complexe notamment pour la vérification de sa validité [DMS+11]. La pratique consiste à combiner des noyaux simples pour en obtenir des noyaux plus complexes associés à la situation rencontrée. Pour simplifier la construction de nouveaux noyaux, il est possible de combiner des noyaux valides pour former un nouveau noyau.

Étant donné deux noyaux valides k_1 et k_2 , définis sur $\mathcal{X}^2(\mathcal{X} \subset \mathbb{R}^N)$, une fonction $f : \mathcal{X} \to \mathbb{R}$, une fonction $\Phi : \mathcal{X} \to \mathbb{R}^N$, un noyau $k_3 : \mathbb{R}^N \times \mathbb{R}^N \to \mathcal{X}$, $a \in \mathbb{R}^+$ et B une matrice $N \times N$ symétrique semi-définie positive, les fonctions suivantes sont des noyaux valides [STC04] :

1. $k(\mathrm{x}, \mathrm{y}) = k_1(\mathrm{x}, \mathrm{y}) + k_2(\mathrm{x}, \mathrm{y})$,
2. $k(\mathrm{x}, \mathrm{y}) = ak_1(\mathrm{x}, \mathrm{y})$,
3. $k(\mathrm{x}, \mathrm{y}) = k1(\mathrm{x}, y)k2(\mathrm{x}, \mathrm{y})$,
4. $k(\mathrm{x}, \mathrm{y}) = f(\mathrm{x})f(\mathrm{y})$,
5. $k(\mathrm{x}, \mathrm{y}) = k_3(\Phi(\mathrm{x}), \Phi(\mathrm{y}))$,
6. $k(\mathrm{x}, \mathrm{y}) = \mathrm{x}B\mathrm{y}^T$.

3. cet espace est appelé espace de Hilbert à noyau reproduisant(RKHS)

3.2.3 Quelques noyaux usuels

Des noyaux de base peuvent être employés dans des situations différentes. Voici quelques uns :

Produit scalaire	: $k(\mathrm{x},\mathrm{y}) =< \mathrm{x},\mathrm{y} >$	
Noyau polynomial	: $k(\mathrm{x},\mathrm{y}) = (< \mathrm{x},\mathrm{y} > +c)^d$	$c \in \mathbb{R}^+$ et $d \in \mathbb{N}$ le degré du polynôme
Noyau Gaussien	: $k(\mathrm{x},\mathrm{y}) = exp(-\sigma\|\mathrm{x}-\mathrm{y}\|^2/2\sigma^2)$	
Noyau sigmoïde	: $k(\mathrm{x},\mathrm{y}) = tanh(a < \mathrm{x},\mathrm{y} > -b)$	

Le noyau polynomial était parmi les premiers noyaux à être utilisés notamment dans le cadre de la T.C [Joa98]. il correspond à une transformation vers un espace d'au plus d variables (ou exactement d variables quand $c = 0$). Il est valide, car il est symétrique et défini positif (produit $d$fois du noyau k(x,y) pour le cas où c=0 et d fois de la somme du noyau k(x,y) = 1 et du noyau de base pour l'autre cas).

Pour le cas $(< \mathrm{x},\mathrm{y} >)^d$, ce noyau correspond au produit scalaire dans l'espace qui contient tous les monômes de degré d : par exemple, si x est la représentation sac de mots (un vecteur) d'un objet textuel dans un espace à 3 dimensions qui a pour composantes (u, v, w) l'espace transformé, pour $d = 2$, est celui qui a pour axes les monômes u^2 ; v^2 ;w^2 ; uv ; vw ; uw (uw, par exemple correspond à la fréquence de ce bi-grammes sans prendre l'ordre des mots en compte). L'utilisation de ce noyau, présente l'intérêt de ne pas formuler les d-grammes ; le calcul de de ce noyau prend à peine plus de temps que le calcul du produit scalaire dans l'espace de départ de dimension n [GY11].

Quant au noyau polynomial $k(x, y) = (1+ < x, y >)^d$, il correspond à un espace vectoriel transformé dont les axes contiennent tous les monômes i.e les d-grammes du degré 0 jusqu'au degré d. Dans ce cas aussi, le calcul de ce noyau est à peine plus complexe que le produit scalaire dans l'espace de départ, tout en offrant implicitement une représentation de dimension énormément plus grande.

Le noyau Gaussien est d'une large utilisation [DMS+11] ; il correspond à un espace de caractéristiques de dimension infinie. Les classifieurs qui l'utilise sont appelés *SVM à fonctions de base radiales* (*SVM* à noyaux *RBF*). Pour illustrer ce noyau, on considère une nouvelle observation x non apprise présentée au modèle *SVM* avec un noyau Gaussien. Pour connaître sa classe, il suffit de regarder le signe de l'expression : $y = sgn \sum_{k \in SV} c^k y^k e^{-\sigma\|X-y\|^2/2\sigma^2}$ (voir sec. 3.1.2.1). Dans ce cas, cette classe est donnée par une somme pondérée de Gaussiennes centrées sur les vecteurs supports. Comme les Gaussiennes ont une décroissance rapide, il y a, en général, un seul terme dominant dans la somme : celui du vecteur support le plus proche de x [GY11].

Le noyau du type tangent hyperbolique n'est valide que pour certaines valeurs bien choisies des paramètres a et b mais il permet de retrouver la structure des réseaux de neurones ml-ticouches usuels sans autant donner de justification théorique[Ish07].

3.3 Conclusion

Issue d'un théorème très ancien qui date de 1909 de Mercer [Mer09], ce n'est toutefois qu'en 1992 que les fonctions noyaux ont trouvé leurs grand intérêt où elles étaient rassemblées avec les hyperplans à marge maximale pour donner naissance aux *SVM* [BGV92].
L'approche *SVM* en tant qu'algorithme de classification est une approche élégante. Elle formule le problème de la discrimination à un problème de séparation géométrique à travers la recherche d'un hyperplan séparateur à marge maximale permettant ainsi une séparation linéaire de deux classes (marge dure). L'introduction des variables ressorts (marge souple) [Vap95] a permet de résoudre certaines limitations pratiques importantes, mais les problèmes non linéairement séparables restaient encore sans solution.

Pour les cas non linéairement séparables, l'idée consiste à projeter les données décrites dans l'espace vectoriel d'entrée vers un autre espace de dimension supérieure, éventuellement de dimension infinie où une séparation linéaire est possible. Les régularités linéaires sont cherchées dans ce nouvel espace. Cela est réalisable grâce aux fonctions noyaux qui appliquent une transformation non-linéaire ϕ vers un nouveau espace (espace de redescription) $\phi(x)$. Le problème de cette formulation est qu'elle implique un produit scalaire entre vecteurs dans l'espace de redescription de dimension élevée, ce qui est coûteux en termes de calculs. L'astuce noyau *kernel trick* permet de remédier à ce problème. Grâce à cette astuce les algorithmes de recherche n'ont pas besoin de connaître les coordonnées des projections des données mais seulement de leurs produits scalaires. Ces produits dans un espace de grande dimension peuvent être transformé en une simple évaluation ponctuelle d'une fonction : la fonction noyau.

Deuxième partie

Notre approche de classification de textes en arabe

Chapitre 4

Noyaux pour données textuelles

La grande flexibilité dans la définition des noyaux a permis de définir une notion adaptée de similitude selon la nature des données [CM02]. Dernièrement l'ingénierie des noyaux a connu une immense publication consacrée à la construction de nouveaux noyaux qui encodent les connaissances essentielles sur les problèmes d'induction étudiés.

Pour les entités textuelles, le problème est de savoir déterminer quelle notion de similarité devrait être calculable dans l'espace de redescription où les chaînes ou les séquences sont projetés ; est-ce que on doit les regrouper par longueur, par composition similaire en sous-séquences, ou toute autre propriété. Dans cet esprit et au fil des années, plusieurs noyaux ont été définis. Ils reposent le plus souvent sur le comptage de sous-chaînes ou de sous-séquences communes aux objets comparés. Cela permet, en génomique par exemple, d'estimer une distance évolutive entre génomes, et éventuellement une similarité de fonction puisque on suppose dans ce domaine que l'évolution a opéré par mutations, suppression ou insertion successives.

Dans la suite nous présenterons d'abord quelques concepts et quelques notations qui nous seront utiles. Ensuite nous présenterons les principales fonctions noyaux définies sur les chaînes et les séquences.

Chaîne, sous-chaîne et sous-séquence

On peut définir conformément à l'usage en informatique, une chaîne (*String*) comme étant une suite consécutive d'éléments de base ou d'un alphabet ; quant aux séquences, elles désignent une suite ordonnée mais pas nécessairement consécutive d'éléments.

Un alphabet ou un vocabulaire Σ est un ensemble fini de symboles. $|\Sigma|$ dénote le nombre

de symbole dans Σ et $|s|$ la longueur de la chaîne s. La chaîne de longueur 0 est la chaîne vide (ϵ). Σ^* dénote l'ensemble de toutes les chaînes : $\Sigma^* = \bigcup_0^\infty \Sigma^n$, où Σ^n est l'ensemble de toutes les chaînes finies de longueur n. La notation $[s = t]$ désigne une fonction booléen qui retourne 1 si s et t sont identiques, et 0 sinon. Le i^{eme} élément de la chaîne s est dénoté par $s(i)$ alors que $s(i, j)$ dénote la sous-chaîne (*substring*) $s(i)s(i + 1) \dots s(j)$ de s.

On dit que t est une sous-chaîne de s s'il existe des sous-chaînes u et v telles que $s = utv$. Si u est vide, alors t est le préfixe de s, et t représente le suffixe de s si v est vide. Les sous-chaînes de longueur n sont généralement désignées par l'appellation *N-grammes*.
La chaîne t est une sous-séquence (*subsequences*) de s si elle peut être obtenue par des symboles (pas forcément consécutifs) de s. Formellement, t est une sous-séquence de s s'il existe une suite croissante d'indices $I = (i_1, ..., i_{|t|})$, $(1 \leq i_1 < ... < i_{|t|} \leq |s|)$ tels que $t_j = s_{i_j}$, pour $j = 1, ..., |t|$.

Dans la littérature, on utilise $t = s[I]$ (ou parfois $t = s(I)$) si t est une sous-séquence de s. La longueur d'une sous-séquence t est dénoté par $|t| = |I|$ qui représente le nombre d'indices, et $l(I) = i_{|t|} - i_1 + 1$ désigne le nombre de caractères de s couvrant la sous-séquence t.

4.1 Noyau *P-spectre*

Le noyau *P-spectre* (*P-spectrum*) [LEN02] est le noyau le plus simple défini sur des chaînes. Il est connu le plus souvent par le noyau *N-grammes*.

Pour ce noyau un texte est représenté dans un espace indexé par toutes les sous-chaînes possibles de symboles de longueur p. La composante u d'une chaîne (ou d'un texte) s dans cet espace vaut le nombre d'occurrences de la sous-chaîne u (de longueur p) dans s et elle est donnée par :

$$\phi_u^p(s) = card\{(v_1, v_2) \quad : s = v_1 u v_2 \quad , \quad u \in \Sigma^p\}$$

Le noyau *P-spectre* est le produit scalaire dans cet espace :

$$K_p(S_1, S_2) =< \phi^p(S_1), \phi^p(S_2) >$$

Il est donc valide par définition.
Il est à noter que la dimensionnalité générée par le N-grammes est trop élevée même avec une valeur de "n" modérée. Mais généralement les N-grammes ne sont pas tous présent dans un corpus réduisant ainsi la dimension d'une façon considérable (par exemple pour le corpus *Reuters*, il n'y a que 8727 unique tri-grammes sans les mots outils).
Exemple : Étant donnée le vocabulaire $\Sigma = \{A, B, D, S, Y\}$. L'espace de projection défini par le noyau *2-spectre* sur ce vocabulaire est l'espace Σ^2 de toutes les paires construites sur Σ :
$\Sigma^2 = \{AA, AB, AD, AS, AY, BA, BB, BD, BS, BY, DA, DB, DD, DS, DY, SA, SB, SD, SS, SY, YA, YB, YD, YS, YY\}$.
Tout mot construit sur Σ peut être représenté dans cet espace. Les représentations des mots S_1 = "SAY" , S_2 = "BAY" , S_3 = "SAD" et S_4 = "BAD" dans cet espace sont données par

le tableau 4.1 (on se contente des composantes non nulles) :

ϕ^2	AD	AY	BA	SA
SAY	0	1	0	1
BAY	0	1	1	0
SAD	1	0	0	1
BAD	1	0	1	0

TABLE 4.1 – Composantes (non nulles) des mots "SAY", "BAY", "SAD" et "BAD" représentés dans l'espace du noyau *2-spectre*

Le noyau K_2 est alors le produit scalaire dans cet espace :

$$K_2(s_1, s_2) =< \phi^2(s_1), \phi^2(s_2) >= \sum_{u \in \Sigma^2} \phi_u^2(s_1)\phi_u^2(s_2)$$

La matrice des noyaux entre les 4 mots de l'exemple précédent est donnée par le tableau 4.2

ϕ^2	SAY	BAY	SAD	BAD
SAY	2	1	1	0
BAY	1	2	0	1
SAD	1	0	2	1
BAD	0	0	1	2

TABLE 4.2 – Matrice du noyau *2-spectre* entre les mots "SAY", "BAY", "SAD" et "BAD"

La complexité du calcul de la fonction noyau d'ordre p entre deux chaînes s et t est $O(p.|s|.|t|)$ puisque l'implémentation (naïve) de ce noyau doit formuler tous les p-grammes des deux chaînes s et t et de les comparer. Plusieurs méthodes de complexité relativement faible (en $O(p.max(|s|, |t|))$ ont été mise en œuvre pour le calcul efficace de ce noyau et cela en se basant sur des algorithmes classique de comparaison de chaînes (*string matching*) utilisant une représentation en "arbre des préfixes" [GY11].

