N° 04/2016 –D/INF

Université des Sciences et de la Technologie Houari Boumediene

Faculté d'Électronique et d'Informatique

Université Rouen Normandie

Thèse en COTUTELLE

Présentée pour l'obtention du grade de DOCTEUR EN SCIENCES

En : INFORMATIQUE

Spécialité : Intelligence Artificielle et Bases de Données Avancées

Par : Mounia HADDOUD

Sujet

INDEXATION ET EXTRACTION DE TERMES-CLES POUR LA RECHERCHE D'INFORMATION

Soutenue publiquement le 11 /07 /2016 devant le jury composé de :

Mme. Hadja Faïza KHELLAF-HANED	Professeur / USTHB	Présidente
M. Mohand BOUGHANEM	Professeur / Université Paul Sabatier - Toulouse	Examinateur
M. Bruno CREMILLEUX	Professeur / Université de Caen	Examinateur
M. Abdelouahab MOUSSAOUI	Professeur / Université Ferhat Abbas – Sétif 1	Examinateur
M. Saïd ABDEDDAÏM	MCF / Université de Rouen	Invité
M. Thierry LECROQ	Professeur / Université de Rouen	Directeur de thèse
Mme. Aïcha AÏSSANI-MOKHTARI	Professeur / USTHB	Directrice de thèse

Indexation et extraction de termes-clés pour la recherche d'information

Mounia Haddoud

Table des matières

Résumé

Dans cette thèse, je me suis intéressée à deux problèmes de fouille de textes : l'extraction automatique de termes-clés dans des documents textuels et la classification de textes. Pour le problème d'extraction automatique de termes-clés, j'ai défini une nouvelle mesure, le DPM-index, qui discrimine les phrases (n-grammes) qui se chevauchent dans un document. J'ai aussi développé un nouveau système d'extraction de termes-clés basé sur l'apprentissage supervisé qui combine 18 descripteurs statistiques. J'ai expérimentalement comparé mes résultats à ceux de 21 méthodes d'extraction de termes-clés sur le corpus d'articles scientifiques SemEval-2010/Task-5. Ma méthode augmente d'un taux de 13 % la reconnaissance des termes-clés mesurée par le F-score. En particulier, le DPM-index augmente la reconnaissance de mon système d'extraction de termes-clés de 9%. Je montre également que quel que soit le paradigme d'apprentissage supervisé (*boosting*, *bagging* et régression) sur ces données pour combiner les 18 descripteurs mon système obtient les meilleurs performances. Pour le problème de classification de textes dans des catégories prédéfinies, j'ai proposé 80 métriques de pondération de termes jamais utilisées pour ce problème et je les ai comparé à 16 métriques de la littérature. Alors que de nombreux travaux antérieurs ont montré l'intérêt d'utiliser une métrique particulière, mes expérimentations suggèrent que les résultats obtenus par ces métriques peuvent être fortement dépendants de la distribution des documents dans les catégories et des mesures de performances utilisées. La solution que j'ai proposé consiste à combiner les métriques proposées afin d'améliorer la qualité de la classification. Plus précisément, j'ai montré sur trois types de corpus différents (ayant des distributions catégorielles différentes) que l'utilisation d'un classifieur SVM qui combine les sorties de classifieurs SVM (qui utilisent chacun une métrique de pondération différente) classe mieux les documents quel que soit le type corpus et quelles que soit les mesures de performance utilisés. La seconde contribution principale apportée au problème de classification est une représentation étendue des termes d'un document dans un espace vectoriel qui permet d'améliorer la prédiction de mon classifieur de textes.

Mots clés

Fouille de données ; Fouille de textes ; Classification de textes ; Extraction de mots-clés ; Exploration de données ; Apprentissage automatique.

Abstract

In this thesis, I focused on two text mining problems : automatic keyphrase extraction in text documents and texts classification. For the automatic keyphrase extraction problem, i define the document phrase maximality index (DPM-index), a new measure to discriminate overlapping keyphrase candidates in a text document. As an application i developed a supervised learning system which uses 18 statistical features, among them the DPM-index and 5 other new features. I experimentally compare my results to those of 21 keyphrase extraction methods on SemEval-2010/Task-5 scientific articles corpus. When all the systems extract 10 keyphrases per document, my method enhances by 13% the F-Score of the best system. In particular, the DPM-index feature increases the F-Score of my keyphrase extraction system by a rate of 9%. This makes the DPM-index contribution comparable to that of the well-known TFIDF measure on such a system. For the text classification problem, i propose 80 metrics never used for the term weighting problem and compare them to 16 functions of the literature. A large number of these metrics were initially proposed for other data mining problems : feature selection, classification rules and term collocations. While many previous works have shown the merits of using a particular metric, my experience suggests that the results obtained by such metrics can be highly dependent on the label distribution on the corpus and on the performance measures used (microaveraged or macroaveraged F1-Score). The solution i propose consists in combining the metrics in order to improve the classification. More precisely, i show that using a SVM classifier which combines the outputs of SVM classifiers that utilize different metrics performs well in all situations. The second main contribution is an extended term representation for the vector space model that improves significantly the prediction of the text classifier.

Keywords

Data mining ; text mining ; Texts classification ; Keywords extraction ; Information retrieval ; Machine learning.

Remerciements

J'adresse mes remerciements aux personnes qui m'ont aidé dans la réalisation de ce mémoire. En premier lieu, je remercie M. Abdeddaïm Saïd , maître de conférences à l'université Rouen Normandie. En tant qu'encadrant de thèse, il m'a guidé dans mon travail et m'a aidé à trouver des solutions pour avancer.

Je remercie aussi Me. Aïssani-Mokhtari Aïcha professeur à l'USTHB et M. Lecroq Thierry professeur à l'université Rouen Normandie pour avoir accepté d'être mes directeurs de thèse.

Je remercie également M. Boughanem Mohand professeur à l'université Paul Sabatier Toulouse, M. Cremilleux Bruno professeur à l'université Caen Normandie et M. Moussaoui Abdelouahab professeur à l'université Ferhat Abbes Sétif pour avoir accepté de rapporter cette thèse, ainsi que Me. Khellaf-Haned Faïza professeur à l'USTHB, qui a accepté d'en être présidente.

Je tiens à remercier l'ensemble des membres du laboratoire RIIMA et de l'équipe TIBS du laboratoire LITIS. Quant à l'ambiance conviviale qui règne aux laboratoires.

Introduction

Avec la croissance du nombre de productions scientifiques, l'extraction manuelle des mots-clés, que j'appelle ici termes-clés, est de plus en plus difficile. Cette tâche nécessite un travail de maintenance des ressources terminologiques utilisées pour l'indexation et des effectifs humains conséquents afin de tenir la charge journalière de données à indexer. Soucieux de faciliter le travail d'extraction des termes-clés de toutes sortes de documents (articles scientifiques, articles journalistiques, nouvelles, etc.) et pour toute sorte d'application (indexation, résumés, publicité ciblée, etc.), de nombreux chercheurs s'intéressent à son automatisation. En témoignent le nombre grandissant d'articles scientifiques à ce sujet [43] ainsi que l'émergence de campagnes d'évaluation [57, 87].

Étant donné un document, l'extraction automatique des termes-clés consiste à trouver les unités textuelles qui décrivent son contenu principal. La difficulté de cette tâche réside dans l'identification des éléments importants vis-à-vis de son contenu, ainsi que leur représentation avec les unités textuelles appropriées. La première difficulté est d'ordre sémantique : il faut réussir à comprendre le document pour en extraire l'essence ; la seconde est d'ordre linguistique et terminologique : il faut déterminer les propriétés linguistiques des termes-clés et connaître le vocabulaire du domaine auquel appartient le document. Par ailleurs, la forme la plus appropriée pour un terme-clé n'est pas nécessairement présente dans le contenu du document, elle peut être implicite.

Plutôt que de comprendre le document, les méthodes d'extraction automatique des termes-clés les plus opérationnelles se fondent sur des statistiques et des modélisations particulières de celui-ci. Pour ce qui est de l'usage d'unités textuelles appropriées, elles se contentent le plus souvent de celles qui occurrent dans le document. De manière générale, des termes-clés candidats sont sélectionnés dans le document d'après des critères prédéfinis (par exemple, ce doit être des groupes nominaux), ces candidats sont analysés et les termes-clés sont ensuite extraits d'entre eux en fonction du résultat de l'analyse. L'analyse des termes-clés candidats du document peut être réalisée

avec deux approches : supervisée ou non supervisée. L'approche supervisée consiste à analyser les caractéristiques des termes-clés de données manuellement indexées pour apprendre à reconnaître les termes-clés. Tandis que l'approche non supervisée consiste à chercher les candidats qui se distinguent des autres dans le document sans analyse des termes-clés choisis par l'expert.

Dans le cadre de cette thèse, j'ai défini une nouvelle mesure, le DPM-index, qui discrimine les phrases (n-grammes) qui se chevauchent dans un document [34]. J'ai aussi développé un nouveau système d'extraction de termes-clés basé sur l'apprentissage supervisé qui combine 18 descripteurs statistiques [34, 35]. J'ai expérimentalement comparé mes résultats à ceux de 21 méthodes d'extraction de termes-clés sur le corpus d'articles scientifiques SemEval-2010/Task-5. Ma méthode augmente d'un taux de 13 % la reconnaissance des termes-clés mesurée par le F-score. En particulier, le DPM-index augmente la reconnaissance de mon système d'extraction de termes-clés de 9%. Je montre également que quel que soit le paradigme d'apprentissage supervisé (*boosting*, *bagging* et régression) sur ces données pour combiner les 18 descripteurs mon système obtient les meilleurs performances.

Dans un second temps, afin de répondre au besoin d'accès efficace aux vastes collections de documents disponibles sur le web, je me suis intéressée au problème de classification de textes. La classification de textes a pour objectif de regrouper les textes similaires, c'est-à-dire thématiquement proches, au sein d'un même ensemble dans une catégorie (classe). Un texte peut appartenir à un nombre de catégories variables, mais toutes les catégories sont prédéfinies. L'intérêt d'une telle démarche est d'organiser les connaissances de façon à pouvoir effectuer, par la suite, une recherche ou une extraction d'information efficace. On distingue dans le domaine de la classification automatique deux types d'approches : la classification supervisée et la classification non supervisée. Dans le cas de la classification supervisée, les documents sont classés après un apprentissage préalable sur un corpus étiqueté.

La classification de documents repose sur différents éléments, comme les classifeurs, le choix des descripteurs ou encore la pondération des descripteurs. La pondération des descripteurs est une étape cruciale dans le processus de classification. De la qualité de celle-ci dépend la qualité des modèles de classifications.

Je propose dans ce travail de nouvelles métriques de pondération de termes qui se base sur les différents éléments d'un corpus : les classes, les documents et les termes. Ces métriques sont dites supervisées car elles utilisent la fréquence de co-occurence des termes avec les catégories observés dans le corpus d'apprentissage. Je compare expérimentalement 93 métriques pour la pondération des termes. Parmi celles-ci seulement 13 d'entre elles ont déjà été utilisées pour le problème de pondération des termes dans la

littérature et 9 sont de nouvelles métriques proposées dans cette thèse. Les 71 métriques restantes ont été utilisées dans d'autres problèmes d'extraction de données mais pas pour la pondération de termes en vue de la classification de documents.

Je montre sur plusieurs corpus que l'utilisation de ces métriques, au lieu de celles déjà utilisées, peut améliorer les performances des classifieurs SVM, en particulier pour la classification d'articles scientifiques biomédicaux [37]. Alors que de nombreux travaux antérieurs ont montré l'intérêt d'utiliser une métrique particulière, mes expérimentations suggèrent que les résultats obtenus par ces métriques peuvent être fortement dépendants de la distribution des documents dans les catégories et des mesures de performances utilisées. La solution que j'ai proposé consiste à combiner les métriques proposées afin d'améliorer la qualité de la classification. Plus précisément, j'ai montré sur trois types de corpus différents (ayant des distributions catégorielles différentes) que l'utilisation d'un classifieur SVM qui combine les sorties de classifieurs SVM (qui utilisent chacun une métrique de pondération différente) classe mieux les documents quel que soit le type de corpus et quelles que soient les mesures de performance utilisés [36]. La seconde contribution principale apporté au problème de classification est une représentation étendue des termes d'un document dans un espace vectoriel qui permet d'améliorer la prédiction de mon classifieur de textes.

Après une introduction générale le manuscrit s'articule autour de deux parties. La première partie contient un état de l'art sur l'extraction automatique des termes-clés puis présente le système d'extraction proposé. Elle est subdivisée en deux chapitres :

— Le premier chapitre présente le principe des systèmes d'extraction automatique et les différentes approches sur lesquelles ils se basent ainsi que les méthodes et outils utilisés.
— Le deuxième chapitre détaille le fonctionnement du système d'extraction proposé et présente les principaux résultats des performances de ce système comparé aux autres systèmes étudiés.

La seconde partie contient un état de l'art sur la classification de textes, les méthodes d'apprentissages et présente les 96 métriques de pondération des termes utilisées pour ce problème. Elle est subdivisée en trois chapitres :

— Le premier chapitre est dédié à présenter les méthodes d'apprentissages utilisées pour la classification de textes.
— Le second chapitre présente le principe des méthodes de classification de textes et les différentes approches adoptées pour ce problème.
— Le troisième chapitre porte sur les 80 nouvelles métriques de pondération des termes étudiées et leurs comparaisons avec les 16 métriques déjà utilisées dans la littérature. Il porte aussi sur une représentation étendue

des termes d'un document. Tous les résultats expérimentaux y sont présentés.

Chapitre 1

Extraction de termes-clés

Sommaire

1.1 Introduction

Étant donné un document textuel, les termes-clés (phrases ou n-grammes) sont des mots ou des expressions représentant les sujets principaux qui sont abordés dans le document, ils s'avèrent de plus en plus utiles avec l'essor de l'Internet et la disponibilité de nombreux documents numériques qu'il faut pouvoir indexer de manière pertinente pour faciliter leur recherche par des utilisateurs. Bien que les termes-clés soient utiles pour de multiples tâches, très peu de documents en sont pourvus, du fait du coût important de production de ceux-ci, en termes de temps et de ressources humaines. Pour y remédier de nombreux chercheurs s'intéressent à l'extraction automatique de ceux-ci et certaines campagnes d'évaluations, telles que SemEval [57], proposent des tâches d'extraction automatique de termes-clés dans le but de confronter les différents systèmes existants. Pour ce faire, les données et la méthode d'évaluation sont les mêmes pour tous les systèmes. L'extraction de termes-clés ne doit pas être confondue avec un problème voisin : l'assignation automatique de termes-clés ou classification de textes. Dans la classification de textes le vocabulaire des termes clés (catégories) est contrôlé (prédéfini) alors que dans le présent problème on doit extraire ces termes sans connaissance préalable.

1.2 Problèmes et applications

Dans ce chapitre, seules les méthodes d'extraction automatique de termes-clés sont présentées. Celles-ci appartiennent à deux catégories distinctes : les méthodes supervisées et les méthodes non-supervisées. Dans le cas supervisé, l'extraction des termes-clés est effectuée grâce à un apprentissage préalable servant à calibrer la méthode avec un corpus dont les documents sont annotés en termes-clés. Les méthodes non-supervisées ne requièrent pas de phase d'apprentissage. Elles exploitent des représentations efficaces des documents ainsi que des propriétés définies à partir de descripteurs statistiques pour extraire les termes-clés parmi des termes candidats.

Les documents sont de types différents : articles scientifiques (la plupart des références bibliographiques sont de ce type), les articles de journaux [130, 50, 117, 118], les transcriptions de meeting [67], les pages web [113, 130, 79, 128, 12, 68, 104, 132], les couriels [113], les réseaux sociaux [65, 134, 55] etc. Parmi les applications de l'extraction automatique des termes-clés : la gestion des librairies digitales [77, 102], la publicité sur le web basée sur le contenu [128, 68, 53, 104, 132], la recommandation de tags basée sur le contenu [76, 91], la recherche documentaire et web [92], les résumés [130, 133],

le clustering des documents [54], l'extension des requêtes [102], et l'analyse automatique de sentiments [17, 23].

1.3 Principes des méthodes d'extraction

Le principe d'un système d'extraction automatique de termes-clés consiste à analyser un document pour en extraire les propriétés qui pourraient distinguer les termes les plus représentatifs de celui-ci parmi d'autres termes. L'extraction automatique peut être vu comme une méthode de classification binaire des termes du document en termes-clés ou non termes-clés. Les méthodes d'extraction sont réparties en deux catégories : les méthodes supervisées et les méthodes non supervisées. Les méthodes supervisées reposent sur un apprentissage sur un corpus de documents dont ont connait les mots clés. Toute ces méthodes utilisent des descripteurs concernant les termes extraits des documents analysés pour vérifier des propriétés permettant d'identifier les termes-clés. Pour améliorer l'extraction de termes-clés une étape de prétraitement est mise en œuvre dans le but de réduire les documents à un ensemble de termes candidats définies chacun par ses descripteurs.

1.4 Prétraitement de texte

Le prétraitement sur les textes peut affecter de manière significative la précision de la méthode d'extraction de termes-clés. Ainsi, il est important de donner quelques détails sur la façon dont nous l'avons mis en place.

1.4.1 Segmentation (Tokenization)

Un système d'indexation, s'il opère sur des documents volumineux, pourra tenir compte des différentes unités d'indexation (unité linguistique) que sont la phrase, le paragraphe, ou le document dans son ensemble.