4.2 Noyau toutes-sous-séquences

Le noyau *toutes sous-séquences* (*All-Subsequences*) [LK03] permet de tenir compte de toute sous-séquences. Pour ce noyau un texte est représenté dans un espace indexé par toutes les sous-séquences (non nécessairement contiguës) possibles de toutes les longueurs possibles.
Le u^{eme} composant de la fonction de projection ϕ dans l'espace de redescription indique la fréquence d'occurrence de la séquence u dans s, et il est défini par :

$$\phi_u = \sum_{i:s[i]=u} 1$$

La contribution de cette composante (pour la chaîne u) dans le produit scalaire entre les deux chaînes s et t est :

$$\begin{aligned} < \phi^u(s), \phi^u(t) > &= \sum_{i:u=s(i)} \sum_{j:u=t(j)} 1 \\ &= \sum_{(i,j):u=s(i)=t(j)} 1 \end{aligned}$$

Le noyau "*toutes sous-séquences*" est le produit scalaire dans cet espace

$$K_A(s,t) =< \phi(S), \phi(t) > \quad = \sum_{u \in I} \sum_{(i,j):u=s(i)=t(j)} 1 \quad = \sum_{(i,j):s(i)=t(j)} 1$$

Ce noyau est valide par définition et i , j sont des n-uplets d'indices strictement ordonnés.

Considérant l'exemple des termes "SAY", "SAA" et "BAY". Le tableau 4.3 représente tous les sous-séquences de ces termes (ici les sous-séquences non possibles telles que "AS" sont ignorées). La matrice de ce noyau est donnée par le tableau 4.4

	ϵ	A	B	S	Y	SA	AY	BA	AA	SY	BY	SAY	SAA	BAY
SAY	1	1	0	1	1	1	1	0	0	1	0	1	0	0
SAA	1	2	0	1	0	2	0	0	1	0	1	0	1	0
BAY	1	1	1	0	1	0	1	1	0	0	1	0	0	1

TABLE 4.3 – Composantes (non nulles) des chaînes "SAY", "SAA" et "BAY" représentées dans l'espace du noyau toutes sous-séquences.

K_A	SAY	SAA	BAY
SAY	8	6	4
SAA	6	12	4
BAY	4	4	8

TABLE 4.4 – Matrice du noyau "toutes sous-séquences" entre les chaînes "SAY","SAA" et "BAY"

Ce noyau est capable de capturer tous les motifs communs à deux séquences cependant l'espace de projection est de très haute dimension où un calcul explicite du produit scalaire dans cet espace devient rapidement infaisable. Il faut recourir à une implémentation récursive. En effet, il suffit de remarquer que toute sous-séquences de s contenant le dernier caractère a de s, tel que $s = s'a$, ne peut apparaître dans t qu'entre le premier caractère de t et la dernière occurrence de a dans t. Ainsi, nous avons :

$$\begin{aligned} k(s,\epsilon) &= 1 \\ k(s'a,t) &= k(s',t) + \sum_{k:t[k]=a} k(s', t[1 \ldots k-1]) \end{aligned}$$

Cette définition récursive permet d'exploiter les techniques de la programmation dynamique pour aboutir à une complexité de $O(|s|.|t|)$ [GY11]. Cependant, il existe une version plus efficace en $O(|s| + |t|)$ [VS03].

4.3 Noyau sous-séquence de longueur fixe

Le noyau "sous-séquence de longueur fixe" (*p-Fixed length Subsequences*) [STC04] constitue une version plus légère du noyau "toutes sous-séquences" en se limitant seulement aux sous-séquences d'une taille fixe p.
Pour ce noyau un texte est représenté dans un espace indexé par toutes les sous-séquences possibles de longueur p. La composante u d'une chaîne s dans cet espace vectoriel vaut le nombre d'occurrences de la sous-séquence u dans s, même sous forme non-contiguë. Ainsi la fonction de projection $\phi(s)$ sera composée des éléments $\phi_u(s)$ tels que $|u| = p$.

Évidemment le noyau est le produit scalaire dans cet espace et il est valide par définition. Un calcul explicite du produit scalaire dans cet espace devient rapidement infaisable et il faut recourir, une fois encore, à une implémentation récursive. Celle-ci donne lieu aux équations de récurrence suivantes :

$$\begin{array}{lll} K_0(s,\epsilon) & = 1 & \\ K_p(s,\epsilon) & = 0 & \text{pour } p > 0 \\ K_p(s'a,t) & = K_p(s',t) & + \sum_{k:t[k]=a} K_{p-1}(s', t[1 \ldots k-1]) \end{array}$$

Cette récursion est définie sur la séquence en retirant à chaque étape le dernier élément de la séquence et en tenant compte de la taille du motif : si le dernier caractère du motif a été fixé, le préfixe du motif ne peut être constitué que de $p-1$ éléments.

L'intérêt de cette récursion réside dans le fait qu'on calculant le noyau pour la longueur p, on a du même coup, comme résultat intermédiaire, tous les noyaux pour les longueurs inférieurs à p.

La complexité de cette implémentation est de $O(p.|s|.|t|)$ [GY11] et comme le noyau précédent il existe une version plus efficace en $O(p.(|s| + |t|))$ [VS03].

4.4 Noyau de sous-séquence avec pénalité sur l'écart

L'un des inconvénients des noyaux de sous-séquences est qu'ils ne tiennent pas compte de la distance séparant les éléments non consécutifs ; par exemple la sous-séquence "gon" est une sous-séquence des trois séquences "gone" , "going" et "galleon", mais elle est plus similaire à la première qu'aux autres. Les noyaux "toutes sous-séquences" et "sous-séquence de longueur fixe attribueront la même valeur aux couples ("gone", "gon"), ("going", "gon") et ("galleon", "gone").

Le noyau de sous-séquence avec pénalité sur l'écart (*GWSK* : *Gap-Weighted Subsequences Kernel*) [LSST+02] permet de tenir compte des "trous" dans les sous-séquences communes. Pour cela ce noyau introduit une fonction ($\lambda \in]0,1]$) de pénalité sur l'écart (*gap penalty*) pour mesurer la distance entre les éléments d'une sous-séquence.

La composante u d'une chaîne s dans cet espace vaut le nombre d'occurences de la sous-séquence u dans s multiplié par le facteur de pénalisation $\lambda^{l(I)}$, la longueur réelle (incluant les

symboles appariés et les trous) de u dans s :

$$\phi_u^p(s) = \sum_{I:u=s[I]} \lambda^{l(I)},\ u \in \Sigma^p.$$

Ainsi le noyau associé peut être donné par :

$$\begin{aligned} K_G(s,t) &= \langle \phi^p(s), \phi^p(t) \rangle \\ &= \sum_{u \in \Sigma^p} \phi_u^p(s).\phi_u^p(t) \\ &= \sum_{u \in \Sigma^p} \sum_{I:u=s[I]} \sum_{J:u=t[J]} \lambda^{l(I)+l(J)} \end{aligned}$$

Pour illustrer ce noyau, on considère les chaînes "bat", "car" et "cat". Pour $p = 2$, la transformation dans l'espace de redescription est donné par le tableau 4.5 :

ϕ	ar	at	ba	br	bt	ca	cr	ct
bat	0	λ^2	λ^2	0	λ^3	0	0	0
car	λ^2	0	0	0	0	λ^2	λ^3	0
cat	0	λ^2	0	0	0	λ^2	0	λ^3

TABLE 4.5 – Projection de toutes les sous-séquences de taille 2 des chaînes "bat", "car" et "cat" pour le noyau *GWSK*.

Il est possible d'employer les termes d'un corpus comme symboles d'un alphabet sur lequel on défini ce noyau. Dans ce cas on retrouve le modèle vectoriel classique (sans tenir compte de l'ordre) pour $p = 1$. Pour $p > 1$ et $\lambda = 1$, on retrouve un noyau qui ressemble au noyau polynomial sur "sacs de mots", mais en tenant compte de l'ordre des termes.

D'autres variantes ont été dérivée à partir de ce noyau. Parmi eux, ceux qui exploitent des informations à priori sur le type de termes manipulés. On peut citer celle qui consiste à utiliser un facteur de pénalisation λ dépendant de la classe du terme. La classe du terme peut être la classe de fréquence du terme (pour obtenir un effet *IDF*), son appartenance à une liste de mots vide, ou sa fonction grammaticale (nom, verbe, adjectif, etc.). Aussi le facteur de pénalisation peut être défini différemment selon que l'on distingue, dans la longueur effective de la sous-séquence u, la contribution des trous (c'est-à-dire des symboles non-appariés) et des symboles appariés. Cela est réalisable en associant deux valeurs de λ à chaque symbole s : $\lambda_{s,match}$ et $\lambda_{s,gap}$.

Pour analyser les performances des différents noyaux de chaînes et de séquences ; différentes expérimentations ont été menées. Les résultats montrent, dans le cadre de la T.C, que les noyaux *GWSK* et *P-spectre* sont plus performants que l'approche standard du sac de mots [LSST+02]. De plus, le noyau *P-spectre* donne de bonnes résultats lorsque p est choisi convenablement. Toutefois, le noyau *GWSK* est le plus performant lorsque la valeur de pondération

est choisie judicieusement [Suj07].

4.5 L'aspect calculatoire et transducteurs

Le recours à des optimisations comme la programmation dynamique peut rendre linéaire, par rapport à la taille des documents, la complexité d'évaluation de chaque fonction noyau. Mais l'évaluation globale de tous les noyaux i.e. la construction de la matrice de Gram (qui peut être très volumineuse pour certains corpus) nécessite des ressources importantes et demande d'autres optimisations telles que les approches d'approximation (de la matrice de Gram) et des techniques d'implémentations efficientes. Approximer cette matrice par une autre matrice moins importante est une alternative intéressante. Cela est réalisable en se limitant à un sous-ensemble restreint de composantes ou d'attributs générés par ces noyaux. La similarité entre ces deux matrices est garantie par des métriques telles que "*mesure d'alignement*" [CKEST01] pour l'implémentation des approches dites approximatives.

Une autre approche radicalement différente consiste à adopter les transducteurs pour l'implémentation de ces noyaux. En effet, les transducteurs fournissent une représentation compacte et simple de noyaux de séquences. En outre, les algorithmes classiques (composition des transducteurs, le plus court chemin ...) peuvent être utilisés pour calculer efficacement ces noyaux [CHM+04].

La section suivante introduit les noyaux rationnels et quelques aspects très sommaires liés aux transducteurs.

4.5.1 Transducteurs

Nous proposons ici un rappel de quelques notions utiles concernant les transducteurs avant de donner la définition des noyaux rationnels. Les transducteurs sont définis avec la notion algébrique de semi-anneau. Une structure algébrique $(\mathrm{k}, \oplus, \otimes, \overline{0}, \overline{1})$ est un semi-anneau si :

$(\mathrm{k}, \oplus, \overline{0})$ constitue un monoïde commutatif où l'élément neutre est $\overline{0}$; $(\mathrm{k}, \otimes, \overline{1})$ forme un monoïde où l'élément neutre est $\overline{1}$; $\otimes$ est distributive par rapport à $\oplus$ et $\overline{0}$ est l'élément absorbant de $\otimes$ i.e $(a \otimes \overline{0}) = (\overline{0} \otimes a) = \overline{0}$ pour tout $a \in \mathrm{k}$.

On peut donner la définition formelle d'un transducteur par [Ber79] :

Définition 4.5.1.1 *Un transducteur pondéré à états finis (WFST* Weighted Finite-State Transducer*) T défini sur le semi-anneau* k *est un 8-tuples* $< \Sigma, \Delta, Q, I, F, E, \lambda, \rho >$ *où* Σ *désigne un alphabet fini d'entrée (destiné à être reconnu),* Δ *désigne une alphabet fini de sortie (ou de réécriture),* Q *est un ensemble fini d'états,* $I \subseteq Q$ *l'ensemble des états initiaux,* $F \subseteq Q$ *l'ensemble des états finaux,* $E \subseteq Q \times (\Sigma \cup \epsilon) \times (\Delta \cup \epsilon) \times \mathrm{k} \times Q$ *un ensemble fini de transitions,* $\lambda : I \to \mathrm{k}$ *la fonctions d'attribution des poids initiaux et* $\rho : F \to \mathrm{k}$ *la fonction d'attribution du poids final.*

Étant donnée une transition $e \in E$, on dénote $p[e]$ son état précédent ou d'origine et $n[e]$ son état suivant ou de destination. Un chemin $\pi = e_1 \dots e_k$ est un élément de E^* dont ses transitions

consécutives sont $n[e_{i-1}] = p[e_i]$, $i = 1 \ldots, k$. On étend n et p à un chemin π en fixant $n[\pi] = n[e_k]$ et $p[\pi] = p[e_1]$. On dénote par $w[\pi]$ le poids du chemin π calculé par la multiplication ($\otimes$) de l'ensemble des poids des transitions qui le constituent : $w[\pi] = w[e_1] \otimes \cdots \otimes w[e_k]$. On dénote aussi $P(p, q)$ l'ensemble de tous les chemins de p à q et $P(p, x, y, q)$ l'ensemble de tous les chemins de p à q qui ont comme entrée $x \in \Sigma^*$ et en sortie $y \in \Sigma^*$. Un *cycle* représente un chemin π où l'origine et la destination coïncident : $p[\pi] = n[\pi]$. Un transducteur T est dit *acyclique* s'il n'admet aucun cycle. T^{-1} dénote le transducteur *inverse* de T obtenu à partir de T en échangeant les labels des entrées et les labels des sorties pour chaque transition ainsi que l'alphabet d'entrée et de sortie. Un transducteur T est dit *régulier* si sa fonction d'attribution de poids de sortie pour n'importe quel couple d'entrée-sortie $(x, y) \in \Sigma^* \times \Sigma^*$ est une application et appartient à k. Dans ce cas le poids associe par T pour le couple (x, y) représente la somme des poids de tous les chemins ayant comme entrée x et comme sortie y et il est donné par :

$$[[T]](x, y) = \sum_{\pi \in p(I,x,y,F)} \lambda(p[\pi]) \otimes w[\pi] \rho[n[\pi]]$$

On suppose désormais que les transducteurs sont acycliques et réguliers.

Les transducteurs sont fermés sous un ensemble d'opérations appelées *opérations rationnelles* dont on peut citer les plus fondamentales :
$\forall (x, y) \in \Sigma^* \times \Sigma^*$ on a :

- L'union (ou la somme) de deux transducteurs pondérés T_1 et T_2 définie par :

$$[[T_1 \oplus T_2]](x, y) = [[T_1]](x, y) \oplus [[T_2]](x, y)$$

- La concaténation (ou la production) de deux transducteurs pondérés T_1 et T_2 définie par :

$$[[T_1 \otimes T_2]](x, y) = \oplus_{x=x_1x_2, y=y_1y_2} [[T_1]](x_1, y_1) \otimes [[T_2]](x_2, y_2)$$

- La composition de deux transducteurs pondérés T_1 et T_2 dont l'alphabet d'entrée est identique à l'alphabet de sortie :

$$[[T_1 o T_2]](x, y) = \oplus_{z \in \Sigma^*} [[T_1]](x, z) \otimes [[T_2]](z, y)$$

La composition est une opération fondamentale car elle permet de construire des transducteurs complexes à partir de ceux moins complexes.

La construction de $T_1 o T_2$ se base sur la correspondance entre les transitions de T_1 et T_2 : étant donnée la transition (q_1,a,b,w_1,q_2) de T_1 et une autre transition (q_1',b,c,w_2,q_2') de T_2. L'algorithme de composition construit tous les transitions ((q_1,q_1'), a , c , $w_1 \otimes w_2$, (q_2,q_2')). La figure 4.1 montre la composition de deux transducteurs pondérés : 4.1-c est la composition de 4.1-a et 4.1-b.

On a au pire cas toutes les transitions de T_1 sortantes de q_1 correspondent à toutes les transitions de T_2 sortantes de q_1', par conséquent l'algorithme de composition est d'une complexité quadratique $O(|T_1||T_2|)$ [CHM03] où $|T|$ représente le nombre d'états et des transitions d'un

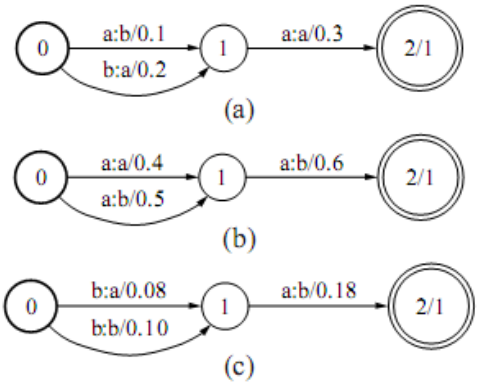

FIGURE 4.1 – Composition de deux transducteurs pondérés.

transducteur T .

4.5.2 Noyaux rationnels

Un noyau rationnel est défini comme suit [CHM+04] :

Définition 4.5.2.1 ***Noyau rationnel***
Un noyau K *défini sur* $\Sigma^* \times \Delta^*$ *est dit rationnel s'il existe un transducteur pondéré* $T < \Sigma, \Delta, Q, I, F, E, \lambda, \rho >$ *défini sur le semi-anneau* k *et une fonction* $\psi : \mathrm{k} \to \mathbb{R}$ *tel que pour tout* $x \in \Sigma^*$ *et* $y \in \Delta^*$ *:*
$\mathrm{K}(x, y) = \psi([[T]](x, y))$.

Aucune restriction est imposée sur la fonction ψ qui peut être une fonction quelconque (par exemple dans le cas du semi-anneau probabilité i.e $(\mathbb{R}_+, +, \times, 0, 1)$ on emploi $\psi : \mathbb{R}_+ \to \mathbb{R}$).

Les noyaux rationnels définissent une classe générale de noyaux basé les transducteurs. Les noyaux de séquences constituent un exemple de cette classe. Les figures 4.2 et 4.3 [LK03] montrent les transducteurs associés respectivement aux noyaux tri-grammes et Gappy tri-grammes pour l'alphabet {a,b} :

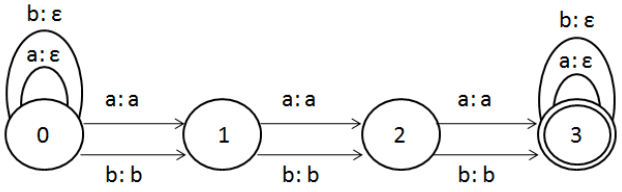

FIGURE 4.2 – Le transducteur associé au noyau tri-grammes [LK03].

Cependant, il faut signaler que ces noyaux sont différents de celui de *GWSK*. Par exemple, le noyau entre les deux chaînes "abc" et "acbc" est évalué à 8 du moment que l'ensemble des sous-séquences communes entre "abc" et "acbc" est $\epsilon, a, b, c, ab, ac, bc, abc$; mais pour le noyau *GWSK* de *Lodhi et al* [LSST+02] est évalué à 10 puisque les sous-séquences "c" et "ac" apparaissent deux fois dans la deuxième chaîne.

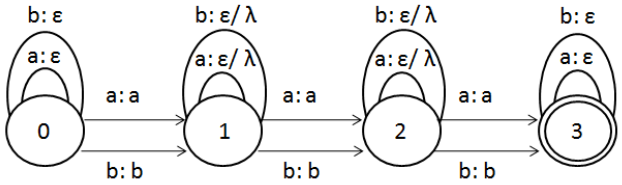

FIGURE 4.3 – Le transducteur associé au noyau Gappy tri-grammes [LK03].

Malheureusement pas tous les transducteurs représentent des noyaux valides. Le théorème suivant fournit un cadre général pour la construction d'un nouveau noyau rationnel valide à partir d'un transducteur quelconque [CHM+04] :

Théorème 4.5.1 *Étant donnée un transducteur pondéré T et supposant que ToT^{-1} est un transducteur régulier, alors ToT^{-1} définit un noyau valide sur $\Sigma^* \times \Sigma^*$*

Pour évaluer un noyau rationnel K entre deux chaînes $(x, y) \in \Sigma^* \times \Sigma^*$ représentées respectivement par deux transducteurs A et B, on peut employer l'algorithme général du calcul des noyaux rationnels. Cet algorithme consiste à construire le transducteur composé $N = AoToB$ où T est le transducteur associé à K ; ensuite calculer $w[N]$ qui représente la somme des poids de tous les chemins de N.

La complexité de cet algorithme revient à la complexité de l'opération de la composition en supposant que $w[N]$ peut être calculé en un temps constant, par conséquent cette complexité est de $O(|T||A||B|)$ [CM09].

4.6 Esquisse de notre système de classification

L'état d'art de la classification de textes en langue arabe que nous avons présenté dans le chapitre 2 a fait ressortir un manque considérable dans les travaux s'intéressant à ce thème. Néanmoins dans ces travaux, différentes techniques ont été exploitées allant d'une mise en œuvre d'une approche statistique basée sur des mesures de distance (*Dice*, *Manhattan*, *Entropie* ...), application des algorithmes classiques (*K-ppv*, *N.B*, *Arbres de décision* ...) jusqu'à l'utilisation des *SVM*. Ces différentes implémentations ont choisi comme composantes élémentaires le caractère (généralement le 3-grammes) ou le terme (terme brute, terme dé-suffixé ou racine).

Concernant les systèmes de classification à base des *SVM*, les résultats empiriques ont montré une nette amélioration dans les performances de ces systèmes (voir détails dans 2.3.2). Toutefois l'exploitation des différents noyaux notamment ceux appropriées pour les données textuelles n'a pas fait l'objet d'une attention particulière.

Exploiter ces différents noyaux est notre objectif notamment pour réaliser une étude comparative d'un système de classification basé sur une composante *terme* par rapport à une composante "*séquence de terme*". Les séquences de termes (ou les phrases) sont plus informatives que les mots seuls. Par exemple le terme composé "رخصة قيادة" (Permis de conduire) désigne un sens particulier qui peut être perdu si on le décompose.

Cette idée est aussi encouragée par la disponibilité d'une panoplie des outils et des bibliothèques permettant l'implémentation des SVM, en plus des fondements théoriques solides, faisant de cette méthode un outil incontournable pour l'apprentissage automatique.

La figure 4.4 montre l'esquisse de notre solution. Elle illustre les différentes étapes des deux phases d'un système de classification : construction du modèle d'apprentissage et prédiction.

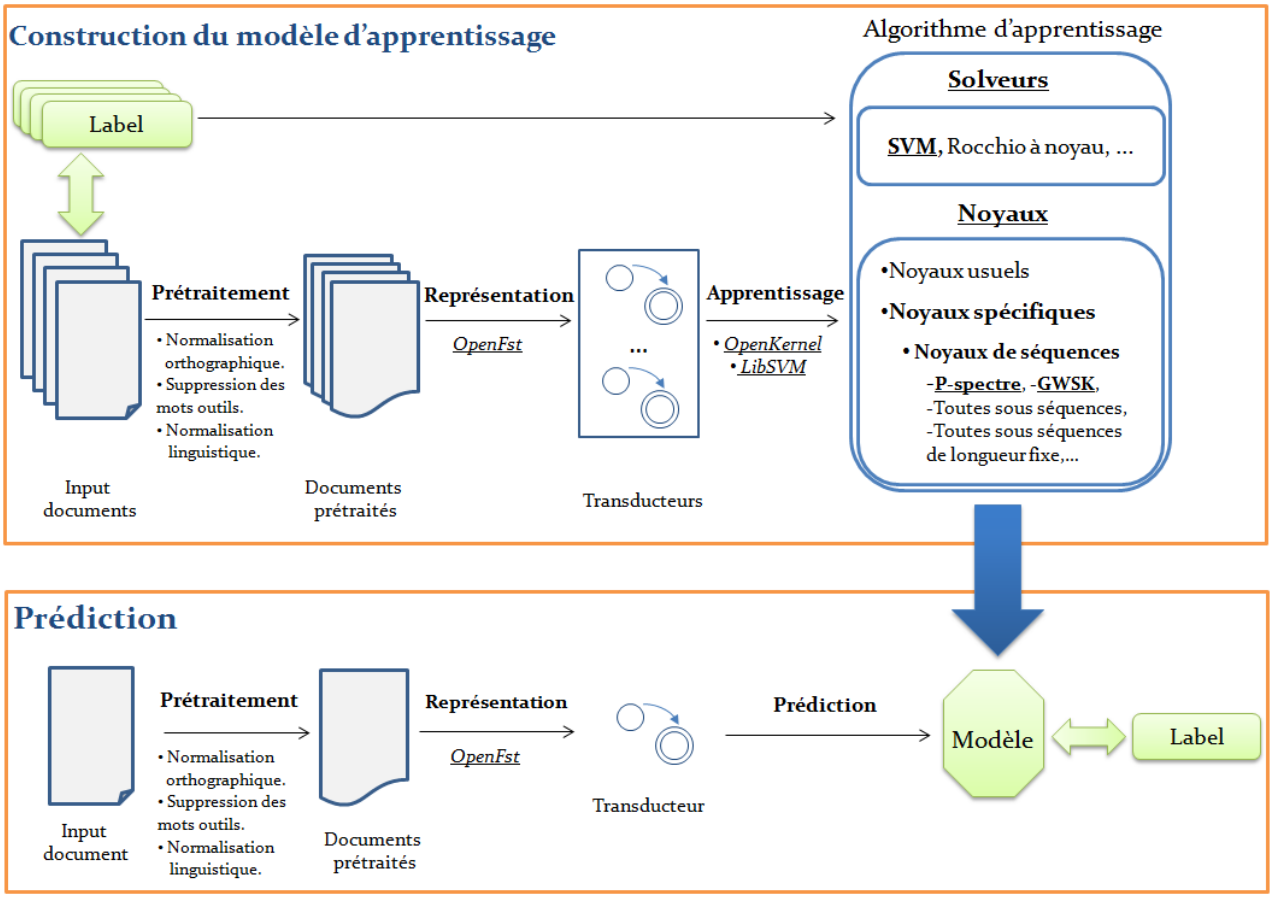

FIGURE 4.4 – Esquisse du classifieur proposé.