Phrases

On effectue le découpage d'un document en phrases en utilisant les points comme séparateurs de phrases ou pour une reconnaissance plus fine on peut utiliser un programme de segmentation.

n-grammes

Les n-grammes sont des séquences ordonnées de n mots. Les n-grammes sont ensuite filtrés. Leur sélection est très exhaustive, elle fournit un grand nombre de termes-clés candidats, ce qui permet de maximiser la quantité de candidats présents dans l'ensemble des termes-clés de référence, mais ce qui maximise aussi la quantité de candidats erronés (bruités).

Les n-grammes sont d'abord filtrés (ponctuations, nombres, symboles, caractères spéciaux ...). Dans cette thèse nous considérons les n-grammes avec un $n \leq 4$.

1.4.2 Étiquetage syntaxique (POS tagging)

L'étiquetage syntaxique consiste à assigner à chaque mot de la phrase sa catégorie syntaxique (nom, verbe, adjectif, ...). L'intérêt de l'étiquetage est qu'il rend possible l'analyse d'un corpus du point de vue linguistique. À partir d'un texte étiqueté, il est en effet possible de caractériser ou de rechercher certaines structures syntaxiques, c'est-à-dire, de s'abstraire du mot pour passer à sa catégorie linguistique. Par exemple, la phrase :

— "Two resistant strains were isolated after four rounds of selection."

est étiquetée par

— "Two/CD resistant/JJ strains/NNS :pl were/VBD isolated/VBN after/IN four/CD rounds/NNS :pl of/IN selection/NN ./."

Dans le cadre de cette thèse nous avons utilisé le Stanford POS-tagger [107] pour la segmentation et l'étiquetage syntaxique.

1.4.3 Racinisation (Stemming)

La racinisation est le nom donné au procédé qui vise à transformer les flexions en leur radical (ou racine).

Exemple :

Les mots suivants ont pour racine "OPTIM" : Optimiser, Optimal, Optimisation, Optimalisation.

La variation d'un terme peut affecter sa fréquence qui est un indicateur important pour savoir si c'est un terme-clé ou pas. La solution consiste à réduire les mots à leur radicaux après effacement des affixes (suffixes, préfixes, postfixes, antéfixes). Ceux sont des techniques de normalisation déjà bien étudiées qui reposent sur des grammaires et des lexiques actuellement assez complets même si le traitement des mots inconnus reste un problème pour la racinisation. Quelque soit l'outil retenu, la façon de procéder est toujours la même : le stemmatiseur recherche selon la forme du mot fléchi et la langue

définie, le radical le plus probable pour ce mot. Il existe différents algorithmes de racinisation [101], nous avons utilisé l'algorithme Lovins stemmer [72] qui est plus agressif que d'autres algorithmes. Ce dernier donne de meilleurs résultats pour la méthode d'extraction de termes-clés [113].

1.4.4 Filtrage linguistique

L'utilisation de filtres linguistiques pour les termes candidats dans les documents à un impact sur la qualité de l'extraction automatique de termes-clés.

Les Patrons syntaxiques

Pour l'informatique, les patrons syntaxiques sont une des techniques possibles pour accéder au contenu informationnel de documents numériques contenant du texte. Ils sont donc utilisés pour rechercher des informations précises, dans le cadre du traitement automatique des langues pour comprendre, résumer, analyser des textes, en modélisation de connaissances (construction et peuplement d'ontologies par exemple) ou encore en extraction d'information. En extraction d'information, l'objectif est de repérer dans des textes des phrases pour ensuite figer les régularités observées dans une représentation, sous un format exploitable automatiquement. L'informatique se focalise donc sur les aspects techniques, la forme et l'efficacité de traitement des textes à l'aide des patrons pour atteindre un objectif donné, plus que sur leur fonctionnement détaillé dans les textes. L'implémentation des patrons syntaxiques correspond généralement à des automates à états finis, à des expressions rationnelles balayées par un programme qui cherche à retrouver, dans un corpus, la séquence d'éléments ainsi définis.

L'utilisation d'une méthode d'identification automatique des patrons en corpus permet d'affiner les observations réalisées et d'obtenir de meilleurs résultats

Selon les connaissances linguistiques, les syntagmes nominaux sont plus susceptibles d'être des termes-clés. Toutefois, la définition de syntagme nominal peut varier d'un système à l'autre.

Turney [111] a proposé de garder comme candidats les syntagmes nominaux correspondant au patron syntaxique suivant :

```
(NN-NNS-NNP-NNPS-JJ) * (NN-NNS-NNP- NNPS-VBG)
```

Hulth [45] considère les 56 patrons syntaxiques les plus fréquemment utilisés pour les termes-clés dans les corpus d'apprentissage et les utilise comme filtre

pour la sélection des termes candidats. De plus, le patron de chaque terme est utilisé comme descripteur de sélection de termes-clés par l'algorithme d'apprentissage.

Nguyen et al. [82] considèrent les termes candidats correspondants au patron syntaxique suivant :

```
NBAR = (NN-NNS-NNP-NNPS-JJ-JJR-JJS) * (NN-NNS-NNP-NNPS)
```

.

et considèrent également la séquence d'étiquettes syntaxiques du terme candidat comme descripteur parmi l'ensemble des descripteurs. Ce patron a été amélioré en ajoutant le motif

```
NBAR EN NBAR
```

dans les travaux qui ont suivi [58, 59].

Krapivin et al. [62] utilisent l'étiquetage syntaxique de chaque phrase comme descripteur pour le classer comme terme-clé ou non. Liu et al. [67] filtrent les termes candidats qui correspondent à

```
(JJ) * (NN-NNS-NNP) +
```

Pal et al. [85] gardent les syntagmes nominaux qui satisfont le patron syntaxique

```
(NN-NNS-NNP-NNPS-JJ) * (NN-NNS-NNP-NNPS-VBG)
```

et utilisent aussi l'étiquetage syntaxique comme descripteur.

1.4.5 Indexation des documents

Toutes les positions d'une racine d'un terme sont conservées. Comme les occurrences d'une racine d'un terme peuvent apparaître sous différentes formes (différents n-grammes), nous avons décidé que le n-gramme d'un terme est le plus fréquent dans le document. Le même problème se pose pour les étiquettes syntaxiques (POS), nous avons considéré deux cas :

- Si la majorité des étiquettes syntaxiques correspondent à des syntagmes nominaux (tel que défini ci-dessus) nous choisissons les étiquettes les plus fréquentes ;
- Sinon une étiquette syntaxique autre que syntagme nominal est choisie.

À la fin de cette étape, le document est transformé en un ensemble de termes, chaque terme t dans le document d est défini comme un tuple :

(n-gramme, racine, étiquettes syntaxiques, d, { positions}).

1.4.6 Indexation du corpus

Afin de calculer les descripteurs dépendants du corpus, nous avons besoin d'indexer tous les documents des données d'apprentissage. Après l'indexation du corpus, chaque terme du document devient un tuple :

(n-gramme, racine, balise, d, {positions}, {documents})

où {documents} représente l'ensemble des documents des données d'apprentissage contenant ce terme, c'est-à-dire un terme qui a la même racine.

1.5 Descripteurs

La conception d'un système d'extraction de termes-clés consiste à sélectionner les propriétés qui pourraient distinguer les termes-clés des autres termes. Ces propriétés sont appelées descripteurs.

1.5.1 Définitions et notations

Certains descripteurs utilisés dans la littérature ont des définitions différentes qui ont des impacts différents sur la qualité d'extraction de termes-clés. Comme des définitions précises de descripteurs ne sont pas nécessairement données dans les articles, les prédictions de termes-clés présentées ne sont guère reproductible, nous avons donc introduit un ensemble de notations et définitions :

- d : un document, $|d|$ sa taille en nombre de mots le constituant et $|d|_s$ sa taille en nombre de phrases qu'il contient ;
- D : la collection de document ou corpus, $|D|$ sa taille ;
- T : l'ensemble de tous les termes sélectionnés à partir du corpus de documents après l'étape de prétraitement, $|T|$ sa taille ;
- T_d : l'ensemble de tous les termes sélectionnés à partir du document d après l'étape de prétraitement, $|T_d|$ sa taille ;
- t : terme de T, $|t|$ sa taille en nombre de mots le constituant ;
- s : phrase, $|s|$ sa taille en nombre de mots la constituant ;
- S_d : les phrases de d, $|S_d|$ sa taille ;
- $S_d(t)$: les phrases de d contenant t, $|S_d(t)|$ sa taille ;
- $\text{head}(d, r)$: la première partie du document d de taille $r|d|$, avec r un ratio compris entre 0 et 1 ;
- $f(t, d)$: la fréquence de t dans d ;
- $f(t, D)$: la fréquence de t dans le corpus D ;
- $\text{df}(t, D)$: le nombre de documents de D où t apparait (fréquence des documents) ;

- $p(t, d)$: une estimation de la probabilité d'un terme t étant donné d : $p(t, d) = \frac{f(t,d)}{\sigma_{t' \in T_d} f(t',d)}$;
- $p(t, D)$: une estimation de la probabilité d'un terme t étant donné D : $p(t, d) = \frac{f(t,d)}{\sigma_{t' \in T} f(t',D)}$;
- $\text{pos}_n(t, d)$: la position de la n-ième occurrence de t dans d ;
- $\text{npos}_n(t, d)$: la n-ième position normalisée. $\text{npos}_n(t, d) = \frac{\text{pos}_n(t,d)}{|d|}$;
- $\text{sent}_n(t, d)$: le nombre de phrases contenant la n-ième occurrence de t dans d ;
- $\text{comp}(t)$: les composants de t, i.e., les mots du n-gramme t ;
- $\text{sub}(t)$: les sous-termes de t, i.e., tous les m-grammes contenus (sous-chaînes de) dans le n-gramme t, avec $m \leq n$;
- $\text{sup}(t, d)$: les super-termes t, i.e., tous les termes sélectionnés s du document d contenant t, mais différents de t ;
- $\text{sup}(t, D)$: les super-termes t dans le corpus D, i.e., tous les termes sélectionnés s du corpus D contenant t, mais différents t ;
- $\text{TF}(t, d)$: la fréquence du terme normalisée d'un terme t dans un document d : $\text{TF}(t, d) = f(t, d)/|d|$;
- $\text{IDF}(t, D)$: l'inverse de la fréquence du document de t dans le corpus D : $\text{IDF}(t, D) = log(|D|/df(t, D))$;
- $\text{TFIDF}(t, d, D)$: $\text{TFIDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$;

1.6 Types de descripteurs

Les descripteurs peuvent être classés en fonction de leur niveau (origine) : les descripteurs liés au niveau terme, les descripteurs liés au niveau document, les descripteurs liés au niveau corpus et les descripteurs liés au niveau bases de connaissances externes.

L'objectif de ces approches est de définir des groupes dont les termes partagent une ou plusieurs caractéristiques communes. Ainsi, lorsque des termes-clés sont extraits à partir de chaque groupe, cela permet de mieux couvrir le document analysé selon les descripteurs utilisés.

Les méthodes d'extraction sont alors traditionnellement classées selon les descripteurs qu'elles utilisent en deux catégories : dépendantes du document ou dépendantes du corpus. Ces deux catégories deviennent quatre si l'on tient compte du fait qu'elles utilisent, ou pas, des bases de connaissances externes. Les descripteurs peuvent également être classés en fonction de leur nature : les descripteurs statistiques, linguistiques et structuraux.

1.6.1 Descripteurs par niveaux

Les descripteurs liés au niveau terme et au niveau document

Les approches dépendantes du document [78, 75, 86, 70, 96, 68, 124] se basent sur les descripteurs liés au terme et les descripteurs liés au niveau document tels que :

— la longueur d'un terme (LEN), i.e., Le nombre de mots qu'il contient [112, 113]

$$\mathrm{LEN}(t) = |t|$$

— la fréquence d'un terme (TF) dans le document [112, 113]

$$\mathrm{TF}(t, d) = \frac{f(t, d)}{|d|}$$

— la première position dans le document $\mathrm{pos}_0(t, d)$ [112, 113, 27, 125]

— la fréquence de co-occurrence du terme avec d'autres termes des documents [75, 78].
Par exemple le coefficient de Jaccard [48] entre deux termes peut être utilisé pour calculer cette fréquence.

Les descripteurs liés au niveau corpus

En plus des descripteurs liés au terme et ceux liés au document, *les approches dépendantes du corpus* (la majorité des références citées) utilisent généralement le nombre de documents du corpus contenant un terme comme un indicateur de sa spécificité. Les termes qui apparaissent fréquemment dans le document, mais rarement dans le reste du corpus sont plus susceptibles d'être des termes-clés, tels que :

— la fréquence inverse du document (IDF) qui dépend du nombre de documents du corpus qui contiennent le terme.

$$\mathrm{IDF}(t, D) = \log \frac{|D|}{\mathrm{df}(t, D)}$$

— TFIDF [27, 125] qui combine TF et IDF,

$$\mathrm{TFIDF}(t, d, D) = \mathrm{TF}(t, d) \times \mathrm{IDF}(t, D)$$

Les descripteurs liés aux bases de connaissances externes

Les approches dépendantes de bases de connaissances externes utilisent des sources telles que

— les bases de données de terminologies : GRISP, MeSH, etc. [46, 77, 30, 71, 7],
— les ressources linguistiques : WordNet, etc. [21, 123],
— les logs des requêtes des moteurs de recherches [21, 123],
— et les résultats des moteurs de recherches : rank scores et/ou text-snippets de Google, Bing, MSN, etc.[114, 79, 53, 12, 19].

Auparavant, des bases de données de terminologies ont été utilisées par des méthodes de *catégorisation de textes* [1] basées essentiellement sur ces informations pour attribuer, *a priori*, les termes-clés à un document.

1.6.2 Descripteurs par nature

Les descripteurs peuvent également être classés en fonction de leur nature :

Les descripteurs statistiques

Les plus utilisés sont
— la longueur d'un terme $\mathrm{LEN}(t)$;
— la fréquence d'un terme dans le document $\mathrm{TF}(t, d)$;
— la fréquence inverse du document qui dépend du nombre de documents du corpus qui contiennent le terme $\mathrm{IDF}(t, D)$;
— $\mathrm{TFIDF}(t, d, D)$.
et une variante
$\mathrm{TFIDF}_2(t, d, D) = \log \mathrm{TF}(t, d) \times \max(0, \log \frac{|D| - \mathrm{df}(t,D)}{\mathrm{df}(t,D)})$.

Les descripteurs linguistiques

Les étiquettes syntaxiques (POS) voir la section 1.4.4 et les syntagmes nominaux [45, 62, 61] sont des descripteurs linguistiques qui tentent de saisir les propriétés linguistiques de termes-clés.

Les descripteurs structuraux

Les descripteurs structuraux fournis par des documents HTML ou XML aident aussi à l'identification des termes-clés [121, 128, 82, 44, 83, 71].
— l'apparition dans le titre (T), lien hypertexte (H), capitalisation (C), .

1.7 Ordonnancement des termes-clés

Les descripteurs sélectionnés sont combinés pour donner un score synthétique à chaque terme du document. Ce score est utilisé pour classer tous les termes candidats, les k-meilleurs termes classés sont les termes-clés que nous recherchons. La façon dont ce score est calculé dépend des méthodes utilisées, certaines utilisent une approche non supervisée alors que d'autres utilisent l'apprentissage supervisé. Certaines approches non supervisées sont tout simplement basées sur une formule de score qui combine les valeurs des descripteurs comme le classique TFIDF ou d'autres expressions [56, 20, 129]. D'autres approches utilisent des algorithmes non supervisés plus sophistiqués [130, 106, 78, 75, 86, 123, 117, 118, 119, 67, 70, 68, 96, 124, 132, 80], généralement une méthode de clustering ou une approche à base de graphes [42] inspirée de l'algorithme, bien connu, PageRank [11]. L'apprentissage supervisé est réalisé avec des approches d'apprentissage comme le bagged C4.5 [113], le modèle naïf Bayesien [27, 125], les réseaux de neurones [50], les Séparateurs à Vaste Marge (SVM) [121, 49], l'entropie maximale [128] .

1.8 Outils et méthodes

1.8.1 Les méthodes non supervisées

Les méthodes non-supervisées d'extraction de termes-clés ont la particularité de s'abstraire du domaine et de la langue des documents à analyser. Cette abstraction est due au fait que les termes candidats sont analysés avec des règles simples déduites à partir de descripteurs statistiques issus seulement du texte analysé, ou bien d'un corpus de référence non annoté. De nombreuses approches sont proposées. Certaines se fondent uniquement sur des statistiques alors que d'autres les combinent avec des représentations plus complexes des documents. Ces représentations peuvent aller de groupes de mots sémantiquement similaires à des graphes dont les nœuds sont des unités textuelles (mots, expressions, phrases, etc.) liées par des relations de recommandation.

Approches statistiques

Si on prenait le descripteur booléen comme poids d'un terme, sa valeur est égal à 1 s'il apparaît, 0 sinon. De même si un même descripteur apparaissait dans deux documents, cela ne veut pas signifier qu'il ait une importance similaire. Définir le poids des descripteurs implique qu'il faut classer les descripteurs selon leur représentation (savoir quel descripteur est plus

représentatif que l'autre) et les ajuster pour mettre en avant les descripteurs les plus discriminants. Dans un contexte où l'objectif final est la comparaison des vecteurs représentants les documents, l'hypothèse souvent retenue est que deux vecteurs partageant des descripteurs rares sont plus discriminants que deux vecteurs partageant des descripteurs fréquents [116].