Rappelons qu'un modèle de classification ou un classifieur, avant d'être utilisé pour prédire la catégorie d'un nouveau document texte, est construit par apprentissage sur un ensemble d'exemples de documents textes pré-labellisés. Dans la phase d'apprentissage, l'ensemble des textes est soumis à un prétraitement avant d'être représenté sous forme de transducteur sur lesquels les SVM construiront un modèle de catégorisation. ce modèle, une fois validé, peut être utilisé pour la prédiction. Notons que les outils utilisés dans la partie expérimentation sont soulignés.

Prétraitement

Dans l'étape de prétraitement, les textes en langue arabe doivent subir une normalisation orthographique de l'ensemble de leurs termes pour normaliser (ou grouper) différentes formes d'une même lettre. Ensuite et après la suppression des mots outils, la normalisation orthographique est généralement opérée. Pour les textes en langue arabe, utiliser les termes tels qu'ils sont (termes brutes) ou choisir entre extraction de racine ou *light-stemming* n'est pas un choix évident. L'état d'art n'a pas permet de tirer une conclusion claire concernant cet aspect : parmi les conclusions de Mesleeh [Mes07] le *light-stemming* n'améliore pas les performances des systèmes de TC ; toutefois il faut rappeler que l'auteur s'est contenté de 162 termes dans la phase de réduction de dimension, ce qui repose la question quant à la généralité de sa conclusion. D'un

autre coté, entre une représentation basée sur les racines ou sur termes dé-suffixés ; rappelons qu'en plus de l'ambiguïté que peut créer le procédé d'extraction de racine (un seul terme peut engendrer plusieurs racines possibles), aucune amélioration significative sur les performances générales d'un système de TC est acquise en utilisant l'extraction de racine à l'instar du système de EL-Hallees [EH08]. Par conséquent, nous avons opté pour l'approche de stemming assoupli qui peut réduire considérablement la taille du vocabulaire (voir chapitre expérimentation) qui est un choix intéressant dans notre cas où chaque terme doit être représenté par une transition.

Représentation

Les documents pré-traités sont transformés sous forme de transducteurs avant de lancer l'apprentissage. Rappelons que les noyaux de séquences en l'occurrence *P-spectre* et *sous séquence avec pénalité sur l'écart* peuvent être évalués efficacement sur ce type de structure.

Cette transformation est réalisée par la bibliothèque *OpenFST*[1]. Cette bibliothèque est consacrée à la construction, la combinaison, l'optimisation des transducteurs pondérés à états finis (*FST*). Elle est souple, efficace et adaptée à un grand nombre de problèmes [ARS+07]. Pour créer un FST, *OpenFST* offre la possibilité de le faire via le mode ligne de commandes ou par programme. Dans le deuxième cas, on peut employer des constructeurs et des fonctions C^{++} pour créer des états, définir un état comme état initial ou final, créer des transitions (avec un alphabet d'entrée et un alphabet de sortie) et leurs associer des poids. On peut aussi convertir un transducteur dans plusieurs formats graphiques standards.

Dans notre cas, chaque document texte est représenté par un transducteur linéaire où les transitions représentent les termes pris dans le même ordre du document. Après leurs création, l'ensemble de ces transducteurs sont regroupés dans un seul fichier archive (fichier *far*) pour l'utiliser pour l'apprentissage.

Apprentissage

La représentation en *FST* permet d'utiliser une bibliothèque assez importante. Il s'agit de la bibliothèque *OpenKernel*[2] qui est une bibliothèque open source pour la manipulation des noyaux rationnels. Elle permet d'implémenter efficacement les noyaux *n-grammes*, *gappy n-grammes* ou tout autre noyaux rationnels. Pour réaliser l'apprentissage, *Openkernel* incorpore une version adaptée de *LIBSVM* [CL11] pour la quelle deux fichiers (*training dataset*, *test dataset*) doivent être spécifiés dans un format standard. Le premier (*training dataset*) est celui utilisé pour l'apprentissage c.à.d. celui sur lequel *LIBSVM* va construire le modèle de classification.

1. source : http ://www.openfst.org. Il s'agit d'un projet open source qui a été développé par *AT&T* et des contributeurs tels que Google Research et NYU Courant Institute.
2. http ://www.openkernel.org/

Le second fichier (*test dataset*) est utilisé pour la prédiction ou pour évaluer le modèle construit ; en comparant les étiquettes réelles de l'ensemble spécifié dans ce fichier par rapport à l'étiquetage réalisé par ce modèle. *LIBSVM* a besoin aussi (en plus de ces deux fichiers et le fichier archive englobant l'ensemble des transducteurs représentants tous les documents) du fichier représentant l'alphabet Σ complet, dans notre cas Σ représente le vocabulaire du corpus.

Dans le chapitre suivant nous présenterons en détail la mise en œuvre de cette approche, le corpus de test et les résultats obtenus.

4.7 Conclusion

La puissance des *SVM* ne réside pas uniquement dans l'utilisation des algorithmes de recherche de régularités linéaires pour la recherche de régularités non linéaires, mais aussi dans l'emploi de ces algorithmes sur des données complexes non nécessairement numériques du moment qu'aucune contrainte n'est imposé sur l'espace d'entrée. Encore, le choix (ou la définition) d'un noyau qui reflète le mieux la similitude entre les données du problème d'induction étudié, associe aux méthodes à noyaux une grande efficacité.

Dans ce contexte, l'ingénierie des fonctions noyaux a connu un vrai élan pour définir de nouveaux noyaux selon la problématique posée et selon la nature des données (textes, arbres, graphes ...) en l'occurrence celles adaptées aux chaînes et séquences telles que les noyaux *P-spectre*, *toutes sous-séquences*, *sous-séquence de longueur fixe*, *sous-séquence avec pénalité sur l'écart*.

Ces noyaux, souffrant du problème de calculabilité, ont vu plusieurs optimisations. Parmi ces solutions celle consistant à leurs associer (ou de leurs construire) des transducteurs comme modèle de représentation. Les noyaux associés aux transducteurs appelés noyaux rationnels définissent une classe générale de noyaux qui non seulement bénéficie d'une représentation compacte par le biais des transducteurs mais aussi ils peuvent être calculés efficacement en utilisant l'opération de composition des transducteurs [CHM+04].

Cette étude nous a mené vers une solution radicalement différentes de la totalité des travaux réalisés jusqu'à ce jour dans le cadre de la classification des textes en langue arabe. Rappelons que les classifieurs à base de *SVM* n'ont exploité qu'une représentation vectorielle des textes en *sac de mots* avec les noyaux usuels. Encore, les aspects liés aux cooccurences et ordre des termes dans un texte sont carrément négligés. Explorer cet axe pour la classification des textes en langue arabe est l'intérêt de cette étude.

Le chapitre suivant présente l'étude empirique de notre solution.

Chapitre 5

Expérimentation

Dans la littérature de la T.C en langue arabe les composants choisis variés entre termes et caractères. Cependant, tous les travaux antérieurs ont ignoré les aspects ordre et cooccurrence de ces composants. notre approche proposée, décrite dans le chapitre précédent, permet de considérer ces aspects.

A fin d'évaluer notre approche, notre démarche consiste à réaliser deux lots d'expérimentation :

- Le premier, réalisé à titre indicatif ; est basée sur une représentation *sac de mots*.
- Le deuxième consiste à choisir les termes comme composants de base dans le but d'évaluer un système de classification basé sur des termes composés (N-grammes de mots).

Rappelons que pour la deuxième expérimentation, les termes d'une séquences ne sont pas forcément consécutifs. Pour la réalisation, nous avons fait recours aux noyaux N-grammes (*P-spectre*) et *Gappy N-grammes*. Ce dernier a deux paramètres k et λ : k représente la taille de la séquence de termes et λ représente le facteur d'écart introduit pour favoriser (ou défavoriser) les séquences dont les termes sont les moins dispersés (plus dispersés) dans une séquence.

Avant de présenter les résultats, nous décrivons notre collection de test sur la quelle on a fait notre expérimentation ainsi que les prétraitements que nous avons réalisé sur cette collection.

5.1 Description du corpus de test

Nos expérimentations sont réalisées sur le corpus SPA (Saudi Press Agency) [AAA+08]. Ce corpus contient 1520 documents divisés sur 6 catégories (Culture, Économie, Générale, Politique, Sociale, sport). Le corpus est codé en Windows-1256. La taille du vocabulaire de ce corpus est d'environ 253.472 termes dont 36.497 termes distincts. Le tableau 5.1 résume les principales caractéristiques de chaque catégorie de ce corpus.

On peut considérer que ce corpus est représentatif car on est dans les mêmes tailles des corpus sur les quels la plupart des travaux ont été réalisés (voir les tableaux 2.10 et 2.11).

Catégorie	Nb docs	Nb termes	Nb termes distincts	Taille moyenne d'un document
Culture	258	44.713	12.521	173
Économie	250	38.766	9.158	155
Générale	255	43.333	12.639	170
Politique	250	34.940	9.720	140
Sociale	258	51.714	11.818	200
Sport	255	40.006	8.641	157
Total	1.526	253.472	36.497	166

TABLE 5.1 – Description du corpus SPA.

5.2 Prétraitements

Rappelons que pour un système de classification, les données doivent être préparées afin d'être utilisées comme entrée d'un système d'apprentissage. Pour la préparation de nos données nous avons effectué l'ensemble de prétraitements suivants :

5.2.1 Normalisation orthographique

En plus des traitements standards qui consistent à éliminer les signes de ponctuation, les chiffres et les caractères non arabes ; les textes en langue arabe doivent subir quelques normalisations orthographiques pour pallier au problème de variation de représentation des caractères arabes. En résumé cette phase consiste à :

- éliminer les diacritiques et le caractère Tatweel "-".
- remplacer (أ ، إ ، آ) par (ا).
- remplacer les lettres (ه, ي) finaux par respectivement (ة, ي).

Cette phase permet de regrouper des mots de la même graphie mais avec des diacritiques différentes dans le même groupe : à titre d'exemple les termes comme (آثار, أثار, إثار) seront regrouper dans le seul terme (اثار).

Cette phase a permis la réduction de la taille du vocabulaire à 33.616 termes soit environ 8% de réduction de la taille initiale du corpus.

5.2.2 Suppression des mots outils

Les mots outils ou mots fonctionnels ne véhiculent aucun sens particulier utile pour la T.C. La liste de mots outils la plus répandue et largement reprise par d'autres travaux dans ce domaine, est celle de Khoja [EK06] renfermant 168 entités qui représentent les prépositions, les pronoms, les adverbes, les particules de conjonction ... etc (cf. [Sec. 2.1.4 1]). Pour notre cas, la liste que nous avons adoptée [1] est présentée dans l'annexe I. Elle contient environs 450

1. La liste est réalisée dans le cadre du projet Ayaspell. Cette liste est téléchargeable depuis http ://arabics-topwords.sourceforge.net/index.php

mots s'étalant par exemple aux pronoms verbaux tels que (آمين , سرعان , دونك , رويدك , شتان), quelques verbes particuliers tels que (نعم , حبذا , ساءما) ....

Cette phase a permis de diminuer considérablement la taille des documents textes puisqu'elle permet d'éliminer les termes très fréquents et présents partout.

5.2.3 Stemming

Dans cette étude nous avons opté pour une approche légère de stemming comme nous l'avons expliqué dans la phase de prétraitement dans la description de l'esquisse de notre solution. Cette approche se traduit par l'emploi d'un *light stemming* de l'ensemble des termes. Cette technique est d'une implémentation aisée par rapport à l'approche de lemmatisation (voir discussion menée dans la section 4.6). La liste des préfixes et des suffixes à éliminer peut être établie selon des statistiques réalisées sur le corpus comme elle peut être préétablie à l'avance. Dans notre contexte, nous avons adopté la liste de Larkey [LBC02] (cf. [Sec. 2.2.1]) qui est largement reprise par plusieurs travaux. Pour son implémentation, nous nous sommes inspirés de l'algorithme donné par *Khoribi* [EKI06] dont voici la description :

Input: T un terme à stemmer

R $\leftarrow$ {ا, ب, ت, ف, ك, و, ي} ; $D_1 \leftarrow$ {فال, بال} ; $D_2 \leftarrow$ {فبال, فكال} ; $D_3 \leftarrow$ {الل} ;

LargeSuffix $\leftarrow$ {هما, كما} ; Suffix $\leftarrow$ {ها, يت, هم, نا, وا, يا, كي, هن, كم, تم, ين, ان, , ات, ون} ;

if $Taille[T] \geq 4$ *ET* $T[0] = T[1] =$ "ل" **then**
  Enlever $T[0], T[1]$;
else
  if $Taille[T] \geq 5$ *ET* $T[1] = T[2] =$ "ل" **then**
    Enlever $T[0], T[1], T[2]$;
  else
    if $Taille[T] \geq 3$ *ET* $T[0] = T[1]$ *ET* $T[0] \in R$ **then**
      Enlever $T[0]$;
    else
      if $Taille[T] \geq 4$ *ET* $T[1] = T[2]$ *ET* $T[1] \in R$ **then**
        Enlever $T[0], T[1]$;
      else
        if $Taille[T] \geq 3$ *ET* $T[0] \in$ {"ف","و"} **then**
          Enlever $T[0]$;
        end
      end
    end
  end
end
if $Taille[T] \geq 4$ *ET* $T[0] =$ "ا" *ET* $T[1] =$ "ل" **then**
  Enlever $T[0], T[1]$;
else
  if $Taille[T] \geq 5$ *ET* $T[0], T[1], T[2] \in D_1$ **then**
    Enlever $T[0], T[1], T[2]$;
  else
    if $Taille[T] \geq 6$ *ET* $T[0], T[1], T[2], T[3] \in D_2$ **then**
      Enlever $T[0], T[1], T[2], T[3]$;
    else
      if $Taille[T] = 6$ *ET* $T[0], T[1], T[2], T[3] \in D_2$ **then**
        Enlever $T[0], T[1], T[2], T[3]$;
      end
    end
  end
end
if $Taille[T] \geq 5$ *ET* $T[N-1], T[N-2]$ *ET* $T[N-3] \in LargeSuffix$ **then**
  Enlever $T[N-1], T[N-2], T[N-3]$;
else
  if $Taille[T] \geq 4$ *ET* $T[N-1], T[N-2] \in Suffix$ **then**
    Enlever $T[N-1], T[N-2]$;
  end
end
return T

Algorithm 1: Light-stemming.

La figure 5.1 donne un exemple d'un document dans sa forme initiale et sa forme pré-traitée (normalisation, suppression des mots outils, light-stemming)

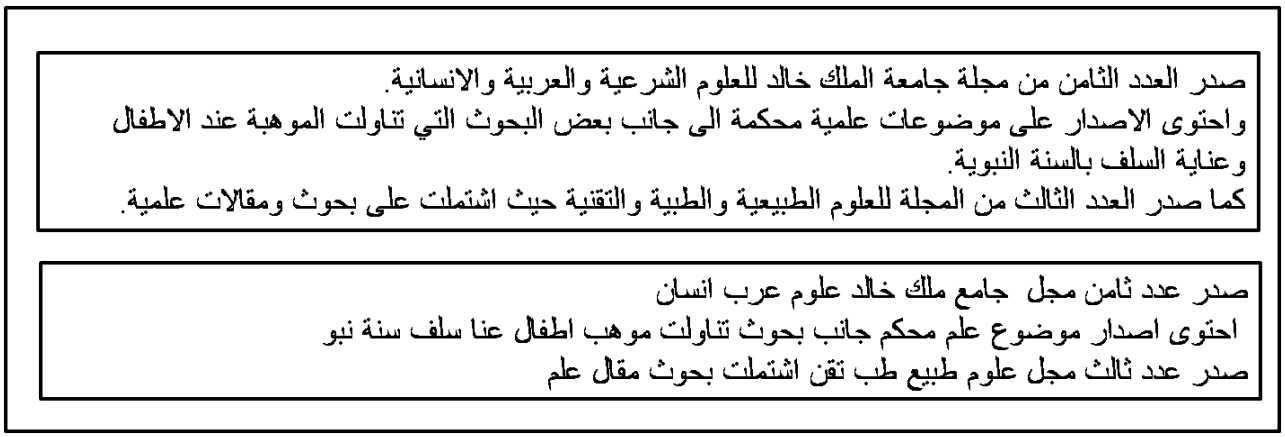

FIGURE 5.1 – Exemple d'un texte avant et après son prétraitement.

Le procédé du *light-stemming* se base aussi sur d'autres règles comme la longueur du terme : par exemple la troncature d'un préfixe à trois lettres d'un mot de longueur quatre n'est pas faisable.

L'étape globale de prétraitement (normalisation, suppression des mots outils, light-stemming) a permis de réduire la taille globale du vocabulaire à 16.242 termes distincts représentant ainsi environ 45% de la taille initiale du corpus.

5.3 Représentation

Nous avons adopté une représentation des textes en transducteurs. Une telle représentation permet de prendre en compte l'ordre et la cooccurrence des termes, ce qui est difficilement réalisable par le modèle VSM (cf. [Sec. 1.2.3]). Cette représentation est réalisé par la bibliothèque *OpenFST* comme nous l'avons expliqué dans l'esquise de notre solution (cf. sec 4.6)

La figure 5.2 donne une partie du transducteur représentant le texte de l'exemple précédent.

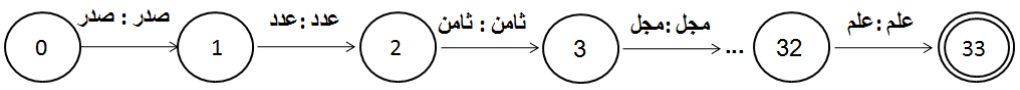

FIGURE 5.2 – Le transducteur associé à la première phrase du corpus SPA.

5.4 Apprentissage

Nous avons utilisé 80% du corpus comme échantillon d'apprentissage et le reste comme échantillon de test pour la validation. Le tableau 5.2 montre le nombre de documents utilisé pour l'apprentissage et celui utilisé pour la validation pour chaque catégorie.

Catégorie	Nb docs. d'apprentissage	Nb docs. test	Total
Culture	200	57	257
Économie	201	50	251
Générale	203	55	258
Politique	200	50	205
Sociale	205	50	255
Sport	205	50	255
Total	1.214	312	1.526

TABLE 5.2 – Répartition du corpus SPA pour l'apprentissage et le test.

On rappel que l'apprentissage est réalisé par le plugin *LIBSVM* incorporé par *Openkernel* qui est une bibliothèque open source pour la manipulation des noyaux rationnels (voir détails dans 4.6). À l'aide de cet outil, nous avons implémenté le noyau N-grammes (P-spectre) et Gappy N-grammes pour une séquence constituée de 2 à 5 termes. Le dernier noyau a été testé avec différents paramètres pour $\lambda = 0.1, \ldots, 0.9, 1$.

5.5 Évaluation

La phase d'évaluation consiste à présenter au modèle construit dans l'étape précédente l'ensemble des documents réservés pour la validation (le 1/5 restant de chaque catégorie). L'évaluation de notre système est faite sur la base des mesures classiques : *Exactitude*, *Précision*, *Rappel* et *F1*.

5.6 Résultats et discussion

Sac de mots

Le tableau 5.3 montre les résultats obtenus par l'approche *sac de mots* pour l'ensemble des catégories. Rappelons que cette approche n'est rien d'autre que le noyau 1-gramme.

Catégorie	Exactitude (%)	Précision (%)	Rappel (%)	F1 (%)
Culture	94.64	93.48	75.44	83.50
Économie	95.58	95.00	76.00	84.44
Générale	84.54	68.75	20.00	30.99
Politique	92.74	86.49	64.00	73.56
Sociale	90.85	86.11	56.36	68.13
Sport	98.42	97.87	92.00	94.85
Moyenne	**92.80**	**87.95**	**63.97**	**74.07**

TABLE 5.3 – Résultats de classification avec l'approche "*sac de mots*".

Premièrement, en vue des résultats moyens obtenus par cette approche par rapport aux résultats réalisés par des classifieurs équivalents (cf. [Sec. 2.3.2]) et soulignant qu'une comparaison entre notre classifieur et ceux mentionnés dans les travaux référencés ne peut être faite (conditions et corpus différents) nous pouvons toute fois, essayer de justifier cela par l'approche de validation "*set validation*" que nous avons adoptée qui est bien inférieure aux autres techniques de validation (*validation croisé* et *leave-one*) et aussi à la réduction de dimension qui améliore les performances de la T.C (cf. [Sec. 1.2.4 4] et [Sec. 2.2.2]) qui n'a pas été utilisée dans notre cas (pour conserver les mêmes conditions que celles appliquées dans la deuxième expérimentation où la réduction de dimension n'a aucun sens).

Pour une lecture primaire des résultats du tableau 5.3 on constate que la catégorie "Sport"', comme il est de coutume dans la T.C, réalise les meilleures performances en termes de toutes les mesures. Cela est dû à un ensemble de termes très spécifiques à cette catégorie (منتخب , كرة , قدم , رياضة, شباب ...) dont leurs fréquences sont très élevées (de l'ordre de 300 et supérieur) ; ce qui explique, dans l'autre sens, les performances faibles de la catégorie "Générale"' ; car celle-ci contient plus de termes distincts (6666 termes soit presque 140 % par rapport à la taille de la catégorie "Sport") dont les plus fréquents sont partagés avec plusieurs catégories.

Le tableau 5.4 montre les dix premiers termes pour la catégorie "Générale" et leurs fré-

quences par rapport aux autres catégories.

Termes	Catégorie	Rang	Fréquence	catégorie	Rang	Fréquence
بن	Générale	(1)	385	Culture	(1)	353
				Sociale	(1)	585
				Économie	(1)	353
عام	Générale	(2)	332	Économie	(4)	237
يوم	Générale	(3)	292	Sociale	(2)	306
سمو	Générale	(4)	240	Culture	(5)	400
رءيس	Générale	(5)	233	Politique	(1)	344
دول	Générale	(6)	190	Culture	(3)	272
امير	Générale	(7)	166	Sociale	(6)	362
منطقة	Générale	(8)	163	Sociale	(2)	408
ملك	Générale	(9)	153	Sociale	(12)	214
عبدالعزيز	Générale	(10)	153	Économie	(10)	184

TABLE 5.4 – Correspodance entre les premiers termes de la catégorie "Générale" et les catégories les plus proches.

Séquence de mots

Le but de la mise en œuvre d'une telle approche est généralement de mettre en évidence un ensemble de termes composés qui peuvent caractériser une catégorie mieux qu'un ensemble de termes pris séparément qui peuvent être d'un emploi très général et partagé par l'ensemble des catégories. Le tableau 5.5 montre quelques termes composés caractérisants et leurs catégories.

Catégorie	Termes composés
Culture	محاضرة عنوان , حرم شريف , عربية سعودية , مكة مكرمة
Économie	مجلس ادارة , داو جونس , ارتفع موءشر , صناعة تجارة
Générale	رءيس مجلس , مدير عام , صاحب سمو , بن عبدالعزيز
Politique	شرق اوسط , رءيس وزراء , امّم متحدة , دول عربية , وزير خارجية
Sociale	عيد فطر , , توزيع جواءيز , امير منطقة , بن عبدالعزيز
Sport	رءيس الاتحاد ,رعاية الشباب , شباب رياضة , كرة قدم

TABLE 5.5 – Exemple de quelques termes composés importants et leurs catégories.

Le tableau 5.6 montre la moyenne des mesures obtenues pour le noyau N-grammes de termes avec N variant de 2 à 5 pour toutes les catégories.

La première remarque qu'on peut constater depuis le tableau 5.6 c'est que notre approche donne de bonnes résultats en termes d'*Exactitude* et *Précision*. La *Précision* atteint le maximum pour une valeur moyenne (3 ou 4) de mots consécutifs. Quant aux *Rappel* et *Exactitude*, elles atteignent leurs maximum pour 2 termes consécutifs (N-grammes = 2).

N-grammes	Exactitude (%)	Précision (%)	Rappel (%)	F1 (%)
2-grammes	86.75	83.15	23.95	37,19
3-grammes	85.12	89.37	12.02	21,19
4-grammes	84.86	88.50	10.17	18,24
5-grammes	84.65	86.11	8.87	16,08

TABLE 5.6 – Résultats du noyau N-grammes.

On peut dire que le noyau N-grammes de termes consécutifs est capable de capturer la similarité entre deux textes pour un motif (taille de fenêtre) adéquat. En contre partie, le N-grammes diminue le *Rappel* et l'*Exactitude*. Pour le *Rappel* on peut dire qu'une séquence de termes trop longue ne permet pas au système de construire des règles flexibles pour en décider de l'appartenance d'un document texte à une catégorie donnée.