La métrique la plus utilisée pour représenter cette idée est le TFIDF et ses dérivés [112, 113, 27, 125], métriques issues du monde de la recherche d'information. Cette variante prend en compte la longueur des documents dans le calcul de discriminance. Le principe du TFIDF est de donner un poids plus important aux descripteurs les plus spécifiques d'un document. Elle repose sur le produit entre la fréquence du terme TF et la fréquence inverse du document IDF. La fréquence du terme correspond au nombre d'occurrences d'un descripteur dans un document et représente le poids du descripteur au sein du document.

La fréquence inverse de document IDF mesure l'importance du descripteur dans le corpus. L'objectif est de donner un poids plus important aux descripteurs qui apparaissent dans peu de documents. Il est calculé en considérant le logarithme de l'inverse de la fréquence de documents qui contiennent le descripteur dans le corpus.

Le TFIDF est obtenu en multipliant TF par IDF du terme t pour le document d : $TFIDF = TF \times IDF$.

Approches à base de graphes

Les approches à base de graphes consistent à représenter le contenu d'un document sous la forme d'un graphe. La méthodologie appliquée est issue de *PageRank* [11], un algorithme d'ordonnancement de pages Web (nœuds du graphe) grâce aux liens de recommandation qui existent entre elles (arcs du graphe). *TextRank* [78] et *SingleRank* [118] sont les deux adaptations de base de PageRank pour l'extraction automatique de termes-clés. Dans celles-ci, les pages Web sont remplacées par des unités textuelles dont la granularité est le mot et un arc est créé entre deux nœuds si les mots qu'ils représentent co-occurrent dans une fenêtre de mots donnée.

Le graphe est noté $G = (N, A)$, où N est l'ensemble des nœuds du graphe et où A est l'ensemble de ses arcs entrants et sortants : $A_{\text{In}} \cup A_{\text{Out}}$. Pour chaque nœud du graphe, un score 1.1 est calculé par un processus itératif destiné à simuler la notion de recommandation d'un terme par d'autres. Ce score à chaque nœud permet d'ordonner les mots par degré d'importance dans le document analysé. La liste ordonnée des mots peut ensuite être utilisée pour

extraire les termes-clés.

$$S(n_i) = (1 - \lambda) + \lambda \times \sum_{n_j \in A_{\text{In}}(n_i)} \frac{p_{j,i} \times S(n_j)}{\sum_{n_k \in A_{\text{Out}}(n_j)} p_{j,k}} \tag{1.1}$$

λ est un facteur d'atténuation qui peut être considéré ici comme la probabilité pour que le nœud n_i soit atteint par recommandation. $p_{j,i}$ représente le poids de l'arc allant du nœud n_j vers le nœud n_i, soit le nombre de co-occurrences entre les deux mots i et j

Dans leurs travaux, [118] s'intéressent à l'ajout d'informations dans le graphe grâce à des documents similaires (voisins) et aux relations de co-occurrences qu'ils possèdent (*ExpandRank*). L'objectif est de faire mieux ressortir les mots importants du graphe en ajoutant de nouveaux liens de recommandation ou bien en renforçant ceux qui existent déjà. L'usage de documents similaires peut cependant ajouter ou renforcer des liens qui ne devraient pas l'être. Pour éviter cela, les auteurs réduisent l'impact des documents voisins en utilisant leur degré de similarité avec le document analysé. Une alternative à ExpandRank, *CollabRank*, également proposée par Wan et Xiao [117], fonctionne de la même manière, mais certains choix des auteurs rendent impossible l'usage du degré de similarité pour réduire l'impact des documents voisins. Les résultats moins concluants de CollabRank tendent à confirmer l'importance de l'usage du degré de similarité.

Dans l'optique d'améliorer encore TextRank/SingleRank, [68] proposent une méthode qui cherche cette fois-ci à augmenter la couverture de l'ensemble des termes-clés extraits dans le document analysé (*TopicalPageRank*). Pour ce faire, ils tentent d'affiner le rang d'importance des mots dans le document en tenant compte de leur rang dans chaque sujet abordé. Le rang d'un mot pour un sujet est obtenu en intégrant à son score PageRank la probabilité qu'il appartienne au sujet 1.2. Le rang global d'un terme candidat est ensuite obtenu en fusionnant ses rangs pour chaque sujet.

$$S_{\text{Sujet}}(N_i) = (1 - \lambda)p(\text{Sujet}|i) + \lambda \times \sum_{N_j \in A_{\text{In}}(N_i)} \frac{p_{j,i} \times S(N_j)}{\sum_{N_k \in A_{\text{Out}}(N_j)} p_{j,k}} \tag{1.2}$$

Les approches à bases de graphe présentées ci-dessus effectuent toutes un ordonnancement des mots du document analysé selon leur importance dans celui-ci. Pour extraire les termes-clés il est donc nécessaire d'effectuer du travail supplémentaire à partir de la liste ordonnée de mots. Dans la méthode TextRank, les k mots les plus importants sont sélectionnés et retournés. La technique utilisée dans les autres méthodes consiste à ordonner les termes candidats en fonction de la somme du score des mots qui les composent.

1.8.2 Les méthodes supervisées

Les méthodes supervisées sont des méthodes capables d'apprendre à réaliser une tâche particulière, soit ici l'extraction de termes-clés. L'apprentissage se fait grâce à un corpus dont les documents sont annotés en termes-clés. L'annotation permet d'extraire les exemples et les contres exemples dont les descripteurs statistiques et/ou linguistiques servent à apprendre une classification binaire. La classification binaire consiste à indiquer si un terme candidat est un terme-clé ou non. De nombreux algorithmes d'apprentissage sont utilisés dans divers domaines. Ils peuvent potentiellement s'adapter à n'importe quelle tâche, dont celle de l'extraction automatique de termes-clés. Les algorithmes d'apprentissage utilisés sont les méthodes bayésiennes, les arbres de décision, les Séparateurs à Vaste Marge (SVM) ou encore des réseaux de neurones. Les méthodes d'apprentissages font l'objet du chapitre 3, nous décrirons seulement leur usage dans cette section.

KEA [125] est une méthode qui utilise une classification naïve bayésienne pour attribuer un score de vraisemblance à chaque terme candidat, le but étant d'indiquer s'ils sont des termes-clés ou non. [125] utilisent trois distributions conditionnelles apprises à partir du corpus d'apprentissage. La première correspond à la probabilité pour que chaque terme candidat soit étiqueté oui (terme-clé) ou non (non terme-clé). Les deux autres correspondent à deux différents descripteurs qui sont le poids TFIDF du terme candidat et sa première position dans le document.

L'un des avantages de la classification naïve bayésienne est que chaque distribution est supposée indépendante. L'ajout de nouveaux descripteurs dans la méthode KEA est donc très aisé. Parmi les variantes de KEA proposées, [27] ajoutent un troisième descripteur : le nombre de fois que le terme candidat est un terme-clé dans le corpus d'apprentissage. L'ajout de ce descripteur permet d'améliorer les performances de la version originale de KEA, mais uniquement lorsque la quantité de données d'apprentissage est très importante. Une autre amélioration de KEA, proposée par [114], tente d'augmenter la cohérence entre les termes candidats les mieux classés. Pour cela, une première étape de classification est effectuée avec la méthode originale. Cette première étape permet d'obtenir un premier classement des termes candidats selon leur score de vraisemblance. Ensuite, de nouveaux descripteurs sont ajoutés et une nouvelle étape de classification est lancée. Les nouveaux descripteurs ont pour but d'augmenter le score de vraisemblance des termes candidats ayant un fort lien sémantique avec certains des termes les mieux classés après la première étape. Enfin, [82] proposent l'ajout des informations concernant la structure des documents. En effet, certaines sections telles que l'introduction et la conclusion dans les articles scientifiques sont

plus susceptibles de contenir des termes-clés qu'une section présentant des résultats expérimentaux, par exemple. Dans leur version modifiée de KEA, ils proposent aussi l'usage de descripteurs linguistiques tels que les étiquettes syntaxiques qui ont prouvé jouer un rôle non-négligeable pour l'extraction de termes-clés [45].

En même temps que KEA [125], [112] met au point l'algorithme génétique GenEx. GenEx est constitué de deux composants. Le premier composant, le géniteur, sert à apprendre des paramètres lors de la phase d'apprentissage. Ces paramètres sont utilisés par le second composant, l'extracteur, pour donner un score d'importance à chaque terme candidat. Plus les paramètres sont optimaux, meilleure est la classification des termes. Pour ce faire, les paramètres sont représentés sous la forme de bits qui constituent une population d'individus que le géniteur fait évoluer jusqu'à obtenir un état stable correspondant aux paramètres optimaux. Dans son article présentant GenEx, [112] discute une autre méthode pour l'extraction de termes-clés. Cette méthode utilise de nombreux descripteurs qui servent à entraîner une forêt de 50 arbres décisionnels de type C4.5 (forêts aléatoires). Dans un arbre de décision, chaque branche représente un test sur l'un des descripteurs d'un terme candidat. Les tests permettent un routage du terme candidat vers la feuille de l'arbre qui détermine sa classe. Grâce à la technique des forêts aléatoires l'extraction automatique de termes-clés est réduite à un vote de chaque arbre pour chaque terme candidat. Cela permet un classement des termes candidats en fonction de leur nombre de votes positifs. Les termes-clés extraits correspondent aux termes candidats les mieux classés.

Les Séparateurs à Vaste Marge (SVM) sont aussi des classifieurs utilisés par les méthodes d'extraction automatique de termes-clés. Ils exploitent divers descripteurs afin de projeter des exemples et des contres-exemples sur un plan, puis ils cherchent l'hyperplan qui les sépare. Cet hyperplan sert ensuite dans l'analyse de nouvelles données. Dans le contexte de l'extraction de termes-clés, les exemples sont les termes-clés et les contres-exemples sont les termes candidats qui ne sont pas des termes-clés. Ce mode de fonctionnement des SVM est utilisé par [122], mais un autre type de SVM est plus largement utilisé dans les méthodes supervisées d'extraction de termes-clés. Il s'agit de SVM qui utilisent de multiples marges représentant des rangs. Ces classifieurs permettent donc d'ordonnancer les termes-clés lors de leur extraction [49]. La méthode KeyWE de [19] utilise ce type de SVM avec le descripteur TFIDF ainsi qu'un descripteur booléen ayant la valeur vraie si le terme candidat apparaît dans un titre d'un article Wikipedia (un terme candidat apparaissant dans le titre d'un article de Wikipedia a une plus forte probabilité d'être un terme-clé). L'ordonnancement des termes candidats par le SVM permet ensuite de contrôler le nombre de termes-clés à extraire (choix

des k termes candidats les mieux classés).

Tout comme [112], [21] utilisent eux aussi une forêt d'arbres C4.5 dans leur méthode d'extraction de termes-clés. Ils utilisent des descripteurs classiques et leur contribution se situe au niveau de l'utilisation d'un descripteur calculé à partir de chaînes lexicales. Une chaîne lexicale lie les mots d'un document selon certaines relations telles que la synonymie, l'hyponymie ou la méronymie. Ces relations permettent de calculer un score qui sert de descripteur. Cette approche est intéressante, mais du fait de limitations des chaînes lexicales actuellement disponibles elle présente l'inconvénient de ne retourner que des mots (aucun multi-mot). Cependant, l'usage d'une forêt d'arbre C4.5 permet un classement des mots à partir de leur nombre de votes positifs. Il est donc envisageable de déduire les termes-clés à partir de la liste ordonnée et pondérée des mots clés (voir les méthodes non-supervisées à bases de graphe – section1.8.1. Une autre méthode pour l'extraction automatique de termes-clés consiste à utiliser un perceptron multi-couches [98]. Un perceptron multi-couches est un réseau de neurones constitué d'au moins trois couches, chaque couche étant composée de neurones. Dans les deux couches extrêmes les neurones représentent respectivement les entrées et les sorties. Les couches centrales sont des couches cachées qui permettent d'acheminer les valeurs des entrées vers les sorties, où de nouvelles valeurs sont obtenues grâce à la pondération des transitions d'un neurone d'une couche vers un neurone de la couche suivante. Les entrées correspondent aux descripteurs d'un terme candidat (ici TFIDF, la position, la taille, etc.) et les sorties représentent les classes qu'il peut prendre (terme-clé ou non terme-clé). La valeur obtenue pour chaque sortie (classe) permet d'obtenir une probabilité pour que le terme candidat analysé soit un terme-clé ou non. Dans leur méthode, [98] utilisent cette probabilité pour ordonner les termes candidats afin de mieux contrôler le nombre de termes-clés à extraire.

1.9 Conclusion

Une brève présentation du problème d'extraction de termes-clés a été donnée tout au long de ce chapitre. On y décrit le principe des méthodes d'extraction et on y présente les différentes méthodes et outils utilisés dans la littérature pour la construction de tels systèmes.

Dans le chapitre qui suit, Le système d'extraction de termes-clés qui a été proposé sera exposé. En effet, nous avons développé un nouveau système basé sur l'apprentissage supervisé qui combine 18 descripteurs statistiques parmi lesquels le DPM-index une nouvelle mesure qui discrimine les phrases qui se chevauchent dans un document.

Chapitre 2

Système d'extraction de termes-clés basé sur la discrimination des chevauchements de phrases

Sommaire

2.1 Introduction

Dans n'importe quel problème d'extraction dans un texte, nous devons choisir entre améliorer la spécificité d'un n-gramme en étendant sa taille n, ce qui réduira sa fréquence, ou bien améliorer sa signification statistique en réduisant sa taille et ainsi réduire la spécificité du document considéré. Choisir parmi les n-grammes de différentes tailles qui se chevauchent nécessite un descripteur de saillance afin d'en extraire les termes significatifs qui sont spécifiques au document considéré. Par exemple, le 3-gramme *multiple sequence alignment* contient deux 2-grammes *multiple sequence* et *sequence alignment*. Ces trois n-grammes peuvent avoir des valeurs de TFIDF proches, comment choisir parmi celui qui est vraiment le terme-clé ? Intuitivement, *multiple sequence* est un mauvais candidat, mais *sequence alignment* ou *multiple sequence alignment* ou bien les deux peuvent être liés au sujet du document textuel. La comparaison des fréquences de ces n-grammes dans le document peut aider un système d'apprentissage à décider quels seront les termes-clés. Ceci est réalisé dans ce chapitre par l'indice de maximalité de phrase dans le document DPM-index (Document Phrase Maximality) qui donne un moyen de contrôler à la fois la taille et la fréquence d'un n-gramme, en préservant à la fois son importance et sa spécificité.

Des travaux récents dans le domaine de l'extraction de termes-clés ont porté principalement sur l'utilisation de connaissances externes (external knowledge information) ou l'information sur la structure du document. Notre objectif général dans ce chapitre est de revenir à l'extraction de la connaissance fondamentale, question abordée par les travaux pionniers dans le domaine qui est : comment extraire des termes-clés dans tout type de document texte sans connaissance *a priori* sur les termes-clés ? A cet effet, nous avons développé un système d'apprentissage supervisé qui utilise 18 descripteurs statistiques, parmi eux le DPM-index et cinq autres nouvelles métriques. Après la sélection des termes-clés candidats à l'aide d'informations linguistiques (patrons syntaxiques), le système offre la possibilité d'évaluer l'efficacité de la contribution de chaque descripteur statistique.

Nous avons expérimentalement comparé nos résultats à ceux de 21 méthodes d'extraction de termes-clés sur le corpus d'articles scientifiques SemEval-2010/Task-5. Lorsque tous les systèmes extraient 10 termes-clés par document, notre méthode augmente d'un taux de 13% le F-score du meilleur système. En particulier, le DPM-index augmente la précision de notre système d'extraction de termes-clés d'un taux de 9%.

Nous montrons également, avec ces données, qu'en combinant ces 18 descripteurs en utilisant n'importe quel paradigme supervisé (boosting, bagging et régression), notre système offre les meilleurs performances.

2.2 Nouvelles métriques

Des travaux récents dans le domaine d'extraction de termes-clés ont porté principalement sur l'utilisation des descripteurs basés sur l'utilisation de bases de connaissances externes ou sur l'information structurelle du document. Notre objectif est de revenir à l'extraction de la connaissance fondamentale, question abordée par les travaux pionniers de Turney et al. [112] et Frank et al. [27], qui consiste à extraire des termes-clés de tout type de document texte en utilisant les descripteurs statistiques sans se baser sur la connaissance *a priori* sur les termes-clés, contrairement à l'approche de classification de textes [1]. Certains descripteurs utilisés dans la littérature ont des définitions différentes qui ont des impacts différents sur la précision des termes-clés extraits. Afin de définir précisément chaque descripteur utilisé nous proposons d'utiliser les notations données dans le chapitre 1.

2.2.1 Maximalité de phrase dans le document (DPM-index)

Dans la littérature relative à l'extraction de termes-clés on ne tient pas compte du fait qu'un terme candidat peut contenir ou être contenu dans un autre terme candidat. Cette relation entre n-grammes peut être exprimée par un descripteur qui caractérise les vrais termes-clés parmi les différents sous-termes, qui se chevauchent, dans un même n-gramme. Par exemple, le 3-gramme *multiple sequence alignment* contient deux 2-grammes *multiple sequence* et *sequence alignment*. Ces trois n-grammes pourraient avoir des valeurs de TFIDF proches, comment choisir les vrais termes-clés parmi eux ? Intuitivement, *multiple sequence* est un mauvais candidat, mais *sequence alignment* ou *multiple sequence alignment* ou les deux pourraient être dans les principaux sujets du document. La comparaison des fréquences de ces n-grammes dans le document considéré peut aider un système d'apprentissage à décider lesquels d'entre eux sont des termes-clés. Le terme *multiple sequence* par lui-même n'est pas un terme-clé, car il n'est pas utilisé en dehors de *multiple sequence alignment*. Cela signifie que la fréquence de *multiple sequence* est égale à la fréquence de *multiple sequence alignment*. On peut donc supposer que si un terme t est contenu dans un superterme s et que $f(t,d) = f(s,d)$, i.e., $(\frac{f(s,d)}{f(t,d)} = 1)$, alors t ne peut pas être un terme-clé.