Pour une meilleure compréhension de ce résultat, nous avons recours à l'analyse des fréquences des termes composés i.e l'analyse de la cooccurence. Nous avons choisi de le faire pour les 2-grammes. Dans ce cas, on constate que le vocabulaire des séquences des termes composés a augmenté considérablement. À titre d'exemple, on est passé de l'ordre de 6.666 termes uniques à 27.191 termes composés pour la catégorie "Culture" et au même temps leurs fréquences ont diminué : 26.825 termes composés ont une fréquence inférieure à 5 contre uniquement 19 termes avec une fréquence supérieure à 30. Les termes composés qui ont un pouvoir discriminant pour une catégorie donnée vont être "noyés" dans un ensemble de termes composés très élevé.

Maintenant pour voir l'impact de la contiguïté des termes sur les performances générales d'un système de classification, nous avons conduit une autre expérimentation. Cette dernière consiste, pour différentes longueur de séquences, à expérimenter différentes valeurs de λ. Cependant pour des considération pratique, *Openkernel* ne permet pas de calculer le noyau *Gappy n-grammes* pour des séquences de taille quelconque ; donc il introduit un autre paramètre (*Gap*) pour se limiter à une longueur donnée d'une séquence : par exemple, pour K=2 i.e 2-grammes et un *Gap* de deux, l'évaluation se contente des sous-séquences composées de deux termes mais sur des séquences étalées sur quatre termes maximum.

Dans notre cas, nous avons évalué un *Gap* variant de 1 à 5 en se basant sur l'idée que des termes composés étalés sur des séquences trop longues sont considérés comme des termes non liés.

La figure 5.3 illustre, pour le 2-grammes, les résultats (en *F1*) pour ces différents paramètres.

On constate que pour des valeurs importantes du facteur λ ; elles influencent négativement les performances : la courbe décroit après certaines valeurs de λ (le *Pic* est atteint généralement pour $\lambda = 0.3$ quelque soit le *Gap* considéré). On remarque aussi que pour des valeurs importante du *Gap* (> 2) le F1 diminue considérablement. Concernant les conclusions tirées sur les meilleures performances réalisées par ce noyau ; on constate qu'elles sont identiques à

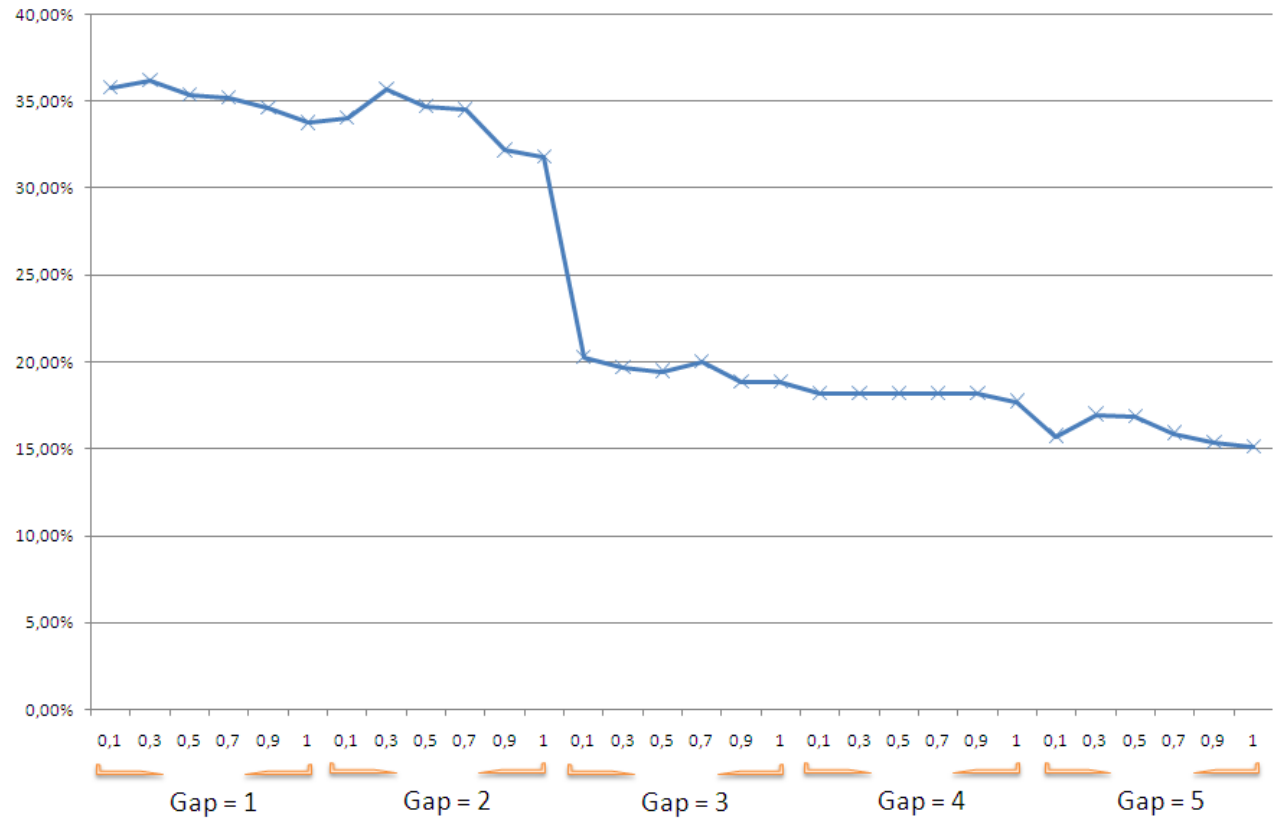

FIGURE 5.3 – Résultats du noyau *Gappy N-grammes* en *F1* pour le 2-grammes.

celles du noyau N-grammes i.e la *Précision* atteint le maximum pour une valeur de K égale à 3 ou 4 ; ainsi que pour le *Rappel* et l'*Exactitude*, qui atteignent leurs maximum pour $K = 2$. Le tableau 5.7 montre les résultats de toutes les mesures pour les meilleures valeurs ($\lambda = 0.3$ et *Gap* variant de 1 à 2).

K	Gap	Exactitude (%)	Précision (%)	Rappel (%)	F1 (%)
2	1	86,59	80,80	23,30	36,17
	2	86,38	80,56	23,12	35,93
3	1	84,91	86,90	11,78	20,73
	2	84,96	86,90	11,07	19,64
4	1	84,25	87,15	10,12	18,13
	2	84,14	86,92	10,04	18,00
5	1	83,88	85,44	07,26	13,38
	2	83,65	85,28	07,13	13,16

TABLE 5.7 – Résultats du noyau *Gappy N-grammes* pour avec les meilleurs paramètres.

On peut conclure qu'un système de classification basé sur les séquences de termes peut améliorer la *Précision*. Dans ce contexte, l'ignorance de l'écart entre les mots est un choix à retenir du moment que les résultats du noyau *Gappy N-grammes* avec les meilleures valeurs des paramètres λ et *Gap* (le tableau 5.7 comparé au tableau 5.6) sont généralement moins performants que les résultats réalisés par le noyau *N-grammes* de termes consécutifs sans écart.

Pour essayer d'expliquer cela, une étude statistique restreinte est menée sur quelques termes composés. Elle montre que l'introduction d'un terme (un adjectif par exemple) se fait généralement après l'expression des termes composés : des séquences comme "الرءيس الحالي للوزراء", "الرءيس السابق للوزراء" ou "الرءيس الاسبق للوزراء" sont très rares ; on emploi plutôt "رءيس الوزراء الحالي", ...

Finalement, les performances obtenues par l'approche *sac de mots* sont généralement meilleures par rapport à l'approche basée sur des séquences de termes. Néanmoins pour un système où la *Précision* est le paramètre dominant, le recours à la deuxième approche est une solution à retenir.

5.7 Conclusion

Le but de notre démarche est d'évaluer l'impact d'une représentation basée sur des séquences de termes sur les performances générales d'un système de classification des textes en langue arabe. La mise en œuvre de cette approche est difficilement réalisable voir impossible sur une représentation de l'ensemble des termes en vecteurs. Pour cela nous avons opté pour une représentation de l'ensemble des documents textes en transducteurs. Cette manière, nous a permis de combiner les *SVM* comme méthode de classification avec différents noyaux appropriés. Le noyaux *Gappy N-grammes* permet de prendre en compte une séquence de termes avec un paramètre d'écart.

Les résultats ont montré qu'une taille de fenêtre adéquate (3 à 4 termes) impacte positivement la *Précision*. Quant au rôle du paramètre d'écart, il est moins intéressant dans ce contexte. En effet, les performances du système diminuent en terme de toutes les mesures.

Conclusion générale et perspectives

Le développement des méthodes à noyau et particulièrement des Séparateurs à Vaste Marge (*SVM*) a marqué le point de convergence de plusieurs concepts essentiels. Parmi ces concepts on trouve le passage au non linéaire, grâce à l'astuce noyau, d'une grande famille d'algorithmes linéaires ne s'appuyant que sur des produits scalaires dans l'espace des entrées. En outre, le deuxième concept essentiel de ces méthodes est l'encapsulation des données par rapport à l'algorithme solveur, permettant ainsi l'utilisation des algorithmes génériques sur différents types de données et pour plusieurs tâches.

Ce deuxième concept est l'innovation majeure de ces algorithmes car il permet d'étendre ces méthodes à des espaces non vectoriels par la définition de similarité via des fonctions noyaux adéquates. Le seul problème est de savoir construire des fonctions noyaux calculables pour des structures de données complexes non vectorielles tels que les textes, les arbres, les graphes ...

Motivé par une telle approche, l'ingénierie des fonctions noyaux a connu une immense publication consacrée à la définition de nouveaux noyaux appropriés au problème d'induction et à la nature des données sous-jacentes. Dans ce contexte, plusieurs noyaux adaptés aux chaînes et séquences telles que les noyaux *P-spectre*, *toutes sous-séquences*, *sous-séquence de longueur fixe*, *sous-séquence avec pénalité sur l'écart* ont vu le jour. La plupart de ces noyaux ont été testés dans les domaines de la bio-informatique et la classification de texte.

Le travail présenté dans ce mémoire se situe dans ce contexte où nous sommes intéressés à l'apprentissage sur des données non structurées en l'occurrence les textes en langue arabe. Pour parvenir à cet objectif, nous avons opté pour les *SVM* comme méthode d'apprentissage. Cette méthode a connu ses premières applications dans le domaine de la classification de textes où elle a réalisé des performances inégalées faisant ainsi de cette méthode une référence pour ce type d'application. Ces résultats intéressants trouvent leurs justification dans les fondements théoriques solides notamment la capacité des *SVM* à faire face à la grande dimensionnalité de l'espace de représentation des données ainsi que leurs capacité de généralisation.

La classification de textes en langue arabe n'a pas bénéficié d'une telle puissance théorique et empirique, comme le nombre de travaux en témoigne. Pour cette langue, la mise en œuvre d'un système de classification de textes rencontre des difficultés supplémentaires. Ces difficultés

sont liées à la morphologie complexes, à la richesse flexionnelle et au phénomène d'agglutination qui caractérisent cette langue. Toutefois et malgré le peu de travaux enregistrés dans ce domaine et particulièrement ceux à base des *SVM* ; les résultats rapportent de bonne performances de ces derniers comparées aux ceux réalisées par les algorithmes classiques (*K-ppv*, *N.B*, *Arbres de décision*,...).

Depuis la première étude de *Sawaf et al.* en 2001, plusieurs approches ont été adoptées dans le cadre de la classification de textes en langue arabe. Ces approches se diversifient selon la composante élémentaire adoptée (terme, caractère) ou le modèle retenu pour la représentation des textes (Modèle vectoriel, Modèle de Markov caché) ainsi que les différents procédés de normalisation linguistique appliqués (light-stemming, extraction de racine par méthodes algorithmiques ou statistiques). La plupart de ces travaux ont été testés sur des corpus d'une taille moyenne (nombre de documents et nombre de catégories), mise à part la première étude évoquée où elle a été testée sur le corpus standard *NewsWire* de LDC d'une taille importante. Aussi, il est à noter que la plupart de ces travaux et notamment ceux à base des *SVM*, adoptaient une représentation vectoriel. De plus, mise à part des noyaux usuels, les autres noyaux notamment ceux appropriés pour les données textuelles n'ont fait l'objet d'une attention particulière dans les démarches entreprise. Encore, on peut évoquer l'absence d'une étude comparative qui s'intéresse à l'effet de la prise en compte des séquences de termes au lieu des termes pris séparément sur les performances d'un système de classification de textes.

Dans notre travail on est resté dans les mêmes conditions concernant la taille des corpus de test où nous avons testé la démarche entreprise sur le corpus SPA qui est d'une taille supérieure à la plupart des corpus de ces travaux. Cependant, notre démarche est radicalement différente. Premièrement nous avons adopté les transducteurs comme modèle de représentation des textes ce qui nous a permis de prendre les aspects, jusqu'alors négligés, de l'ordre et les cooccurrences des termes dans les documents textes. Les transducteurs offrent l'avantage d'une représentation compacte des données textuelles, comme ils définissent une classe de noyaux appelés noyaux rationnels dont les noyaux de séquences constituent un exemple de cette classe. Les noyaux rationnels présentent l'intérêt d'être calculés efficacement en utilisant les opérations définies sur les transducteurs. Basé sur cette représentation, l'évaluation de plusieurs variantes de N-grammes de caractères ou de termes est possible sans avoir à redéfinir le module de construction de cette représentation puisque la représentation est la même ; uniquement les paramètres du noyau N-grammes (N variant de 2 à 5) doivent être redéfinis.

L'étude empirique nous a permis d'évaluer l'impact de la prise en compte des séquences de termes sur les performances d'un système de classification basé sur ce type de composante par rapport à un système dont la composante de base est le terme tout court. Les résultats ont montré une légère amélioration dans la *Précision* mais au détriment du *Rappel*. Une séquence de longueur moyenne (3 à 4 termes) permet d'atteindre les meilleures performances. D'un autre coté l'introduction du paramètre d'écart dans les séquences n'améliore aucunement

les performances.

Comme perspective, nous projetons de réaliser une étude sur le noyau Gappy N-grammes de caractère pour voir l'effet d'introduction du paramètre d'écart sur les N-grammes de caractère. Cette mise en œuvre nécessitera le recours à des techniques d'optimisation puisque ce noyau a besoin de ressources assez importantes notamment pour une représentation basée sur les transducteurs où chaque caractère représente une transition. En même temps, cela nous permettra d'évaluer d'autres choix que le 3-grammes pris comme valeur par défaut dans tous les travaux de la T.C en langue arabe.

Bibliographie

[AAA+08] A Althubaity, A Almuhareb, S Alharbi, A Al-Rajeh, and M Khorsheed. KACST Arabic Text Classification Project : Overview and Preliminary Results. 2008.

[Abd04] Ahmed Abdelali. *Improving Arabic Information Retrieval Using Local Variations in Modern Standard Arabic*. PhD thesis, Department of Computer Science. New Mexico Institute of Mining and Technology., 2004.

[ACS05] A. Abdelali, J. Cowie, and H. Soliman. Building A Modern Standard Arabic Corpus. *Workshop on Computational Modeling of Lexical Acquisition. The Split Meeting. Split, 25th to 28th of July*, 2005.

[AE99] Kjersti Aas and Line Eikvil. Text Categorisation : A Survey, 1999.

[AES10] Abdelmalek Amine, Zakaria Elberrichi, and Michel Simonet. Evaluation of Text Clustering Methods Using WordNet. *International Arab Journal of Information Technology*, 7(4) :351, 2010.

[AHAAT+08] S Al-Harbi, A Almuhareb, A Al-Thubaity, M. S. Khorsheed, and A Al-Rajeh. Automatic Arabic Text Classification. In *JADT 2008 : Proceedings of The 9th International Conference on the Statistical Analysis of Textual Data*, 2008.

[AKCS00] Ion Androutsopoulos, John Koutsias, Konstantinos Chandrinos, and Constantine D. Spyropoulos. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *SIGIR*, pages 160–167, 2000.

[AKE94] Ibrahim A. Al-Kharashi and Martha W. Evens. Comparing Words, Stems, and Roots as Index Terms in an Arabic Information Retrieval System. *JASIS*, 45(8) :548–560, 1994.

[AKS03] R. Alshalabi, G. Kanaan, and H. Serhan. New Approach for extracting Arabic roots. In *2003 Arab conference on Information Technology (ACIT 2003), Alexandria, Egypt*, pages 42–59, 2003.

[Als11] S. Alsaleem. Automated Arabic Text Categorization Using SVM and NB. In *International Arab Journal of e-Technology*, number 2, 2011.

[Aro50] Nachman Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc*, 68(3) :337–404, 1950.

[ARS+07] Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. OpenFst : A general and efficient weighted finite-state transducer library. In *Implementation and Application of Automata*, pages 11–23. Springer, 2007.

[Ber79] Jean Berstel. *Transductions and Context-Free Languages*. Teubner Studienbücher, Stuttgart, 1979.

[BGV92] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A Training Algorithm for Optimal Margin Classifiers. In David Haussler, editor, *COLT*, pages 144–152. ACM, 1992.

[BSAS94] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic Query Expansion Using SMART : TREC 3. In *TREC*, 1994.

[Buc04] Tim Buckwalter. Issues in Arabic orthography and morphology analysis. In *Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages*, Semitic '04, pages 31–34, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[Cav94] William B. Cavnar. Using An N-Gram-Based Document Representation With A Vector Processing Retrieval Model. In *TREC*, 1994.

[CG02] Aitao Chen and Fredric C. Gey. Building an Arabic Stemmer for Information Retrieval. In *TREC*, 2002.

[CH67] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1) :21–27, 1967.

[CHM03] Corinna Cortes, Patrick Haffner, and Mehryar Mohri. Weighted Automata Kernels–General Framework and Algorithms. *a : a*, 1 :1, 2003.

[CHM+04] Corinna Cortes, Patrick Haffner, Mehryar Mohri, Kristin Bennett, and Nicolò Cesa-bianchi. Rational kernels : Theory and algorithms. *Journal of Machine Learning Research*, 5 :1035–1062, 2004.

[CKEST01] Nello Cristianini, Jaz Kandola, Andre Elisseeff, and John Shawe-Taylor. On optimizing kernel alignment, 2001.

[CL11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3) :27, 2011.

[CM02] Antoine Cornuéjols and Laurent Miclet. *Apprentissage artificiel - Concepts et algorithmes*. Eyrolles, 2002.

[CM09] Corinna Cortes and Mehryar Mohri. Learning with Weighted Transducers. In *Proceedings of the 2009 conference on Finite-State Methods and Natural Language Processing : Post-proceedings of the 7th International Workshop FSMNLP 2008*, pages 14–22, Amsterdam, The Netherlands, 2009. IOS Press.

[CS96] William W. Cohen and Yoram Singer. Context-sensitive Learning Methods for Text Categorization. In *SIGIR*, pages 307–315, 1996.

[CST10] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2010.

[Dah87] A. Dahdah. *Dictionnaire des termes de déclinaison et de structure en grammaire arabe universelle : (arabe-français)(français-arabe)*. Librairie du Liban, 2eme edition, 1987.

[Dar02] Kareem Darwish. Building a shallow Arabic morphological analyzer in one day. In *40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 1–8, 2002.

[DDJ+01] Kareem Darwish, David S. Doermann, Ryan C. Jones, Douglas W. Oard, and Mika Rautiainen. TREC-10 Experiments at University of Maryland CLIR and Video. In *TREC*, pages 549–562, 2001.

[DDL+90] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by Latent Semantic Analysis. *JASIS*, 41(6) :391–407, 1990.

[DMS+11] G. Dreyfus, J.M. Martinez, M. Samuelides, M.B. Gordon, F. Badran, and S. Thiria. *Apprentissage statistique : Réseaux de neurones - Cartes topologiques - Machines à vecteurs supports*. Eyrolles, 2011.

[DPHS98] Susan T. Dumais, John C. Platt, David Hecherman, and Mehran Sahami. Inductive Learning Algorithms and Representations for Text Categorization. In *CIKM*, pages 148–155, 1998.

[Duw07] Rehab M. Duwairi. Arabic Text Categorization. *Int. Arab J. Inf. Technol.*, 4(2) :125–132, 2007.

[EBR04a] Mohamed Elkourdi, Ahmed Bensaid, and Tajjeeddine Rachidi. Automatic Arabic Document Categorization Based on the Naïve Bayes Algorithm. In *COLING 20th Workshop on Computational Approaches to Arabic Script-based Languages*, pages 51–58, Geneva, 2004.

[EBR+04b] Mohamed Elkourdi, Ahmed Bensaid, Tajjeeddine Rachidi, Abdellah Chekayri, and Mohamed Mhamdi. A Concatenative Aproach to Non-Vocalized Arabic Root Extraction. In *the Sixth Conference on Language Engineering*, pages 51–58, Cairo, 2004.

[EH08] Alaa El-Halees. A Comparative Study on Arabic Text Classification. *Egyptian Computer Science Journal*, 30(2), 2008.

[Ek05] Dina El-kassas. *une étude contrastive de l'arabe et du français dans une perspective de génération multilingue*. PhD thesis, Université Paris 7, UFR Linguistique Denis Diderot, Paris, France, 2005.

[EK06] Ibrahim Abu El-Khair. Effects of stop words elimination for Arabic information retrieval : a comparative study. *International Journal of Computing & Information Sciences*, 4(3) :119–133, 2006.

[EKI06] R El-Khoribi and M Ismael. An intelligent system based on statistical learning for searching in Arabic text. *ICGST International Journal on Artificial Intelligence and Machine Learning*, 2006.

[ES06] A. Esuli and F. Sebastiani. SentiWord Net : A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation*, IT'06, pages 417–422, 2006.

[FB91] N. Fuhr and C. Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3) :223–248, 1991.

[Gär03] Thomas Gärtner. A survey of kernels for structured data. *ACM SIGKDD Explorations Newsletter*, 5(1) :49–58, 2003.

[GHF09] Tarek F. Gharib, Mena B. Habib, and Zaki T. Fayed. Arabic Text Classification Using Support Vector Machines. *International Journal of Computers and Their Applications*, 16(4) :192–199, 2009.

[GJZ03] E. Gaussier, C. Jacquemin, and P. Zweigenbaum. In E. Gaussier and M.-H. Stéfanini editors, Traitement automatique des langues et recherche d'information. In *Assistance Intelligente à la Recherche d'Information*, pages 71–96, 2003.

[Gué06] Sébastien Guérif. *Réduction de dimension en Apprentissage Numérique Non Supervisé*. PhD thesis, Université Paris, 2006.

[GY11] Éric Gaussier and François Yvon. *Modèles statistiques pour l'accès à l'information textuelle*. Hermès / Lavoisier, 2011. Collection : Recherche d'information et Web.

[Hab08] Mena Badieh Habib. An intelligent system for automated Arabic Text Categorization, 2008.

[HIMM02] T Hirao, H Isozaki, E Maeda, and Y Matsumoto. Extracting Important Sentences with Support Vector Machines. In *COLING*, pages –, 2002.

[Hof00] Thomas Hofmann. Learning the similarity of documents : An information-geometric approach to document retrieval and categorization. *Advances in neural information processing systems*, 12 :914–920, 2000.

[Ish07] Anis Ben Ishak. *Sélection de variables par les machines à vecteurs supports pour la discrimination binaire et multiclasse en grande dimension*. PhD thesis, 2007.

[Jai04] Simon Jaillet. Catégorisation automatique de documents. Technical report, LIRMM, Montpellier, LIRMM UMR 5506, France, 2004.

[Jal03] Radwan Jalam. *Apprentissage automatique et catégorisation de textes multilingues*. PhD thesis, Université lumière Lyon 2, 2003.

[Joa98] Thorsten Joachims. Text Categorization with Suport Vector Machines : Learning with Many Relevant Features. In *ECML*, pages 137–142, 1998.

[Kad04] Youcef Kadri. Traduction des requêtes pour la recherche d'information translinguistique Anglais-Arabe. In *la conférence sur le Traitement Automatique des Langues Naturelles (TALN)*, pages 291–296, 2004.

[Kan05] *Automatic Text Classification using naïve Bayesian Algorithm on Arabic language*, 2005.

[KG99] S. Khoja and R. Garside. Stemming Arabic text. Technical report, Computing Department, Lancaster University, Lancaster, 1999.

[Khr06] Laila Khreisat. Arabic Text Classification Using N-Gram Frequency Statistics A Comparative Study. In *DMIN*, pages 78–82, 2006.

[KS96] Daphne Koller and Mehran Sahami. Toward Optimal Feature Selection. In *ICML*, pages 284–292, 1996.

[KWA97] Jyrki Kivinen, Manfred K Warmuth, and Peter Auer. The Perceptron algorithm versus Winnow : linear versus logarithmic mistake bounds when few input variables are relevant. *Artificial Intelligence*, 97(1) :325–343, 1997.

[LBC02] Leah S. Larkey, Lisa Ballesteros, and Margaret E. Connell. Improving stemming for Arabic information retrieval : light stemming and co-occurrence analysis. In *SIGIR*, pages 275–282, 2002.

[LC01] Leah S. Larkey and Margaret E. Connell. Arabic information retrieval at UMass. In *TREC*, 2001.

[LEC+04] Christina Leslie, Eleazar Eskin, Adiel Cohen, Jason Weston, and William Stafford Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20 :467–476, 2004.

[LEN02] Christina Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel : a string kernel for SVM protein classification. *Pacific Symposium On Biocomputing*, 575(50) :564–575, 2002.

[Lew92] David D. Lewis. An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task. In *SIGIR*, pages 37–50, 1992.