Maintenant décider si t = *sequence alignment* est un terme-clé ou non, dépendra de sa fréquence par rapport à la fréquence de s = *multiple sequence alignment*. Prenons, par exemple, un document d où *multiple sequence alignment* apparait 10 fois. Si *sequence alignment* apparait 100 fois $(\frac{f(s,d)}{f(t,d)} = 0,1)$, il est plus susceptible d'être un terme-clé que s'il apparait

11 fois ($\frac{f(s,d)}{f(t,d)} = 0,91$). Plus le 2-gramme *sequence alignment* est utilisé en dehors de *multiple sequence alignment*, plus il est susceptible d'être dans les principaux thèmes du document considéré. Au contraire, quand la fréquence de *sequence alignment* est proche de celle de *multiple sequence alignment*, cela signifie que le document traite spécifiquement de la question de *multiple sequence alignment*. Supposons maintenant que nous avons trois documents d_1, d_2 et d_3 avec $t =$ *sequence alignment* contenu dans les super-termes :

- $s_1 =$ *multiple sequence alignment*
- $s_2 =$ *pairwise sequence alignment*
- $x = w$ *sequence alignment*, avec w un mot différent de *multiple* et *pairwise*.

Les fréquences de t et ses super-termes sont données dans le tableau 2.1. On peut observer que plus le maximum du rapport $\frac{f(s,d)}{f(t,d)}$ est proche de 1, plus t n'est probablement pas un terme-clé. Nous définissons l'indice de maximalité de phrase dans les documents d'un terme t comme suit :

$$\text{DPM-index}(t,d) = 1 - \max_{s \in \sup(t,d)} \left(\frac{f(s,d)}{f(t,d)}\right) \tag{2.1}$$

Tel que nous l'avons défini, le DPM-index (t,d) tend vers 1 lorsque t n'est pas inclu dans un super-terme qui a une fréquence similaire dans le document d. À l'inverse, il tend vers 0 lorsque t est presque toujours une partie d'un autre terme de d, ce qui indique qu'il n'est probablement pas un terme-clé pertinent pour ce document. Par le biais de la métrique DPM-index, nous pouvons voir immédiatement sur l'exemple du tableau 2 que le document d_1 ne traite pas le thème de $t =$ *sequence alignment* qui a un DPM-index de $0,09$, mais plutôt du thème $s_1 =$ *multiple sequence alignment*. *A contrario*, dans le document d_3 son DPM-index vaut $0,91$. Le document d_2 représente une situation intermédiaire où le DPM-index qui vaut $0,45$ indique que *sequence alignment* est plus susceptible d'être dans les principaux sujets de d_2 plutôt que dans ceux de d_1. Ces exemples de chevauchements potentiels entre syntagmes nominaux candidats *multiple sequence alignment*, *multiple sequence* et *sequence alignment* peut survenir dans n'importe quel système d'extraction où nous avons à choisir entre étendre la taille n d'un n-gramme et réduire sa fréquence, ou au contraire, de réduire n et d'augmenter sa fréquence. le DPM-index donne un moyen de contrôler à la fois la taille et la fréquence d'un n-gramme en préservant sa signification et sa spécificité dans le document d. Le DPM-index d'un terme dans un document ressemble à première vue à la C-value d'un terme dans un corpus [28], qui est :

$$\text{C-value}(t,D) = \log |t| f(t,D) - s \in \sup(t,D) f(s,D) |\sup(t,D)| \tag{2.2}$$

La métrique C-value est connue pour être un descripteur essentiel dans l'assignation de terminologie. Il pourrait être adapté à l'extraction de termes-clés d'un document en remplaçant la fréquence du document dans le corpus $f(s, D)$ par la fréquence dans le document $f(s, d)$. Nous désignons cette adaptation par docC-value :

$$\text{docC-value}(t, d) = \log |t| f(t, d) - s \in \sup(t, d) f(s, d) |\sup(t, d)| \qquad (2.3)$$

Le DPM-index diffère complètement du docC-value car il utilise le maximum des fréquences des super-termes plutôt que leur moyenne comme c'est le cas dans le docC-value. La moyenne n'est pas efficace pour l'extraction des termes-clés car elle ne relève pas le fait qu'un terme t est presque toujours contenu dans un super-terme particulier.

doc.	f(t, d)	f(s_1, d)	$\frac{\text{f}(s_1,d)}{\text{f}(t,d)}$	f(s_2, d)	$\frac{\text{f}(s_2,d)}{\text{f}(t,d)}$	f(x, d)	$\frac{\text{f}(x,d)}{\text{f}(t,d)}$	DPM-index(t, d)
d_1	11	10	0.91	1	0.09	0	0	**0.09**
d_2	11	6	0.55	5	0.45	0	0	**0.45**
d_3	11	1	0.09	1	0.09	1 $(\forall x)$	0.09	**0.91**

TABLE 2.1 – Le DPM-index du terme t = *sequence alignment* dans trois documents

Dans le tableau 2.1, si on considère d_1 et d_2, la moyenne des fréquences des super-termes de t est la même 5.5, alors que leur maximum est de 10 pour d_1 et 6 pour d_2. Cela conduit à un DPM-index de 0,09 pour d_1 et 0,45 pour d_2, tandis que leur valeur docC-value est identique. Ainsi, déterminer si t est un terme-clé ou non ne dépend pas de la moyenne des fréquences des super-termes mais de leur maximale. Cela a été confirmé par nos expérimentations, les métriques C-value et docC-value n'améliorent pas la précision de l'extraction de termes-clés.

2.2.2 K-moyennes

La plupart des systèmes d'extraction de termes-clés considèrent la première position d'un terme candidat comme descripteur. La position de la première occurrence d'un terme est connue pour être un descripteur très utile à la prédiction de terme-clé. Dans notre cas, nous voulons tenir compte de la distribution de toutes ses positions dans le document. Nous supposons que les positions des termes-clés dans le document sont distribuées différemment des autres termes. Ainsi, nous proposons d'utiliser comme descripteurs la moyenne de ces positions et leurs 2-moyennes.

Nous rappelons que les k-moyennes est le fait de partitionner les données en k groupes (clusters) de telle sorte que la somme des distances euclidiennes carrées de chaque moyenne du groupe soit minimisée. Ce problème est souvent résolu par une heuristique car il est connu pour être NP-complet en général, même quand $k = 2$ ou lorsque la dimension est égal à 2 [120]. Cependant, dans notre cas, les n positions d'un terme sont définies dans une dimension pour laquelle il existe un algorithme de programmation dynamique dont la complexité est $O(kn^2)$ en temps qui garantit l'optimalité [120].

Formellement les k-moyennes μ_m $(m = 1, \ldots, k)$ des positions normalisées $x_i = \text{npos}_i(t, d)$ (i=1,. . .,n) d'un terme t dans le document d sont telles qu'il existe une partition de ces positions dans k clusters E_m $(m = 1, \ldots, k)$ où $\sum_m \sum_{x_i \in E_m} (x_i - \mu_m)^2$ est minimale.

2.2.3 TFIDF ratio

Quand un terme-clé est un n-gramme qui contient plus d'un composant, il est fréquent que l'un d'eux soit un mot spécifique au document. Dans [129] ce composant est appelé *mot noyau* et est défini simplement comme un mot avec une fréquence supérieure à un certain seuil. Dans notre cas, nous n'avons pas utilisé un seuil fixe et plutôt que d'utiliser TF pour caractériser ce composant nous utilisons TFIDF. On définit le descripteur TFIDFRatio(t, d, D) comme étant le rapport entre le TFIDF d'un terme t et la valeur maximale de la TFIDF d'un composant de t. Cet indicateur tend à être faible lorsque le terme possède un composant avec un TFIDF élevé.

$$\text{TFIDFRatio}(t, d, D) = \frac{\text{TFIDF}_1(t, d, D)}{\max\limits_{c \in \text{comp}(t)} (\text{TFIDF}_1(c, d, D))}$$

2.2.4 DPM-TFIDF

Dans nos expérimentations, nous avons remarqué que la combinaison de la métrique DPM-index avec TFIDF améliore la qualité des résultats de 10 à 15 % lorsque nous l'utilisons comme un score non supervisé pour l'extraction automatique de termes-clés.

$$\text{DPM} - \text{TFIDF}(t, d, D) = \text{DPM} - \text{index}(t, d) \times \text{TFIDF}(t, d, D)$$

2.3 Classifieur

Notre système d'extraction de termes-clés utilise un classifieur basé sur un algorithme d'apprentissage supervisé. En raison de la difficulté du problème d'extraction de termes-clés, plutôt que d'utiliser directement les sorties du classifieur, on utilise la probabilité qu'un terme soit classé comme un terme-clé [125]. Ces probabilités sont utilisées comme des scores pour générer une liste ordonnée de termes-clés. Selon un paramètre fixe k, le système affiche les k-meilleurs termes comme étant les termes-clés prédits. Nous utilisons la régression logistique comme algorithme d'apprentissage. Nous avons aussi essayé d'autres algorithmes d'apprentissage, par exemple les arbres de décision C4.5, les forêts aléatoires et LogitBoost mais dans tous les cas, c'est la régression logistique qui donne de bons résultats. Nous avons utilisé Weka [39] pour la mise-en-œuvre de ces méthodes. La précision du système dépend fortement de la manière dont les 18 descripteurs sont transformés en entrées pour l'algorithme de régression logistique. Après, avoir essayé différentes méthodes d'extraction de données, nous avons découvert que les meilleurs résultats sont obtenus après discrétisation et binarisation des descripteurs. Pour expliquer cela, nous devons rappeler quelques notions liées à la régression logistique. Un classifieur est en mesure d'estimer la probabilité $P(Y = C|X)$ d'une instance représentée par son vecteur de descripteurs X à appartenir à la classe $Y = C$. Cette estimation consiste à entrainer l'algorithme d'apprentissage sur un ensemble d'instances (X_i, Y_i) où la classe Y_i de chaque instance est connue. Suite à cette étape, le modèle obtenu peut prédire $P(Y = C|X)$ pour tout X donné. Dans notre cas, c'est égal à 1 si le terme est un terme-clé et 0 sinon, le vecteur de descripteurs X représente un terme candidat et $P(Y = 1|X)$ prédit la probabilité qu'un terme candidat est un terme-clé ou pas. La régression logistique tente d'estimer $P(Y = 1|X)$ compte tenu de cette hypothèse :

$$\log \frac{P(Y_i = 1|X_i)}{1 - P(Y_i = 1|X_i)} = \alpha_0 + \sum_{j=1}^{m} \alpha_j x_{ij} = z \tag{2.4}$$

où $X_i = (x_{i1}, x_{i2}, \ldots, x_{im})$ est le vecteur des valeurs des m descripteurs de l'instance d'apprentissage ce qui signifie que pour le modèle de régression logistique, nous avons :

$$P(Y_i = 1|X_i) = \frac{1}{1 + e^{-z}} \tag{2.5}$$

Les valeurs des paramètres α_j $(j = 0, \ldots, m)$ qui correspondent le mieux avec les données d'apprentissage sont estimées par un algorithme d'optimi-

sation, un algorithme de descente de gradient stochastique par exemple.

Étant donné un terme t_i, nous nous intéressons à l'estimation $p_i = P(Y_i = 1|X_i)$ la probabilité que t_i soit un terme-clé. Tout d'abord, nous considérons que les descripteurs sont donnés à l'algorithme sans prétraitement ce qui signifie que les $X_i = (x_{i1}, x_{i2}, ..., x_{im})$ sont exactement les valeurs des $m = 18$ descripteurs de t_i. Dans ce cas p_i dépend d'un produit linéaire des $\alpha_j x_{ij}$ de chaque valeur du descripteur x_{ij} et son paramètre α_j selon l'équation (2.4). Toutefois, si l'on considère que le descripteur j est TFIDF par exemple, quelle que soit α_j, il est peu probable que la fonction (2.4) soit linéaire dans $x_{ij} = \text{TFIDF}(t_i, d)$.

Afin de faire face à la non-linéarité des effets des descripteurs sur la probabilité des termes-clés, nous proposons de discrétiser [24] chaque descripteur et binariser les intervalles résultants de la discrétisation. Précisément, la fonction logit devient :

$$\log \frac{P(Y_i = 1|X_i)}{1 - P(Y_i = 1|X_i)} = \alpha_0 + \sum_{j=1}^{m} \sum_{k=1}^{n_j} \alpha_{jk} I(x_{ij} \in \mathcal{I}_{jk}) \tag{2.6}$$

Où $\mathcal{I}_{jk}$ est le k-ième intervalle résultant de la discrétisation du descripteur j, n_j le nombre de ces intervalles et $I(B) = 1$ si B est vrai, 0 sinon.

En remplaçant $\alpha_j x_{ij}$ par $\sum_{k=1}^{n_j} \alpha_{jk} I(x_{ij} \in \mathcal{I}_{jk})$ dans la fonction logit nous ne supposons plus la linéarité des effets des descripteurs sur la probabilité des termes-clés. Chaque α_j est substitué par les paramètres α_{jk} $(k = 1, \ldots, n_j)$ qui sont estimés par régression logistique. Ainsi, la fonction logit de l'équation (2.6) dépend de n'importe quelle fonction de chaque descripteur j qui est approximée par α_{jk} dans chaque intervalle $\mathcal{I}_{jk}$.

2.4 Tests et résultats

2.4.1 Benchmark

Afin d'évaluer notre système, nous avons utilisé les données du corpus SemEval-2010/task5 : Extraction automatique de termes-clés dans des articles scientifiques [59]. Ces données consistent en 244 documents de la "ACM Digital Library". Les documents ont été sélectionnés à partir de quatre domaines de recherche : systèmes distribués, recherche et extraction d'information, apprentissage et sciences sociales et comportementales. Pour chaque article au plus 15 termes-clés ont été attribués manuellement par les auteurs et les lecteurs des articles. 144 documents sont utilisés pour l'apprentissage et l'évaluation se fait sur 100 articles. Le principal avantage de l'utilisation du corpus SemEval-2010/task-5 est que nous pouvons comparer nos résultats

		terme-clé réel	
		oui	non
terme-clé prédit	oui	**VP** (vrai positif)	**FP** (faux positif)
	non	**VN** (vrai negatif)	**FN** (faux negatif)

TABLE 2.2 – Définitions des termes vrais/faux positifs/négatifs.

à ceux obtenus par 19 équipes qui ont participé au défi. En outre, deux études récentes [80, 129] ont également utilisé ces données. Nous avons suivi la procédure indiquée dans le défi pour l'évaluation de notre système. Dans cette tâche, les méthodes ont été comparées en utilisant trois paramètres d'évaluation de correspondance exacte. Une métrique d'évaluation de correspondance exacte mesure combien les termes-clés générés automatiquement correspondent exactement à ceux attribués manuellement. Des métriques plus souples pourraient être utilisées [57], cependant la correspondance exacte est plus stricte et nous permet de comparer nos résultats avec ceux des 21 équipes. Plus précisément, les trois indicateurs utilisés sont : la précision qui représente la proportion de termes-clés extraits qui correspondent à ceux attribués manuellement, le rappel qui est la proportion de termes-clés attribués manuellement qui sont extraits par le système et le F_1-score est défini comme suit :

$$F_1\text{-score} = 2 \times \frac{\text{précision} \times \text{rappel}}{\text{précision} + \text{rappel}}$$

La précision et le rappel sont calculés en comptant le nombre de termes vrais/faux positifs/négatifs trouvés (voir la table 2.2 et l'équation 2.7).

$$\textbf{précision} = \frac{\textbf{TP}}{\textbf{TP+FP}} \ \textbf{rappel} = \frac{\textbf{TP}}{\textbf{TP+FN}} \tag{2.7}$$

2.4.2 Résultats

Nous avons utilisé les données SemEval-201/task-5 dans deux expérimentations.

— Tout d'abord, nous avons suivi la procédure du défi afin de comparer nos résultats à ceux des autres systèmes. Dans cette procédure, 144 documents sont utilisés pour l'apprentissage et 100 documents pour l'évaluation.

— Ensuite, nous avons regroupé les 244 documents et nous les avons utilisé pour la validation croisée afin de confirmer les résultats.

a- Procédure du Challenge

Les performances de chaque système sont données pour différents nombres de termes-clés candidats : 5, 10 et 15. Le tableau 2.3 montre les performances de chaque système, il est ordonné selon l'ordre décroissant des F_1-scores par rapport au top 15 des termes-clés. Notre système occupe le premier rang dans les trois cas de nombre de termes-clés candidats (5, 10, 15) et pour les trois métriques utilisées.