[LK02] Edda Leopold and Jörg Kindermann. Text Categorization with Support Vector Machines. How to Represent Texts in Input Space ? *Machine Learning*, 46(1-3) :423–444, 2002.

[LK03] Christina Leslie and Rui Kuang. Fast kernels for inexact string matching. *Learning Theory and Kernel Machines*, pages 114–128, 2003.

[LPY94] Elizabeth D. Liddy, Woojin Paik, and Edmund S. Yu. Text Categorization for Multiple Users Based on Semantic Features from a Machine-Readable Dictionary. *ACM Trans. Inf. Syst.*, 12(3) :278–295, 1994.

[LSST+02] Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *J. Mach. Learn. Res.*, 2 :419–444, 2002.

[Mer09] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Royal Soc. (A)*, 83(559) :69–70, 1909.

[Mes07] Abdelwadood Mohd A. Mesleh. Chi Square Feature Extraction Based Svms Arabic Text Categorization System. In *ICSOFT*, pages 235–240, 2007.

[Mit97] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[MLS99] D. H. Miller, T. Leek, and R. Schwartz. A hidden Markov model information retrieval system. In *SIGIR'99*, pages 214–221, 1999.

[MMC+01] James Mayfield, Paul McNamee, Cash Costello, Christine D. Piatko, and Amit Banerjee. JHU/APL at TREC 2001 : Experiments in Filtering and in Arabic, Video, and Web Retrieval. In *TREC*, 2001.

[MMR+01] Klaus-Robert Muller, Sebastian Mika, Gunnar Ratsch, Koji Tsuda, and Bernhard Scholkopf. An introduction to kernel-based learning algorithms. *IEEE transactions on neural networks*, 12(2) :181–201, 2001.

[MN98] Andrew McCallum and Kamal Nigam. A Comparison of Event Models for Naive Bayes Text Classification. In *Learning for Text Categorization : Papers from the 1998 AAAI Workshop*, pages 41–48, 1998.

[Mot10] Saad Motaz. Arabic Morphological Tools for Text Mining. In *6th International Conference on Electrical and Computer Systems EECS'10*, pages –, Lefke, North Cyprus, 2010.

[MPR00] Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted Finite-State Transducers in Speech Recognition, 2000.

[MRS08] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[Nak07] Didier Nakache. *Extraction automatique des diagnostics à partir des comptes rendus médicaux textuels*. PhD thesis, Thèse de doctorat en informatique, CNAM Paris, 2007.

[NZCG12] Attia Nehar, Djelloul Ziadi, Hadda Cherroun, and Younes Guellouma. An efficient stemming for Arabic Text Classification. In *Innovations in Information Technology (IIT)*, pages 328–332. IEEE, 2012.

[Pla99] John C. Platt. Advances in kernel methods. chapter Fast training of support vector machines using sequential minimal optimization, pages 185–208. MIT Press, Cambridge, MA, USA, 1999.

[Por80] Martin F Porter. An algorithm for suffix stripping. *Program : electronic library and information systems*, 14(3) :130–137, 1980.

[Réh11] S. Réhel. *Catégorisation automatique de textes et cooccurrence de mots : Catégorisation automatique de textes et cooccurrence de mots provenant de documents non étiquetés*. Editions Universitaires Europeennes, 2011.

[Roc71] J. Rocchio. *Relevance feedback in information retrieval*, pages 313–323. Prentice Hall, 1971.

[SA08] Majdi Sawalha and Eric Atwell. Comparative Evaluation of Arabic Language Morphological Analysers and Stemmers. In *COLING (Posters)*, pages 107–110, 2008.

[SA10] Motaz K Saad and Wesam Ashour. OSAC : Open Source Arabic Corpora. In *6th International Conference on Electrical and Computer Systems (EECS10), Lefke, Cyprus.*, 2010.

[Seb99] Fabrizio Sebastiani. A tutorial on automated text categorisation. In *1st Argentinian Symposium on Artificial Intelligence*, pages 7–35, 1999.

[Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1) :1–47, 2002.

[SFH06] M. M. Syiam, Z. T. Fayed, and M. B. Habib. AN INTELLIGENT SYSTEM FOR ARABIC TEXT CATEGORIZATION. In *IJICIS*, volume 6, 2006.

[SG02] Michele Sebag and Patrick Gallinari. Apprentissage artificiel : acquis, limites et enjeux. *Assises*, 2002.

[Sha01] Claude E. Shannon. A mathematical theory of communication. *Mobile Computing and Communications Review*, 5(1) :3–55, 2001.

[SHP95] Hinrich Schütze, David A. Hull, and Jan O. Pedersen. A Comparison of Classifiers and Document Representations for the Routing Problem. In *SIGIR*, pages 229–237, 1995.

[SM84] Gerard Salton and Michael McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, 1984.

[SRZ07] Majed Sanan, M. Rammal, and K. Zreik. L'accès multilingue à l'information scientifique et technologique : limitations des moteurs de recherche en langue arabe. In *10 eme Conférence Internationale sur le Document Électronique (CIDE'07)*, page 100, Nancy, France, 2007.

[STC04] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[Str00] Mathieu Stricker. *Réseaux de neurones pour le traitement automatique du langage : conception et réalisatiion de filters d'informations.* PhD thesis, Laboratoire d'électronique de l'ESPCI (Paris), l'Université Pierre et Marie Curie -Paris VI, 2000.

[Suj07] Aseervatham Sujeevan. *Apprentissage à base de Noyaux Sémantiques pour le Traitement de Données Textuelles.* PhD thesis, Université Paris-Nord-Paris XIII, 2007.

[SWDH09] Dina A Said, Nayer M Wanas, Nevin M Darwish, and N Hegazy. A study of text preprocessing tools for Arabic Text Categorization. In *The Second International Conference on Arabic Language*, pages 230–236, 2009.

[SZN01] H. Sawaf, J. Zaplo, and H. Ney. Statistical Classification Methods for Arabic News Articles. In *Arabic Natural Language Processing Workshop (ACL2001),Toulouse, France.*, 2001.

[TAS90] Murat Tayli and Abdulah I. Al-Salamah. Building Bilingual Microcomputer Systems. *Commun. ACM*, 33(5) :495–504, 1990.

[TEZH09] Fadi Thabtah, Mohammad Ali H. Eljinini, Mannam Zamzeer, and Waeel Musa Hadi. Naïve Bayesian Based on Chi Square to Categorize Arabic Data. *Communications of the IBIMA*, 10(2) :158–163, 2009.

[TI95] Makoto Tokunaga and Takenobu Iwayama. Cluster-based text categorization : a comparison of category search strategies. In *ACM SIGIR conference on research and development in information retrieval*, page 273, 1995.

[Val84] Leslie G. Valiant. A Theory of the Learnable. *Commun. ACM*, 27(11) :1134–1142, 1984.

[Vap95] Vladimir N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[Vap98] Vladimir Vapnik. *Statistical learning theory.* Wiley, 1998.

[VF99] Emmanuel Viennet and Rodrigo Fernandez. Machines à vecteurs de support et réseaux de neurones : comparaisons expérimentales pour l'identification de visages. *Proceedings CAP*, 99, 1999.

[VS03] S. V. N. Vishwanathan and Alex Smola. Fast Kernels for String and Tree Matching. *Advances in Neural Information Processing Systems*, 15, 2003.

[XFW02] Jinxi Xu, Alexander Fraser, and Ralph Weischedel. Empirical Studies in Strategies for Arabic Retrieval. *SIGIR 02 Tampere, Finland*, pages 269–274, 2002.

[YL99] Yiming Yang and Xin Liu. A Re-Examination of Text Categorization Methods. In *SIGIR*, pages 42–49, 1999.

[YP97] Yiming Yang and Jan O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In *ICML*, pages 412–420, 1997.

ANNEXE I : Liste des mots outils

Liste des mots outils

حرف

بيد	لاسيما	إذاً	لن	لو	لولا	لوما	نعم	إن	لات
ما	لا	أَنَّ	إنَّ	علَّ	كَأَنَّ	لَعَلَّ	لَكِنَّ	لَيْتَ	آي
كي	كلا	كلاهما	كليكما	كليهما	إلّا	لكن	فيم	فيما	هل
سوف	هلّا	قد	إمّا	كأنّا	كما	لكي	لكيلا	إلى	رُبَّ
على	عن	في	مِن	عَمَّا	حَتَّى	منذ	مذ	لم	لمّا
أجل	إذن	إي	بلى	جلل	جير	كلَّا	إذما	لئن	أمّا
ألا	أما	أم	أو	بل	ثُمَّ	أيا	هيا	أن	قطّ
لمّا	إنّما	لكنَّما							

ضمير

بك	بكم	بكما	بكن	بنا	به	بها	بي	لك	لكم
لكما	لكن	لنا	له	لها	لي	أنا	أنت	أنتِ	أنتم
أنتما	أنتن	نحن	هم	هما	هن	هو	هي	إياك	إياك
إياكم	إياكما	إياكن	إيانا	إياه	إياها	إياهم	إياهما	إياهن	إياي

اسم

سوى	غير	متى	أنّى	أيّ	أيّان	أين	بكم	بما	بماذا
بمن	كم	كيف	ما	ماذا	متى	مما	ممن	من	أنّى
أيّ	أيّان	أين	أينما	حيثما	كيفما	ما	متى	من	مهما
أولئك	أولئكم	أولاء	أولالك	تانِ	تانِك	تلك	تلكم	تلكما	تِه
تِي	تَيْنِ	تينك	ثمّ	ثمّة	ذا	ذاك	ذانِ	ذانك	ذلك
ذلكم	ذلكما	ذلكن	ذِه	ذوا	ذواتا	ذواتي	ذِي	ذَيْنِ	ذينك
كذلك	هَؤلاء	هَاتانِ	هَاتِه	هَاتِي	هَاتَيْنِ	هاهنا	هَذا	هَذانِ	هَذِه
هَذِي	هَذَيْنِ	هكذا	هنا	هناك	هنالك	أي	إذ	إذا	بعض
تجاه	تلقاء	جميع	حسب	حيث	سبحان	سوى	شبه	كل	لعمر
لما	مثل	مع	معاذ	نحو	أقل	أكثر	الألاء	الألى	التي
الذي	الذين	اللائي	اللاتي	اللتان	اللتيا	اللتين	اللذان	اللذين	اللواتي
أَيُّ	ذا	ذات	ما	مَنْ	أب	أخ	حم	ذو	فو
أجمع	جميع	عامة	عين	كل	كلتا	نفس	دون	ريث	عند
عوض	قبل	كلّما	لدن	لدى	الآن	آناء	آنفا	أبدا	أصلا
أمد	أمس	أول	أيّان	إذ	بعد	تارة	حين	صباح	ضحوة
غدًا	غداة	مرّة	مساء	يومئذ	خلال	أمام	أين	إزاء	بين
تحت	ثمّ	خلف	شمال	ضمن	فوق	يمين	حوالى	حول	بضع
ذيت	فلان	كأيّ	كأين	كأيّن	كذا	كم	كيت		

اسم فعل

آهاً	بسّ	حايِ	صهْ	صهٍ	طاق	طَق	عَدَسْ	كِخ	نَخْ
هَجْ	وا	واهاً	وَيْ	آمينَ	آه	أُفٍّ	أُفٍّ	أمامك	أمامكَ
أوّهْ	إلَيْكَ	إلَيْكَ	إليكَ	إلِيكم	إليكم	إليكما	إليكنّ	إيهٍ	بخٍ
بَسْ	بطآن	بَلْهَ	حَذارِ	حيَّ	حيَّ	دونك	رويدك	سرعان	شَتَّانَ
عليك	مكانَك	مكانَك	مكانَك	مكانكم	مكانكما	مكانكنّ	مه	ها	هاؤم
هاكَ	هلمَّ	هيّا	هيت	هَيْهَاتَ	وا	وراءَك	وُشْكَانَ	ويكأنّ	

فعل

بِئْسَ	حَبَّذَا	سَاءَ	سَاءَمَا	نِعْمَ	نِعِمّا	حاشا	خلا	عدا	طالما
قلما	ابتدأ	اخلولق	انبرى	أخذ	أقبل	أنشأ	أوشك	جعل	حرى
شرع	طفق	عسى	علق	قام	كاد	كرب	هبّ	ارتدّ	استحال
انقلب	آض	أصبح	أضحى	أمسى	بات	تبدّل	تحوّل	حار	راح
رجع	صار	ظلّ	عاد	غدا	كان	ماانفك	مابرح	مادام	مازال
مافتئ	لَيْسَ	لَسْتُ	لَسْنَا	لَسْتَ	لَسْتِ	لَسْتُمَا	لَسْتُمْ	لَسْتُنَّ	لَيْسَتْ
لَيْسَا	لَيْسَتَا	لَيْسُوا	لَسْنَ						

حرف ابجدي

ء	ؤ	ئ	آ	أ	أ	ب	ت	ة	ث
ج	ح	خ	د	ذ	ر	ز	س	ش	ص
ض	ط	ظ	ع	غ	ف	ق	ك	ل	م
ن	ه	و	ى	ي					