Pour 10 termes-clés, notre système augmente d'un taux de 13% ($\frac{29,4\%}{26,0\%} - 1$) pour le F_1-score de Humb [71]. Notez que contrairement à notre système, Humb utilise des descripteurs structuraux et des descripteurs de différentes bases de connaissances externes afin d'améliorer ses performances. Ces bases de connaissances (GROBID / TEI, GRiSP et HAL) sont spécifiques aux articles scientifiques. Enfin, le plus important de ces résultats est qu'en utilisant uniquement des descripteurs statistiques sur des termes linguistiquement filtrés, notre système surpasse les autres, sans perte en généralité.

Afin de mesurer la contribution de chaque descripteur par rapport aux performances globales du système, nous représentons dans le tableau 2.4 les résultats obtenus par notre méthode lorsqu'on élimine chaque descripteur parmi les descripteurs d'apprentissage. Selon ces expérimentations, nous pouvons voir que le DPM-index est le plus important des descripteurs proposés, le retirer diminue de façon significative la performance de notre système. Lorsque DPM-index est utilisé, le F_1-score du système d'extraction de termes-clés augmente d'un taux de 9% (pour le top 10 des termes-clés), ce qui rend sa contribution comparable à celle de tous les descripteurs liés au niveau corpus, y compris celle du populaire TFIDF. Notez aussi que l'utilisation de nos 6 descripteurs proposés augmente le F_1-score par un taux de 13,5% pour le top 10 des termes-clés. Nous avons également testé notre système en utilisant différents algorithmes d'apprentissage. Le but était, bien sûr, de trouver le meilleur classifieur. Comme on peut le constater dans le tableau 2.5, combiner les 18 descripteurs en utilisant n'importe quel paradigme d'apprentissage (boosting, bagging et régression) donnent les meilleures performances confirmant que nos résultats ne dépendent pas d'une approche spécifique de l'apprentissage.

b- La validation croisée

Nous avons utilisé la validation croisée afin de confirmer la stabilité et la robustesse des descripteurs. Les 244 documents de l'ensemble du corpus sont divisés au hasard en 5 parties de 48 documents (reste 4 documents). Le test est réalisé sur chaque sous-ensemble et le reste est utilisé pour l'apprentis-

sage. Les résultats sont en moyenne pour toutes les exécutions. Le tableau 2.6 présente les résultats de la validation croisée obtenus par notre méthode lorsqu'on élimine chaque descripteur parmi les descripteurs d'apprentissage. Ces résultats confirment que le DPM-index améliore l'extraction de termes-clés (de 9,6%). Nos trois autres métriques ont moins d'impact sur le F_1-score quand elles sont utilisées individuellement, mais ensemble avec le DPM-index elles améliorent significativement les performances. Le tableau 2.7 donne les résultats que nous avons obtenus avec les différents algorithmes d'apprentissage. La validation croisée avec LogitBoost et la régression logistique donnent des résultats similaires en moyenne.

Système	Candidats top 5			Candidats top 10			Candidats top 15		
	P	R	F	P	R	F	P	R	F
Notre système	**44.8%**	**15.3%**	**22.8%**	**36.2%**	**24.7%**	**29.4%**	**28.3%**	**28.9%**	**28.6%**
HUMB	39.0%	13.3%	19.8%	32.0%	21.8%	26.0%	27.2%	27.8%	27.5%
[129]	-	-	-	-	-	-	26.2%	26.8%	26.0%
WINGNUS	40.2%	13.7%	20.5%	30.5%	20.8%	24.7%	24.9%	25.5%	25.2%
KP-Miner	36.0%	12.3%	18.3%	28.6%	19.5%	23.2%	24.9%	25.5%	25.2%
SZTERGAK	34.2%	11.7%	17.4%	28.5%	19.4%	23.1%	24.8%	25.4%	25.1%
ICL	34.4%	11.7%	17.5%	29.2%	19.9%	23.7%	24.6%	25.2%	24.9%
SEERLAB	39.0%	13.3%	19.8%	29.7%	20.3%	24.1%	24.1%	24.6%	24.3%
KX_FBK	34.2%	11.7%	17.4%	27.0%	18.4%	21.9%	23.6%	24.2%	23.9%
DERIUNLP	27.4%	9.4%	13.9%	23.0%	15.7%	18.7%	22.0%	22.5%	22.3%
Maui	35.0%	11.9%	17.8%	25.2%	17.2%	20.4%	20.3%	20.8%	20.6%
DFKI	29.2%	10.0%	14.9%	23.3%	15.9%	18.9%	20.3%	20.7%	20.5%
DP-Seg [80]	33.3%	10.5%	16.0%	23.7%	16.9%	19.7%	19.2%	21.3%	20.2%
BUAP	13.6%	4.6%	6.9%	17.6%	12.0%	14.3%	19.0%	19.4%	19.2%
SJTULTLAB	30.2%	10.3%	15.4%	22.7%	15.5%	18.4%	18.4%	18.8%	18.6%
UNICE	27.4%	9.4%	13.9%	22.4%	15.3%	18.2%	18.3%	18.8%	18.5%
UNPMC	18.0%	6.1%	9.2%	19.0%	13.0%	15.4%	18.1%	18.6%	18.3%
JU_CSE	28.4%	9.7%	14.5%	21.5%	14.7%	17.4%	17.8%	18.2%	18.0%
LIKEY	29.2%	10.0%	14.9%	21.1%	14.4%	17.1%	16.3%	16.7%	16.5%
UvT	24.8%	8.5%	12.6%	18.6%	12.7%	15.1%	14.6%	14.9%	14.8%
POLYU	15.6%	5.3%	7.9%	14.6%	10.0%	11.8%	13.9%	14.2%	14.0%
UKP	9.4%	3.2%	4.8%	5.9%	4.0%	4.8%	5.3%	5.4%	5.3%

TABLE 2.3 – Notre système comparé à 21 systèmes par rapport à la précision (P), le rappel (R) et le F_1-Score (F)

Descripteurs	Candidats top 5			Candidats top 10			Candidats top 15		
	P	R	F	P	R	F	P	R	F
Tous les desc.	44.8%	15.3%	22.8%	**36.2%**	**24.7%**	**29.4%**	**28.3%**	**28.9%**	**28.6%**
Tous - TFIDF ratio	43.2%	14.7%	22.0%	35.5%	24.2%	28.8%	27.7%	28.3%	28.0%
Tous - DPM-TFIDF (*)	**45.2**%	**15.4**%	**23.0**%	35.2%	24.0%	28.5%	27.7%	28.3%	28.0%
Tous - k-moyennes	44.6%	14.2%	22.7%	35.9%	24.5%	29.1%	27.4%	28.0%	27.7%
HUMB	*39.0%*	*13.3%*	*19.8%*	*32.0%*	*21.8%*	*26.0%*	*27.2%*	*27.8%*	*27.5%*
Tous - DPM-index (**)	39.0%	13.3%	19.8%	33.2%	22.6%	26.9%	26.5%	27.1%	26.8%
Tous - Desc. niv. corpus (***)	43.2 %	14.7%	22.0%	32.8%	22.4%	26.6%	26.0%	26.6%	26.3%
Tous - Nos desc.	38.6%	13.2%	19.6%	32.0%	21.8%	25.9%	25.8%	26.4%	26.1%
WINGNUS	*40.2%*	*13.7%*	*20.5%*	*30.5%*	*20.8%*	*24.7%*	*24.9%*	*25.5%*	*25.2%*

(*) DPM-TFIDF est remplacé par TFIDF.
(**) On retire le DPM-index et le DPM-TFIDF est remplacé par TFIDF.
(***) On retire IDF, logTFIDF, KLD, DPM-TFIDF et TFIDFRatio.

TABLE 2.4 – Performances de notre système apres suppression d'un descripteur ou d'un groupe de descripteurs comparé aux systèmes HUMB et WINGNUS

Methodes d'apprentissage	Candidats top 5			Candidats top 10			Candidats top 15		
	P	R	F	P	R	F	P	R	F
Régression logistique	**44.8%**	**15.3%**	**22.8%**	**36.2%**	**24.7%**	**29.4%**	**28.3%**	**28.9%**	**28.6%**
LogitBoost	43.2%	14.7%	22.0%	34.8%	23.7%	28.2%	27.7%	28.4%	28.0%
Bagged C4.5	40.8%	13.9%	20.8%	33.2%	22.6%	26.9%	27.2%	27.8%	27.5%
HUMB	*39.0%*	*13.3%*	*19.8%*	*32.0%*	*21.8%*	*26.0%*	*27.2%*	*27.8%*	*27.5%*
Fôrets aléatoires	43.0%	14.7%	21.9%	32.6%	22.2%	26.4%	26.0%	26.6%	26.3%
WINGNUS	*40.2%*	*13.7%*	*20.5%*	*30.5%*	*20.8%*	*24.7%*	*24.9%*	*25.5%*	*25.2%*

TABLE 2.5 – Performances de notre système avec différentes méthodes d'apprentissage comparé aux systèmes HUMB et WINGNUS

2.5 Conclusion

Dans ce chapitre nous avons présenté une nouvelle mesure le DPM-index qui discrimine les n-grammes qui se chevauchent dans un document. Nous avons développé un nouveau système d'extraction de termes-clés basé sur l'apprentissage supervisé qui combine 18 descripteurs statistiques parmi eux le DPM-index. On a montré expérimentalement, que la contribution de cette dernière est comparables aux descripteurs liés au niveau corpus pour l'extraction de 10 termes-clés. Les résultats de ce système montrent une amélioration par rapport aux autres systèmes sans utiliser aucune base de connaissance externe ou descripteurs structuraux, ce qui nous semble rendre notre système plus souple et adaptable à d'autres types de corpus.

TABLE 2.6 – Performances de la validation croisée (moyenne et écart type) aprés suppression d'un descripteur ou d'un groupe de descripteurs

Features	Candidats top 5			Candidats top 10			Candidats top 15		
	P	R	F	P	R	F	P	R	F
Tous les descripteurs	46.7± 2.1%	15.5± 1.0%	23.3± 1.4%	35.9± 1.4%	23.8± 1.3%	28.6± 1.4%	28.5± 0.8%	28.4± 1.5%	28.4± 1.1%
Tous – TFIDF ratio	45.5± 3.1%	15.1± 1.4%	22.7± 1.9%	35.4± 1.3%	23.5± 1.4%	28.2± 1.4%	28.4± 0.9%	28.3± 1.5%	28.3± 1.2%
Tous - DPM-TFIDF	45.6± 2.1%	15.1± 1.0%	22.7± 1.4%	35.2± 1.1%	23.4± 1.3%	28.1± 1.3%	28.1± 0.7%	27.9± 1.2%	28.0± 0.9%
Tous - k-moyennes	45.6± 1.8%	15.1± 1.0%	22.7± 1.3%	35.5± 1.5%	23.5± 1.5%	28.3± 1.5%	28.0± 1.3%	27.9± 1.9%	27.9± 1.6%
Tous - DPM-index	43.9± 2.3%	14.6± 1.1%	21.9± 1.5%	32.7± 0.5%	21.7± 0.8%	26.1± 0.7%	26.4± 0.6%	26.3± 1.1%	26.4± 0.8%
Tous - Desc. niv. corpus.	44.3± 3.7%	14.7± 1.6%	22.1± 2.2%	33.3± 1.3%	22.1± 1.4%	26.6± 1.4%	26.5± 0.9%	26.4± 1.5%	26.4± 1.2%
Tous - Nos desc.	41.6± 1.8%	13.8± 1.0%	20.7± 1.3%	31.3± 0.7%	20.8± 1.0%	25.0± 0.9%	25.6± 1.5%	25.5± 2.0%	25.6± 1.7%

Methode d'apprentissage	Candidats top 5			Candidats top 10			Candidats top 15		
	P	R	F	P	R	F	P	R	F
Régression logistique	46.7± 2.1%	15.5± 1.0%	23.3± 1.4%	35.9± 1.4%	23.8± 1.3%	28.6± 1.4%	28.5± 0.8%	28.4± 1.5%	28.4± 1.1%
LogitBoost	45.5± 1.8%	15.1± 1.0%	22.7± 1.3%	36.1± 1.7%	24.0± 1.7%	28.8± 1.8%	28.6± 1.2%	28.5± 1.9%	28.5± 1.5%
Bagged C4.5	44.2± 2.3%	14.7± 1.0%	22.0± 1.4%	35.7± 1.5%	23.7± 1.3%	28.4± 1.3%	27.9± 0.9%	27.7± 1.3%	27.8± 1.1%
Fôrets aléatoires	46.6± 1.7%	15.5± 0.9%	23.2± 1.2%	35.2± 1.0%	23.3± 1.1%	28.0± 1.1%	27.4± 1.0%	27.3± 1.5%	27.3± 1.2%

TABLE 2.7 – Performances de la validation croisée (moyenne et écart type) avec différentes méthodes d'apprentissage

Chapitre 3

Méthodes d'apprentissage

Sommaire

3.1 Introduction

Dans le domaine de l'apprentissage automatique, différents types de méthodes ont été mises au point pour un grand nombre de problèmes. On peut distinguer les méthodes de classification et les méthodes de régression. Il sera question dans ce chapitre des algorithmes adaptés aux deux problèmes qui concernent cette thèse : la classification de textes et l'extraction de termes-clés qui sont tous les deux des problèmes de classification (de textes et de termes respectivement).

3.2 Fonction de classification

Les algorithmes d'apprentissage pour la classification divergent beaucoup dans leur méthodologie mais ils partagent toutefois des caractéristiques communes. La construction de la majorité d'entre eux comporte deux étapes principales.

— D'abord, la définition d'une fonction qui associe à une instance (document ou terme) une valeur représentant son degré d'appartenance à une classe. Cette fonction prend différentes formes selon le type de classifieur.
— Puis, un seuil qui va servir lors de la prise de décision finale, à savoir si oui ou non une instance va être acceptée ou rejetée de la classe. Si la fonction du classifieur retourne une valeur (score) supérieure au seuil pour une instance donnée, on décide alors de l'associer à la classe.

Plusieurs méthodes de détermination du seuil sont possibles et le choix de l'une d'entre elles peut influencer significativement la performance d'un classifieur. Le choix du seuil peut se faire, entre autres, selon les stratégies suivantes :

— Assigner à chaque instance les k classes pour lesquelles le classifieur lui aura donné les plus hauts scores ;
— associer à chaque classe les k meilleurs instances, c'est-à-dire les instances pour lesquels le classifieur aura donné les plus hauts scores pour la classe en question ;
— choisir un seuil différent pour chaque classe, pour maximiser la performance sur cette classe, évaluée à l'aide d'un ensemble de validation ;
— appliquer une méthode d'apprentissage qui apprend le nombre k de classes (variables) à associer à chaque instance.

3.3 Les k-voisins les plus proches (kNN)

L'algorithme des k-voisins les plus proches ("k-nearest neighbors" ou kNN) est une méthode d'apprentissage à base d'instances. Il ne comporte pas de phase d'apprentissage en tant que telle. Les instances faisant partie de l'ensemble d'apprentissage sont seulement indexées dans une structure de données permettant un accès rapide aux plus proches voisins. Lorsqu'une nouvelle instance à classer arrive, elle est comparée aux instances d'apprentissage à l'aide d'une mesure de similarité. Ses k plus proches voisins sont alors considérés : on observe leur classe et celle qui revient le plus parmi les voisins est assignée à l'instance nouvelle. C'est là une version de base de l'algorithme que l'on peut raffiner. Souvent, on pondère les voisins par la distance qui les sépare du nouveau texte. On accorde plus de poids, lors de la prise de décision, aux instances plus similaires.

La valeur de k est un des paramètres à déterminer lors de l'utilisation de ce type de classifieur. Souvent, on procède à une optimisation de ce paramètre à l'aide de tests sur une portion des instances. Il est à noter que si l'on ne pondère pas les voisins, la valeur que l'on choisit pour k va être plus critique, plus déterminante en rapport avec la performance du classifieur. Par contre, si l'on décide de pondérer les voisins, l'importance du paramètre k va être atténuée. On peut se permettre de considérer un plus grand nombre de voisins, sachant que plus ils diffèrent de l'instance à classer, moins ils ont d'impact sur la prise de décision. Cependant, il demeure nécessaire de limiter le nombre de voisins pour s'en tenir à un temps de calcul raisonnable.

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

D'autres façons de calculer la similarité des instances sont utilisées, comme la similarité cosinusoïdale, et sont préférables dans le cas de la classification de textes pour plusieurs raisons.

- Premièrement, la similarité cosinusoïdale permet de comparer des textes de longueurs différentes en normalisant leur vecteur.
- De plus, elle met l'accent plutôt sur la présence de mots que sur l'absence de mots. Justement, la présence de mots est probablement plus représentative de la catégorie du texte que l'absence de mots.

Précisément, soit un document d et une classe y, chaque document d constitué de termes x_i $(i = 1, \ldots, n)$ est représenté par un vecteur de poids :

$$W_d = (w_y(x_1, d), w_y(x_2, d), \ldots, w_y(x_n, d)) \tag{3.1}$$

La similarité cosinusoïdale entre deux instances d_i et d_j est définie comme suit :

$$COS_y(d_i, d_j) = \frac{W_{d_i} W_{d_j}}{||W_{d_i}||||W_{d_j}||}$$

où $||W||$ représente la norme du vecteur W.

Notons $D_y(d_i, k)$ les k plus proches voisins d_i au sens de la similarité COS_y. Le score d'appartenance d'un document d_i à la catégorie y est définie comme suit :

$$score(d_i, y) = \sum_{d_j \in D_y(d_i,k)} COS_y(d_i, d_j) \times \Phi(d_j, y)$$

où $\Phi(d_j, y)$ désigne la fonction qui vaut 1 si le document d_j appartient à la catégorie y, 0 sinon.

Comme il a déjà été mentionné, le classifieur kNN n'implique pas de phase d'apprentissage en tant que telle. Cela nous amène à discuter d'une classe d'apprentisseurs qu'on peut qualifier de paresseux (*lazy*). La seule opération préalable est le stockage des exemples d'apprentissage. L'apprentissage est repoussé au moment où une nouvelle instance à classer arrive. Par le fait même, la plus grosse part de l'effort requis en termes de temps de calcul est fourni au moment même de la classification.

Une des caractéristiques de l'apprentissage à base d'instances est qu'il n'y a pas de construction d'une description explicite de la fonction à apprendre (dans notre cas, l'appartenance à une classe). L'avantage est qu'on n'estime pas qu'une seule fois la fonction pour tout l'espace, mais on l'estime plutôt localement et différemment pour chaque nouvelle instance. L'inconvénient est que bien que le coût d'apprentissage du classifieur soit faible, car il ne fait que mémoriser les exemples d'apprentissage, le coût de classification de nouvelles instances peut être élevé, puisque c'est à ce moment que tout le calcul se fait. Cependant, une bonne indexation des exemples aide beaucoup à pallier ce problème.

3.4 Plus proche centroïde et Rocchio

Les méthodes de Rocchio [95, 47] et du plus proche centroïde [40] sont parmi les plus anciens algorithmes de classification de textes destinés à l'amélioration des systèmes de recherche documentaires. Dans ces deux méthodes chaque classe est représentée par un vecteur prototype, qui peut être par exemple la moyenne des vecteurs représentant chaque document appartenant à une

classe (le centroïde moyen). L'avantage des classifieurs à base de centroïdes est la simplicité, l'interprétabilité et la rapidité. Pour un expert, le vecteur prototype est plus compréhensible qu'un réseau de neurones par exemple. L'apprentissage de ce type de classifieur est de plus très rapide et est souvent précédé par une sélection et une réduction de termes.

Étant donné un ensemble de documents appartenant à la même classe et leurs représentations vectorielles, le vecteur centroïde est traditionnellement défini comme le vecteur obtenu en faisant la moyenne ou en ajoutant les poids des différents termes présents dans les documents [40]. Ces poids sont généralement non supervisés.

La classification consiste alors à associer à chaque document la classe (ou les M classes) correspondant(s) au(x) vecteur(s) prototype(s) centroïde(s) la (les) plus proche(s) au sens de la similarité cosinusoïdale.

Précisément, soit un document d et une classe y, chaque document d constitué de termes x_i $(i = 1, \ldots, n)$ est représenté par un vecteur de poids :

$$W_d = (w_y(x_1, d), w_y(x_2, d), \ldots, w_y(x_n, d)) \tag{3.2}$$

Le vecteur centroïde moyen (Arithmetical average centroid) est défini par :

$$W_y^{\mathrm{AAC}} = \frac{\sum_{d \in y} W_d}{f(*y)}$$

Le poids de chaque terme dans le vecteur centroïde moyen est donc :

$$w_y^{\mathrm{AAC}}(x) = \frac{\sum_{d \in y} w_y(x, d)}{f(*y)}$$

Le poids de chaque terme dans le vecteur centroïde somme (Sum centroid) est défini comme :

$$w_y^{\mathrm{SC}}(x) = \sum_{d \in y} w_y(x, d)$$

On parle de classifieur Rocchio lorsque chaque vecteur prototype d'une classe est définie comme la différence pondérée entre la moyenne des vecteurs des documents appartenant à la classe (documents positifs) et la moyenne des vecteurs des documents négatifs.

Le poids de chaque terme dans le vecteur Rocchio (Rocchio centroid) est défini comme :

$$w_y^{\mathrm{RC}}(x) = \alpha \sum_{d \in y} w_y(x, d) - \beta \sum_{d \notin y} w_y(x, d)$$

où α et β sont des constantes.

Récemment, de nouvelles méthodes basées sur les centroïdes ont été proposées, elles différent par les métriques utilisés dans :

— la pondération des termes des documents à classer ;
— la pondération des termes dans les vecteurs prototypes des centroïdes.

Debole et Sebastiani [15] ont utilisé la méthode Rocchio avec des pondération de termes de documents supervisées (Pearson's χ^2, Information gain, Gain ratio). Ren et al. [94] calculent des centroïdes en se basant sur les métriques de pondération de termes de document qu'ils ont proposé (ICF and $\text{ICS}_\delta\text{F}$).

Guan et al. [33] ont proposé une fonction centroïde CFC (class feature centroid) de pondération des termes qui n'est pas réductible à la moyenne ou l'ajout de poids de termes de documents. Le poids de chaque terme dans le vecteur CFC est défini comme : $w_y^{\text{CFC}}(x) = b^{\frac{f(xy)}{f(*y)}} \log(\frac{M}{f_c(x)})$.

Ce poids dépend d'une constante b (ils ont considérés $b = e = 2,71828\ldots$ et $b = e - 1$) et de fréquences qui ont été définies dans le chapitre 1 :

— le nombre $f(xy)$ de documents appartenant à la classe y et contenant le terme x,
— le nombre $f(*y)$ de documents appartenant à la classe y,
— le nombre $f_c(x)$ de classes contenant un document contenant le terme x,
— le nombre de classes M.

Dans [33] les vecteurs représentant les documents sont calculés en utilisant une pondération non supervisée des termes (TFIDF).

Nguyen et al. [81] ont proposé un schéma de pondération de centroïdes qui généralise les centroïdes CFC en utilisant les mesures de divergences Kullback-Leibler (KL) et Jensen-Shannon (JS) pour la pondération des termes des documents et des vecteurs prototypes de centroïdes.

Bouillot et al. [8] proposent des variantes de la métrique CFC en étudiant l'influence du nombre de classes, de documents et de termes dans la classification des textes courts.

3.5 Les machines à vecteurs de support (SVM)

Les machines à vecteurs de support, appelées aussi séparateurs à vaste marge, sont des techniques d'apprentissage supervisé basées sur la théorie de l'apprentissage statistique ou automatique. Les SVM sont apparus en 1995 suite aux travaux de Vapnik [14]. SVM traite initialement d'un problème de classification binaire.

Le but de SVM est de déterminer si un élément appartient à une classe ou pas. On dispose d'un ensemble de données et on cherche à séparer ces données en deux groupes. L'algorithme SVM permet de trouver un hyperplan séparateur entre ces deux groupes. Pour optimiser la séparation SVM cherche l'hyperplan pour lequel la distance entre la frontière des deux groupes et les

points les plus proches est maximale, c'est le principe de maximisation de la marge.

Dans certains cas les données sont non linéairement séparables. Il n'existe donc pas un hyperplan séparateur. Deux options sont possibles pour palier à ce problème :

— Définir une constante qui permet de tolérer une marge d'erreur (marge souple). On peut distinguer une constante pour les classes positive C_p et une constante pour les classes négatives C_n. Les deux constantes sont en relation par un coefficient j. Soit $C_p = j \times C_n$.

— Chercher une autre séparation non linéaire : pour faciliter les calculs de cette frontière non linéaire on utilise une fonction noyau qui permet de passer dans un autre espace vectoriel de plus grande dimension où une frontière linéaire existe. Parmi les fonctions noyaux on peut citer les noyaux polynomiaux, gaussiens etc.

SVM présente une solution au problème de classification binaire. Pour le problème à plusieurs classes on utilise les approches classiques permettant de passer d'un classifieur binaire à classifier multi-classes, deux approches sont couramment utilisées :

— Un–versus-tout : dans ce cas on construit autant de classifieurs que de classes, pour une instance donnée, parmi tous les classifieurs qui ont donné un résultat positif, on considère celui qui a retourné la plus grande marge.

— Un-versus-un : si n est le nombre de classes, alors on construit tous les classifieurs possibles en prenant les classes deux à deux, donc $n(n–1)/2$, pour une instance donnée, elle appartiendra à la classe citée le plus souvent après application de tous ces algorithmes.

3.6 Arbres de décision

Un arbre de décision est la traduction de divisions successives de l'espace des instances en régions distinctes chacune dominée par une classe. Pour diviser l'espace des instances on utilise à chaque fois le descripteur qui discrimine le mieux les instances de la région considérée. En répétant plusieurs fois cette opération on obtient un arbre. On classe une nouvelle instance en parcourant l'arbre à partir de la racine jusqu'à la feuille représentant la région dans laquelle on le classe. Dans ce parcours à chaque nœud de l'arbre on teste le critère utilisé pour la division de la région (le descripteur choisi et le seuil utilisé) lors de la construction de l'arbre. La procédure de classification obtenue a une traduction immédiate en terme de règles de décision. Les systèmes de règles obtenus sont particuliers car l'ordre dans lequel on examine les

attributs est fixé et les règles de décision sont mutuellement exclusives.

Dans toutes les méthodes il faut :

— décider si un nœud est terminal, c'est-à-dire décider si un nœud doit être étiqueté comme une feuille représentant une région qu'on ne divise plus. En considérant les critères suivants par exemple : toutes les instances sont dans la même classe, il y a moins d'un certain nombre d'erreurs (instances appartenant à une classe qui n'est pas la classe majoritaire dans la région) etc.

— diviser à chaque nœud les instances de la région selon un critère probabiliste, statistique ou issue de la théorie de l'information. Ce critère permet de quantifier la capacité des descripteurs à discriminer les instances pour choisir le meilleur. Dans les arbres aléatoires on considère uniquement un sous-ensemble de descripteurs choisis aléatoirement.

— affecter une classe à une feuille (région). On attribue la classe majoritaire dans la région. La fréquence des instances appartenant à la classe majoritaire dans la région permet d'estimer une probabilité représentant le degré d'appartenance à la classe.

Les méthodes vont différer par les choix effectués pour les critères de terminaison, division et affectation d'une classe.

Le schéma général des algorithmes de construction d'un arbre de décision est le suivant :

Algorithm 1 Algorithme d'apprentissage générique

Require: Langage de description ;
Require: Échantillon S

```
repeat
    Décider si le nœud courant est terminal
    if le nœud est terminal then
        Affecter une classe
    else
        Sélectionner un test et créer le sous-arbre
    end if
    Passer au nœud suivant non exploré s'il en existe
until obtenir un arbre de décision
```

end

3.7 Ensemble d'apprentissage

L'objectif des méthodes d'ensemble est de combiner les prédictions de plusieurs estimateurs de base construits avec un algorithme d'apprentissage

donné en vue d'améliorer la généralisabilité/robustesse sur un seul estimateur.

Deux familles de méthodes d'ensemble sont généralement distinguées :

— les méthodes basées sur le consensus (moyen, majoritaire etc.). Le principe est de construire plusieurs estimateurs indépendamment, puis de faire un consensus sur leurs prédictions. L'estimateur combiné est généralement meilleur que tous les estimateurs de base, car sa variance est réduite. Exemples : bagging, forêts d'arbres aléatoires etc.

— les méthodes basées sur le boosting. Les estimateurs de base sont construits de façon séquentielle et on essaie de réduire le biais de l'estimateur combiné. Le but est de combiner plusieurs modèles faibles pour produire un ensemble puissant. Exemples : AdaBoost, Gradient boosting etc.

Les apprenants de bases sont généralement produits à partir des données d'apprentissage par un algorithme d'apprentissage de base qui peut être l'arbre de décision, réseaux de neurones ou d'autres types d'algorithmes d'apprentissage. La plupart des méthodes d'ensemble utilisent une seule méthode d'apprentissage de base pour produire des apprenants de bases homogènes, mais il y a aussi des méthodes qui utilisent de multiples algorithmes d'apprentissage pour produire des apprenants hétérogènes. Dans ce dernier cas, il n'y a pas d'algorithme d'apprentissage de base unique et donc, on préfère appeler les apprenants : apprenants individuels ou apprenants composants plutôt que des apprenants de bases.

Ce domaine a commencé avec les travaux de Hansen et Salamon [41] à la fin des années 1980, dans lesquels ils ont constaté que les prédictions faites par la combinaison d'un ensemble de classifieurs sont souvent plus précises que les prédictions faites par le meilleur classifieur unique. Schapire [99] a montré que les apprenants faibles peuvent être boostés.

3.7.1 Construction d'un ensemble d'apprentissage

En règle générale, un ensemble est construit en deux étapes.

— Dans un premier temps, un certain nombre d'apprenants de base sont produits en parallèle (indépendamment) ou en séquentiel (un apprenant de base a une influence sur les apprenants ultérieurs).

— Ensuite, les apprenants de bases sont combinés pour être utilisés.

Pour obtenir un bon ensemble, les apprenants de bases doivent être aussi précis que possible et les plus divers possibles [63]. Il y a plusieurs manières d'estimer la précision des apprenants (par validation croisée etc.). La diversité des apprenants de bases peut se faire en utilisant diverses techniques telles que le sous-échantillonnage, l'exemple, la manipulation les descripteurs,

la manipulation des sorties, l'injection aléatoire dans des algorithmes d'apprentissage, ou la combinaison simultanée de ces techniques. Il existe de nombreuses méthodes d'ensemble efficaces. Dans ce qui suit nous allons présenter brièvement trois méthodes représentatives, le boosting [99, 29], le bagging [9] et le stacking [126]. Pour la simplicité du propos, nous considérons uniquement le cas de la classification binaire.

Dans ce qui suit X et Y désignent respectivement l'espace des instances et l'ensemble des classes en supposant $Y = \{-1, +1\}$.

$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$ désigne l'ensemble des données d'apprentissage, où $x_i \in X$ et $y_i \in Y (i = 1, \ldots, m)$.

3.7.2 Boosting

Le boosting est une famille d'algorithmes car il en existe de nombreuses variantes. Nous considérons ici AdaBoost [29] l'algorithme le plus connu. Premièrement, il attribut des poids égaux à tous les exemples d'apprentissage. On dénote par D_t la distribution des poids à la t-ème itération d'apprentissage. À partir l'ensemble de données d'apprentissage et de D_t l'algorithme génère un apprenant de base $h_t : h_t : X \to Y$ en faisant appel à l'algorithme d'apprentissage de base. Ensuite, il utilise les exemples d'apprentissage pour tester h_t et les poids des exemples mal classés seront augmentés. Ainsi, une mise à jour de la distribution de poids D_{t+1} est obtenue. À partir de l'ensemble de données d'apprentissage et de D_{t+1} AdaBoost génère un autre apprenant de base en appelant à nouveau l'algorithme d'apprentissage de base. Un tel processus est répété T fois (itérations), et l'apprenant final est choisi par un vote à la majorité pondérée des apprenants de bases de T, où les poids des apprenants sont déterminés au cours de l'apprentissage. Dans la pratique, l'algorithme d'apprentissage de base peut être un algorithme qui utilise des exemples d'apprentissages pondérés directement ; autrement dit, les poids peuvent être exploités par un échantillonnage des exemples d'apprentissages en fonction de la répartition du poids D_t. Le pseudo-code de AdaBoost est présenté dans ce qui suit :

Algorithm 2 L'algorithme AdaBoost

Require: Ensemble de données $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$;
Require: Algorithme d'apprentissage L ;
Require: Nombre d'itérations d'apprentissage T ;
Ensure: $H(x) = (sign(f(x)) = sign \sum_{t=1}^{T} \alpha_t h_t(x)$
1: $D_1(i) = 1/m$; % initialiser le poids d'apprentissage
2: **for** $t = 1, \ldots, T$ **do**

3: $h_t = L(D, D_t)$;% Entraine un apprenant h_t de D en utilisant la distribution D_t
4: $\epsilon_t = \Pr_{i\sim D_i}[h_t(x_i) \neq y_i]$; % Mesure l'erreur de h_t
5: $\alpha_t = 1/2 \ln \frac{1-\epsilon_t}{\epsilon_t}$; % Détermine le poids de h_t
6: $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \exp(-\alpha_t)$ si $h_t(x_i) = y_i$;
7: $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \exp(\alpha_t)$ si $h_t(x_i) \neq y_i$;
8: $= D_t(i)\exp(-\alpha_t y_i h_t(x_i))Z_t$;
% MAJ de la distribution, où Z_t est le facteur de normalisation qui permet à D_{t+1} d'être une distribution
9: **end for**

end

3.7.3 Bagging

L'algorihme de bagging [9] entraîne un certain nombre d'apprenants de bases chacun à partir d'un échantillon de bootstrap différent en appelant un algorithme d'apprentissage de base. Un échantillon de bootstrap est obtenu par sous-échantillonnage de l'ensemble de remplacement, la taille d'un échantillon est la même que celle de l'ensemble données d'apprentissage. Ainsi, pour un échantillon bootstrap, quelques exemples d'apprentissage peuvent apparaître, mais d'autres ne le peuvent pas. la probabilité qu'un exemple apparaisse au moins une fois est d'environ 0,632. Après avoir obtenu les apprenants de bases, l'algorithme de bagging les combinent par la majorité votée et la classe la plus votée est prédite. Le pseudo-code de l'algorihme de bagging est donné dans la figure 3. Une variante du bagging, les forêts aléatoires [10], a été jugée comme l'une des méthodes les plus puissantes parmi les méthodes d'ensemble. L'algorithme est présenté dans ce qui suit :

Algorithm 3 The Bagging algorithm
Require: Ensemble de données $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$;
Require: Algorithme d'apprentissage L ;
Require: Nombre de itérations d'apprentissage T.
Ensure: $H(x) = \text{argmax}_{y\in Y} \sum_{t=1}^{T} l(y = h_t(x))$
1: **for** $t = 1, \ldots, T$ **do**
2: $D_t = \text{Bootstrap}(D)$; % Générer un échantillon Bootstrap à partir de D
3: $h_t = L(D_t)$; % Entrainer un apprenant de base h_t de l'échantillon Bootstrap
4: **end for**

end

3.7.4 Stacking

Dans une mise en œuvre typique de l'algorithme de stacking [126], un certain nombre de premier niveau des apprenants individuels sont générés à partir des données d'apprentissage en utilisant différents algorithmes d'apprentissage. Ces apprenants sont ensuite combinés par un apprenant de second niveau qui est appelé méta-apprenant. Le pseudo-code de l'algorithme de stacking est présenté dans ce qui suit :

Algorithm 4 The Stacking algorithm

Require: Ensemble de données $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$;
Require: Algorithmes d'apprentissages de premier niveau $L_1, \ldots, L_T$;
Require: Algorithmes d'apprentissages de premier niveau L.
Ensure: $H(x) = h'(h_1(x), \ldots, h_T(x))$

1: **for** $t = 1, \ldots, T$ **do**
2: $h_t = L_t(D)$ % Entrainer un apprenant de premier niveau h_t en appliquant un algorithme d'apprentissages de premier niveau L_t à l'ensemble de données original D
3: **end for**
4: $D' = \Phi$; % Générer un nouvel ensemble de données
5: **for** $t = 1, \ldots, T$ **do**
6: **for** $t = 1, \ldots, T$ **do**
7: $z_{it} = h_t(x_i)$ % Utiliser h_t pour classer l'ensemble d'exemples x_i
8: **end for**
9: $D' = D' \cup ((z_{i1}, z_{i2}, \ldots, z_{iT}), y_i)$
10: **end for**
11: $h' = L(D')$. % Entrainer un apprenant de premier niveau h_t en appliquant un algorithme d'apprentissages de premier niveau L_t au nouvel ensemble de données D'

end

D'une manière générale, il n'y a pas de méthode d'ensemble qui surpasse les autres méthodes d'ensemble. Des études empiriques sur les méthodes d'ensemble existent dans la littérature telles que [6, 105, 84]. Auparavant, on pensait qu'utiliser plus d'apprenants de bases donnerait de meilleures performances, mais Zhou et al. [135] ont donné le théorème "many could be better than all" qui montre que ce ne soit pas le cas. Ces ensembles sont appelés ensembles sélectifs. Il est à noter, qu'en plus de la classification et de la régression, les méthodes d'ensemble ont également été conçue pour le clustering [103] et d'autres types de tâches d'apprentissage.

3.8 Conclusion

Ce chapitre présente quelques algorithmes d'apprentissages, particulièrement ceux utilisés dans cette thèse pour le problème de la classification de textes qui sera exposé dans le chapitre suivant.

Chapitre 4

Classification de textes

Sommaire

4.1 Introduction

Avec le développement d'Internet nous sommes en présence d'une quantité énorme d'information électronique. Le coût de plus en plus faible du stockage fait en sorte que cette quantité augmente continuellement. Parallèlement à cela, les sources de données ne cessent de se diversifier, aujourd'hui pour accéder à une information on peut avoir recours à des journaux électroniques, à des sites spécialisés, à des réseaux sociaux, etc. Pour pouvoir gérer cette quantité grandissante de données et en tirer le plus d'information possible il devient nécessaire de l'organiser et de la catégoriser.

La classification de texte est l'étiquetage automatique des textes en langage naturel avec des classes (catégories) thématiques prédéfinies. Ces deux dernières décennies, un grand nombre de techniques d'apprentissage ont été proposées pour classer automatiquement et organiser les documents textuels [100, 1]. Ces études ont été motivées par la croissance exponentielle de textes disponibles en ligne. Les domaines d'applications sont nombreux, la classification des articles de presse, des pages Web et des publications scientifiques avec vocabulaire contrôlé, l'analyse des sentiments et de l'extraction d'opinion auprès des réseaux sociaux, le filtrage de spam, la classification des protéines en biologie.

4.2 Problèmes et applications

Dans les systèmes de classification de textes, les documents sont prétraités afin de les convertir en données à un algorithme d'apprentissage. Traditionnellement, chaque document textuel est converti en un vecteur où chaque dimension correspond à un terme qui est représenté par la valeur de son poids. Cette valeur sera utilisée dans le processus d'apprentissage. Comme le poids reflète l'importance du terme dans le document, un choix approprié de la métrique utilisée pour la pondération des termes est cruciale pour une classification correcte.

Les métriques de pondération de termes non supervisées traditionnelles comme la métrique populaire TFIDF ne dépendent que de la fréquence des termes dans le document et (l'inverse) du nombre de documents d'apprentissage contenant ce terme. Des métriques de classification de textes supervisées ont été développées afin de prendre en compte les classes des documents du corpus [15, 16, 18, 26, 64, 69, 5, 74, 3, 81, 94, 17].

L'idée est de construire une métrique qui discrimine les termes selon la classe pour laquelle nous testons l'appartenance du document. Une telle mesure devrait être informative sur la relation d'un terme du document t à une

classe c, et discriminative pour séparer les documents positifs des documents négatifs de la classe c.

Alors que les poids des termes non supervisés ne dépendent que de la fréquence des termes dans le document d'apprentissage et du nombre des documents du corpus contenant ce terme, les poids supervisés d'un terme sont calculés pour chaque classe et utilisent le nombre de documents appartenant à la classe contenant ce terme. Intuitivement, les poids supervisés des termes mesurent le degré de corrélation entre le terme présent dans un document et l'appartenance de ce document à la classe.

4.3 Principe des méthodes de classification de textes

Les méthodes de classification de textes réduisent au préalable chaque document textuel en un vecteur dont chaque entrée représente le poids d'un terme. La classification est mise en œuvre par un algorithme d'apprentissage qui classe chaque document comme appartenant ou non à une catégorie (classe) prédéfinie. Le poids de chaque terme représente un descripteur du document pour la méthode d'apprentissage.

4.3.1 Représentation vectorielle

Quelle que soit la méthode de classification retenue, la première opération consiste à représenter les documents de façon à ce qu'ils puissent être traités automatiquement par les classifieurs. La plupart des approches utilisent pour cela la représentation vectorielle des documents La représentation vectorielle ou modèle vectoriel (VSM pour Vector Space Model) [97] a été initialement développée pour le système SMART [32]. Le principe consiste à représenter chaque document de la collection comme un point de l'espace, i.e. un vecteur de coordonnées dans l'espace vectoriel. Les coordonnées correspondent en fait aux descripteurs composant le document.

Actuellement cette représentation est utilisée par la plupart des moteurs de recherche, les systèmes de recommandation et de filtrage collaboratif [66] ou encore les algorithmes mesurant les relations sémantiques [115, 90]. Cette représentation permet de construire des modèles simples et robustes de classification de documents. Elle permet aussi de mettre en évidence les descripteurs les plus intéressants pour chaque classe rendant ces modèles compréhensibles lors de la phase d'apprentissage. Un corpus composé de m classes et n descripteurs peut être représenté par une matrice $X_{m \times n}$ avec m lignes et n colonnes où chaque dimension du vecteur w_{ij} est le poids du j-ème

descripteur de la i-ème classe. De nombreuses méthodes ont été proposées dans la littérature pour définir le poids w_{ij} d'un descripteur.

4.3.2 Métriques de pondération

Les nombreuses méthodes proposées dans la littérature peuvent généralement être divisées en deux groupes :

— Les méthodes de pondérations supervisées dans lesquelles on va utiliser les informations relatives à l'appartenance de la classe.
— Les méthodes de pondérations non supervisées qui ne tiennent pas compte de cette information.

L'idée de la pondération est de quantifier le poids d'un descripteur en fonction de son importance afin de le différencier des autres.

4.3.3 Classification

La classification des textes utilise un algorithme d'apprentissage qui opère sur les représentations vectorielles des documents. Nous avons abordé les algorithmes d'apprentissage dans le chapitre précédent et consacrons ce chapitre aux méthodes de pondération spécifiques à la classification de textes.

4.4 Définitions et notations

Nous considérons un corpus D de N documents et d un document particulier de D. Soit x qui désigne un descripteur nominal de d représentant soit :

— t un terme qui apparait dans d ;
— (t, n) un terme qui apparait au moins n fois dans d ;
— ou (t, p) un terme dont la première position est inférieure ou égale à p dans le document d.

Chaque document peut appartenir à une ou plusieurs catégories (ou classes) étiquettes $c_1, c_2, \ldots, c_M$.

Nous désignons par y une classe c_i en particulier.

On note $\bar{x}$ le fait que le descripteur x n'est pas présent dans d et par $\bar{y}$ le fait que d ne fait pas partie de la classe y.

Le nombre de documents contenant le descripteur x et appartenant à la classe y est noté par $f(xy)$ et représente la fréquence des documents.

En général, $f(uv)$ désigne le nombre de documents contenant u et appartenant à v, u étant x, $\bar{x}$ ou $*$ (les documents contenant un terme) et v étant y, $\bar{y}$ ou $*$ (documents appartenant à une classe).

	y	$\bar{y}$	$*$
x	$f(xy) = f_{11} = a$	$f(x\bar{y}) = f_{12} = b$	$f(x*) = f_1$
$\bar{x}$	$f(\bar{x}y) = f_{21} = c$	$f(\bar{x}\bar{y}) = f_{22} = d$	$f(\bar{x}*)$
$*$	$f(*y) = f_2$	$f(*\bar{y})$	$f(**) = N$

TABLE 4.1 – Tableau de contingence du descripteur nominal x et la classe y

	y	$\bar{y}$	$*$
x	$\hat{f}(xy) = \frac{f(x*)f(*y)}{N}$	$\hat{f}(x\bar{y}) = \frac{f(x*)(N-f(*y))}{N}$	$\hat{f}(x*)$
$\bar{x}$	$\hat{f}(\bar{x}y) = \frac{(N-f(x*))f(*y)}{N}$	$\hat{f}(\bar{x}\bar{y}) = \frac{(N-f(x*))(N-f(*y))}{N}$	$\hat{f}(\bar{x}*)$
$*$	$\hat{f}(*y)$	$\hat{f}(*\bar{y})$	N

TABLE 4.2 – Tableau des fréquences attendues du descripteur nominal x et de la classe y

Ces fréquences sont représentées dans le tableau de contingence 4.1 dans lequel le nombre de documents est désigné par N, $f(xy)$ par a et f_{11}, $f(x, y)$ par b et f_{12}, et ainsi de suite.

Beaucoup de paramètres sont basés sur l'estimation de la la probabilité notée $p(uv)$ qu'un document contenant u appartienne à la classe v, u étant x,$\bar{x}$ ou $*$ et v étant y, $\bar{y}$ ou $*$.

Sous l'hypothèse de vraisemblance maximale cette probabilité est estimée par

$$p(uv) = \frac{f(uv)}{N}$$

Certaines métriques sont basées sur la différence entre les fréquences observées et les fréquences attendues. Les fréquences de contingences attendues sous l'hypothèse d'indépendance nulle H_0 sont données dans le 4.2. Peu de mesures utilisent le nombre de classes contenant un document qui contient un élément x. Cette quantité est désignée par $f_c(x)$ et correspond à :

$$f_c(x) = |\{y|f(xy) > 0\}|$$

Attribuer un poids à un descripteur x qui est associé à un terme dans un document étiqueté par y dépend de la corrélation entre x et y dans le corpus d'apprentissage. Cette corrélation peut être estimée par différents paramètres, toutes les métriques utilisées dans ce document ne dépendent que de quatre valeurs :

— N : le nombre de documents de l'ensemble d'apprentissage ;
— $f(xy)$: la fréquence jointe ;
— $f(x*)$ et $f(*y)$: les fréquences marginales ;

— $f_c(x)$: le nombre de classes contenant un document qui contient le descripteur x ;
— M : le nombre de classes.

4.5 Les méthodes de pondérations non supervisées

La pondération du descripteur est une étape importante pour améliorer l'efficacité des classifieurs. L'idée de la pondération est de quantifier le poids d'un descripteur en fonction de son importance afin de le différencier des autres. Intuitivement, il est assez simple d'imaginer que si un même descripteur apparait dans une classe A mais pas dans une classe B, il ne peut avoir le même poids pour c_i et c_j dans la représentation vectorielle. Avec une pondération booléenne, le poids d'un descripteur vaut 1 s'il apparaît, 0 sinon. De même si un même descripteur apparaissait dans deux classes, cela ne veut pas signifier qu'il ait une importance similaire. Si on utilise la fréquence du descripteur, le poids d'un descripteur vaudra le nombre d'occurrences du descripteur dans la classe. Définir le poids des descripteurs implique deux phases :

— savoir quel descripteur est plus représentatif qu'un autre ;
— mettre en avant le poids des descripteurs les plus discriminants et limiter le poids des descripteurs les moins importants pour la classification.

Les premières métriques de pondérations de termes pour la classification de textes étaient non supervisées et généralement empruntées au domaine de la recherche d'informations (IR). La métriques plus simple de IR est la représentation binaire BIN qui attribue un poids de 1 si le terme apparaît dans la document et 0 sinon. On peut affecter au terme un poids TF qui dépend de sa fréquence dans le document. Différentes variantes de fréquence du terme ont été présentées, par exemple, la fréquence de terme brut RTF ou son logarithme (1 + TF).

TFIDF est la métrique de pondération la plus couramment utilisée dans la classification de textes. TFIDF est le produit de TF et IDF, la fréquence de documents inverse, qui favorise les termes rares par rapport aux termes fréquents dans le corpus en prenant en compte la longueur des documents dans le calcul de discriminance. Cependant, il existe quelques inconvénients à utiliser des métriques de pondérations non supervisées, en effet, l'information liée à la classe est omise.

BIN, TF, IDF, TFIDF et leurs variantes font donc partie des méthodes

de pondérations non supervisées. Les approches de pondérations supervisées utilisent une information complémentaire liée à la classe, il serait logique qu'elles soient plus précises et donc plus pertinentes.

4.6 Les méthodes de pondérations supervisées

Des études antérieures ont proposé différentes métriques de pondérations supervisées où le facteur fréquence des documents IDF dans TFIDF est remplacé par un facteur qui utilise l'information préalable sur le nombre de documents appartenant à chaque classe. Plusieurs métriques couramment utilisées ont été testées dans la littérature [15, 16, 18, 5], par exemple :

— chi-deux (χ^2)

$$\sum_{i,j} \frac{(f_{ij} - \hat{f_{ij}})}{\hat{f_{ij}}}$$

— informations gain (IG)

$$\sum_{u \in \{x,\bar{x}\}} \sum_{v \in \{y,\bar{y}\}} p(uv) \log \frac{p(uv)}{p(u*)p(*v)}$$

— gain ratio (GR)

$$\frac{\sum_{u \in \{x,\bar{x}\}} \sum_{v \in \{y,\bar{y}\}} p(uv) \log \frac{p(uv)}{p(u*)p(*v)}}{\sum_{v \in \{y,\bar{y}\}} p(v) \log p(v)}$$

— odds ratio (OR)

$$\frac{ad}{bc}$$

Les premières études obtiennent une amélioration avec les métriques de pondérations des termes TF.χ^2, TF.IG, TF.GR et TF.OR en utilisant les SVM.

Une classification de textes avec les SVM, plus précise, a été obtenue en utilisant la métrique séparation Bi-Normal (BNS) [25] pour la pondération de termes supervisée [26]. Dans son étude, Forman a testé deux variantes (en considérant la fréquence du terme ou non)

$$\text{BNS} = |F^{-1}(p(x|y)) - F^{-1}(p(x|\bar{y}))|$$

avec F^{-1} fonction de répartition normale de distribution. Il a remarqué qu'un "meilleur F-score a été obtenu en utilisant les descripteurs binaires avec la BNS " (BIN.BNS) mais le "rappel était un peu meilleur avec des descripteurs

TF.BNS". Cette observation montre que la fusion TF avec un facteur tel que BNS est problématique, c'est-à-dire, de ne pas utiliser TF a un meilleur rendement F_1-Score mais diminue la fraction des classes pertinentes pour un document prédit.

Plus récemment, d'autres métriques spécifiques ont été proposées pour le problème de pondérations de termes supervisées. Liu et al. [69] utilisent une pondération de termes basée sur les probabilités (PB) pour aborder le problème de la répartition déséquilibrée des documents entre les classes.

$$\mathrm{PB} = \log(1 + \frac{a^2}{bc})$$

Lan et al. [64] utilisent une métrique de pondération de termes TF.RF basée sur la métrique Fréquence de pertinence (relevant frequency RF).

$$\mathrm{RF} = \log_2(2 + \frac{a}{\max(b, 1)})$$

La relation et les différences entre ces métriques de pondération de termes sont étudiées dans [2].

Martineau et Finin [74] proposent une métrique TF.δIDF. où IDF est remplacé par l'inverse de la différence de fréquence de documents par classe (δIDF).

$$\delta\mathrm{IDF} = \log(\frac{p(x|y)}{p(x|\bar{y})})$$

Altinçay and Erenel [3] combinent la métrique RF avec la métrique information mutuelle MI et la différence des probabilités des occurrences de termes dans l'ensemble des documents appartenant à la classe et dans son ensemble complémentaire.

$$\mathrm{RF}_{OR} = \log_2(2 + \frac{a}{\max(b, 1)})(1 + p(x|y) - p(x|\bar{y}))$$

$$\mathrm{RF}_{\chi^2} = \log_2(2 + \frac{a}{\max(b, 1)})|p(x|\bar{y}) - p(x|y)|$$

Nguyen et al. [81] proposent un schéma de pondération basé sur les métriques Kullback- Leibler (KL) et la divergence Jensen-Shannon (JS) avec les classifieurs centroides.

Ren et al. [94] essaient deux métriques TF.IDF.ICF et TF.IDF.ICS$_\delta$F qui incorporent à TFIDF, la fréquence de classe inverse (ICF) et la fréquence inverse de la densité de l'espace de la classe (ICS $_\delta$F).

$$\mathrm{IDF.ICF} = \log(1 + \frac{N}{f(x*)}) \log(1 + \frac{M}{f_c(x)})$$

$$\text{IDF.ICS}_\delta F = \log(1 + \frac{N}{f(x*)}) \log(1 + \frac{M}{\sum_{c_i \in C} p(x|c_i)})$$

Deng et al. [17] et Fattah [23] adaptent et comparent les différentes métriques de pondération pour la classification de textes dans l'analyse des sentiments.

Cette application est également considérée dans les deux articles pré-cités [74] et [81].

Badawi et al. [4] proposent un framework basé sur l'utilisation des statistiques de co-occurrences de paires de termes pour la sélection et la pondération de termes dans la classification binaire de textes.

Scalante et al. [22] utilisent la programmation génétique pour la pondération de termes. Ko [60] utilise un système de pondération basé sur le ratio de pertinence du terme (Term Relevance Ratio TRR).

La table 4.3 résume les métriques qui ont déjà été utilisées pour le problème de la pondération des termes dans la littérature [16, 18, 26, 69, 64, 74, 3, 94].

Metrique	Formule
IDF	$\log(\frac{N}{f(x*)})$
Pearson's χ^2 test [16]	$\sum_{i,j} \frac{(f_{ij} - \hat{f_{ij}})}{\hat{f_{ij}}}$
Information gain [16]	$\sum_{u \in \{x,\bar{x}\}} \sum_{v \in \{y,\bar{y}\}} p(uv) \log \frac{p(uv)}{p(u*)p(*v)}$
Gain ratio [16]	$\frac{\sum_{u \in \{x,\bar{x}\}} \sum_{v \in \{y,\bar{y}\}} p(uv) \log \frac{p(uv)}{p(u*)p(*v)}}{\sum_{v \in \{y,\bar{y}\}} p(v) \log p(v)}$
Odds ratio [18]	$\frac{ad}{bc}$
Log odds ratio [26]	$\log \frac{ad}{bc}$
Forman log odds ratio [26]	$\lvert \log \frac{ad}{bc} \rvert$
Bi-normal separation (BNS) [26]	$\lvert F^{-1}(p(x\|y)) - F^{-1}(p(x\|\bar{y})) \rvert$ [1]
Probability based term weight [69]	$\log(1 + \frac{a^2}{bc})$
Pointwise mutual information [69]	$\log \frac{p(xy)}{p(x*)p(*y)}$
Relevance frequency [64]	$\log_2(2 + \frac{a}{\max(b,1)})$
Relevance frequency$_{OR}$ [3]	$\log_2(2 + \frac{a}{\max(b,1)})(1 + p(x\|y) - p(x\|\bar{y}))$
Relevance frequency$_{\chi^2}$ [3]	$\log_2(2 + \frac{a}{\max(b,1)}) \lvert p(x\|\bar{y}) - p(x\|y) \rvert$
δIDF [74]	$\log(\frac{p(x\|y)}{p(x\|\bar{y})})$
IDF.ICF [94]	$\log(1 + \frac{N}{f(x*)}) \log(1 + \frac{M}{f_c(x)})$
IDF.ICS$_\delta$F [94]	$\log(1 + \frac{N}{f(x*)}) \log(1 + \frac{M}{\sum_{c_i \in C} p(x\|c_i)})$

1. F^{-1} est l'inverse de la fonction de répartition de la loi normale.

TABLE 4.3: Metriques utilisées pour la pondération supervisée des termes

4.7 Conclusion

Ce chapitre a traité le problème de classification de textes, on y présente le principe des systèmes de classification ainsi que les méthodes et outils utilisés pour leurs réalisations. Le chapitre qui suit exposera les nouvelles métriques que nous avons proposé pour la pondération des termes dans ces systèmes.

Chapitre 5

Nouvelles métriques de pondération supervisée de termes pour la classification de textes

Sommaire

5.1 Introduction

Dans ce chapitre, nous proposons d'expérimenter un grand nombre de métriques pour la pondération de termes, ces métriques ont été proposées dans d'autres problèmes d'extraction de données afin de mesurer la corrélation entre deux événements. Nous utilisons des métriques collectées à partir d'articles traitant la sélection de descripteurs [127, 25], la pondération supervisée de termes [16, 18, 26, 69, 3], les règles de classification [31] et les collocations de termes [88]. Nous comparons expérimentalement 93 métriques pour la pondération de termes. Seulement 13 d'entre elles ont été utilisées pour le problème de pondération des termes dans la littérature et 9 sont de nouvelles métriques proposées dans cette thèse. Il semble que l'utilisation de ces métriques, au lieu de celles déjà utilisées, peut améliorer les performances des classifieurs SVM. En outre, nous montrons que la combinaison de métriques améliore la qualité de la classification. Alors que de nombreux travaux antérieurs ont montré le bien-fondé de métriques particulières [15, 16, 18, 26, 64, 69, 5, 74, 3, 81, 94, 17]. Nos expérimentations suggèrent que les résultats obtenus par ces métriques peuvent être très dépendants de la distribution des étiquettes sur le corpus et également des mesures de performances utilisées (ou micro-moyenne macro-moyenne). Cependant, nous montrons que l'utilisation d'un classifieur SVM qui combine les sorties de classifieurs SVM qui utilisent différentes métriques améliore significativement les performances de la classification de textes dans toutes les situations (distribution des étiquettes et mesures de performances utilisées).

La deuxième contribution principale de ce chapitre est la représentation étendue des termes pour le modèle d'espace vectoriel. En suivant le schéma utilisé par la métrique TFIDF, qui est le produit de la fréquence du terme (TF) et de l'inverse de la fréquence du document (IDF), des métriques supervisées alternatives ont été formulées en remplaçant l'IDF par une métrique supervisée [3]. Toutefois, la fusion par produit de TF avec un facteur tel que IDF, χ_2 ou odds ratio est problématique : on ne saisit pas pourquoi un TF deux fois plus grand serait équivalent à un IDF, χ_2 ou odds ratio deux fois plus grand. Ainsi, au lieu de mélanger la fréquence du terme avec la fréquence du document dans une même formule, nous proposons, comme alternative, de convertir le document texte en un vecteur où chaque dimension représente un descripteur de la forme (t, n), ce qui signifie que le terme t apparaît dans le document au moins n fois. Précisément, si le terme t apparaît 10 fois dans le document, nous générons tous les descripteurs (t, n) avec $n = 1, 2, 4$ et 8 (puissances de 2 afin de limiter le nombre de descripteurs). C'est pourquoi, plutôt que d'associer le poids TFIDF à un terme t, nous affectons, dans notre modèle, le poids IDF aux descripteurs (t, n) qui

dépend du nombre inverse de documents contenant au moins n fois un terme t. Dans ce modèle, la fréquence du terme dans le document est intégrée de manière adéquate à la fréquence du document. Nous proposons également un autre type de descripteur de terme étendu fondé sur l'idée que les termes qui sont les plus corrélés avec le sujet du document ont tendance à apparaître au début. Nous générons les descripteurs des positions du terme (t, p), ce qui signifie que la première position de t dans le document est inférieure ou égale à p. Les résultats expérimentaux montrent que l'utilisation de ces deux représentations étendues améliore considérablement la prédiction de la classification de textes.

5.2 Représentation étendue des termes d'un documents

La classification de textes est traditionnellement réalisée en appliquant une méthode d'apprentissage à une représentation du document textuel. Dans le modèle d'espace vectoriel, le document est représenté par un vecteur dans l'espace de terme. Chaque dimension de l'espace vectoriel représente un terme qui est la valeur de son poids. Cette valeur sera utilisée dans le processus d'apprentissage. Dans ce chapitre, nous proposons de représenter chaque dimension avec sa fréquence minimale ou sa première position minimale dans le document. Nous appelons ces alternatives, les représentations étendues de terme.

5.2.1 Terme

Dans cette représentation classique, les termes sont considérés comme les dimensions de l'espace d'apprentissage. Un terme peut être un seul mot ou une phrase (n-gramme).

5.2.2 Terme - fréquence

Le nombre d'occurrences d'un terme t dans un document d est par lui-même une propriété que nous proposons d'utiliser comme descripteur. Prenons par exemple, un terme particulier t tel que 25% des documents où t apparaît appartiennent à la classe c. Si 45% des documents où t apparaît au moins 3 fois sont dans la classe c, alors le terme t est probablement plus corrélé à la classe c lorsque sa fréquence est supérieur à 2. Par conséquent, nous proposons un descripteur de la forme (t, n) représentant les documents contenant t avec une fréquence de terme d'au moins n. Si un document

d contient dix fois un terme t, nous devons générer dix descripteurs (t, i) $(i = 1, 2, \ldots, 10)$ signifient que t se produit au moins une fois, deux fois, ..., dix fois. Cela pourrait inutilement augmenter le nombre de descripteurs, pour cela, nous considérons seulement les n puissances de 2. Si t se produit dix fois, nous générons les descripteurs $(t, 1)$, $(t, 2)$, $(t, 4)$ et $(t, 8)$. Le nombre de descripteurs de fréquences associées à un terme t qui apparaît n fois dans un document d sera $\log_2 p$ dans le pire des cas. Dans la pratique, cependant, la plupart des termes ont une fréquence basse et le nombre de descripteurs augmente modestement comme nous allons le montrer dans les expérimentations (voir section5.4).

5.2.3 Terme - position

La plupart des termes qui sont liés aux sujets principaux d'un document apparaissent au début. Afin de valider cette hypothèse nous proposons des descripteurs de la forme (t, p), ce qui signifie que la première position de t dans le document est inférieure ou égale à p. La position étant définie comme le nombre de mots précédant les occurrences du terme. Comme pour les descripteurs de fréquences de t, nous ne générons que (t, p) avec des p puissances 2. Par exemple, si un premier terme t apparaît à la position 5 dans un document de la taille de 100 mots, nous générons les descripteurs $(t, 8)$, $(t, 16)$, $(t, 32)$ et $(t, 64)$ signifient que la première position de t est inférieure ou égale à 8, 16, 32 et 64. le nombre de descripteurs associés à un terme t qui apparaît dans un document d à la première position p seront $\log_2 |d|$ dans le pire des cas, où $|d|$ est la taille de d en nombre de mots. Cependant, en utilisant les descripteurs de fréquences de terme, cela augmente modérément le nombre de descripteurs total (voir section 5.4).

5.3 Nouvelles métriques de pondérations supervisées

Les métriques de pondération supervisées tentent d'attribuer un poids élevé à un descripteur qui est particulièrement présent dans les documents appartenant à une classe. Par conséquent, une bonne métrique de pondération de termes doit être une mesure de corrélation observée entre les deux événements dans l'ensemble des documents d'apprentissage : contenir un terme et appartenir à une classe. Dans cette section, nous proposons d'utiliser un grand nombre de métriques proposées pour d'autres problèmes d'extraction de données, mais non encore utilisés pour la pondération de termes, afin de mesurer la corrélation entre les deux événements.

Le tableau 5.1 présente les 80 métriques de pondération que nous proposons en plus des 16 métriques déjà utilisées et décrites dans la table 4.3. La plupart de ces métriques sont collectées à partir d'articles traitant de la sélection de descripteurs [127, 25, 93], pondération supervisée de termes [16, 18, 26, 69, 74, 3, 94], les règles de classification [31] et collocations de termes [88]. Les 9 dernières métriques sont proposées dans cette thèse.

Metrique	Formule
Joint probability	$p(xy)$
Conditional probability	$p(y\|x) = \frac{p(xy)}{p(x*)}$
Reverse conditional probability	$p(x\|y)$
Mutual dependency	$\log \frac{p(xy)^2}{p(x*)p(*y)}$
Log frequency biased	$\log \frac{p(xy)^2}{p(x*)p(*y)} + \log p(xy)$
Normalized expectation	$\frac{2f(xy)}{f(x*)f(*y)}$
Mutual expectation	$\frac{2f(xy)}{f(x*)f(*y)} + p(xy)$
Salience	$\log \frac{p(xy)^2}{p(x*)p(*y)} + \log f(xy)$
t test	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{f(xy)(1-(f(xy)/N))}}$
z score	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{\hat{f}(xy)(1-(\hat{f}(xy)/N))}}$
Poisson significance	$\frac{\hat{f}(xy)-f(xy)\log\hat{f}(xy)+\log f(xy)!}{\log N}$
Log likelihood ratio	$-2\sum_{i,j} f_{i,j} \log \frac{f_{i,j}}{\hat{f}_{i,j}}$
Squared log likelihood ratio	$-2\sum_{i,j} f_{i,j} \frac{\log f_{i,j}^2}{\hat{f}_{i,j}}$
Russel-Rao	$\frac{a}{a+b+c+d}$
Sokal-Michiner	$\frac{a+d}{a+b+c+d}$
Rogers-Tanimoto	$\frac{a+d}{a+2b+2c+d}$
Hamann	$\frac{(a+d)-(b+c)}{a+b+c+d}$
Third Sokal-Sneath	$\frac{b+c}{a+d}$
Jaccard	$\frac{a}{a+b+c}$
First Kulczynski	$\frac{a}{b+c}$
Second Sokal-Sneath	$\frac{a}{a+2(b+c)}$
Second Kulczynski	$\frac{1}{2}(\frac{a}{a+b}\frac{a}{a+c})$
Yulle's ω	$\frac{\sqrt{ad}-\sqrt{bc}}{\sqrt{ad}+\sqrt{bc}}$
Yulle's Q	$\frac{ad-bc}{ad+bc}$
Driver-Kroeber	$\frac{a}{\sqrt{(a+b)(a+c)}}$
Fifth Sokal-Sneath	$\frac{ad}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
Pearson	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$

Baroni-Urbani	$\frac{a+\sqrt{ad}}{a+b+c+\sqrt{ad}}$
Braun-Blanquet	$\frac{a}{\max(a+b,a+c)}$
Simpson	$\frac{a}{\min(a+b,a+c)}$
Michael	$\frac{4(ad-bc)}{(a+d)^2+(b+c)^2}$
Mountford	$\frac{2a}{2bc+ab+ac}$
Fager	$\frac{a}{\sqrt{(a+b)(a+c)}}$
Unigram subtuples	$\log\frac{ad}{bc} - 3.29\sqrt{\frac{1}{a}\frac{1}{b}\frac{1}{c}\frac{1}{d}}$
U cost	$\log(1+\frac{\min(b,c)+a}{\max(b,c)+a})$
S cost	$\log(1+\frac{\min(b,c)+a}{a+1})^{-\frac{1}{2}}$
R cost	$\log(1+\frac{a}{a+b})\log(1+\frac{a}{a+c})$
T combined cost	$\sqrt{\text{U cost} \times \text{S cost} \times \text{R cost}}$
Phi	$\frac{p(xy)-p(x*)p(*y)}{\sqrt{p(x*)p(*y)(1-p(x*))(1-p(*y))}}$
Kappa	$\frac{p(xy)+p(\bar{x}\bar{y})-p(x*)p(*y)-p(\bar{x}*)p(*\bar{y})}{1-p(x*)p(*y)-p(\bar{x}*)p(*\bar{y})}$
J measure	$\max[p(xy)\log\frac{p(y\|x)}{p(*y)} + p(x\bar{y})\log\frac{p(\bar{y}\|x)}{p(*\bar{y})},$ $p(xy)\log\frac{p(x\|y)}{p(x*)} + p(\bar{x}y)\log\frac{p(\bar{x}\|y)}{p(\bar{x}*)}]$
One-way J measure	$p(xy)\log(\frac{p(y\|x)}{p(*y)}) + p(x\bar{y})\log(\frac{p(\bar{y}\|x)}{p(*\bar{y})})$
Gini index	$\max[p(x*)(p(y\|x))^2 + p(\bar{y}\|x)^2) - p(*y)^2$ $+p(\bar{x}*)(p(y\|\bar{x}))^2 + p(\bar{y}\|\bar{x})^2) - p(*\bar{y})^2,$ $p(*)(p(x\|y))^2 + p(\bar{x}\|y)^2) - p(x*)^2$ $+p(*\bar{y})(p(x\|\bar{y}))^2 + p(\bar{x}\|\bar{y})^2) - p(\bar{x}*)^2]$
One-way Gini index	$p(x*)(p(y\|x))^2 + p(\bar{y}\|x)^2) - p(*y)^2$ $+p(\bar{x}*)(p(y\|\bar{x}))^2 + p(\bar{y}\|\bar{x})^2) - p(*\bar{y})^2$
Confidence	$\max[p(y\|x), p(x\|y)]$
Laplace	$\max[\frac{f(xy)+1}{f(x*)+2}, \frac{f(xy)+1}{f(*y)+2}]$
One-way Laplace	$\frac{f(xy)+1}{f(x*)+2}$
Conviction	$\max[\frac{p(x*)p(*\bar{y})}{p(x\bar{y})}, \frac{p(\bar{x}*)p(*y)}{p(\bar{x}y)}]$
One-way conviction	$\frac{p(x*)p(*\bar{y})}{p(x\bar{y})}$
Piatersky-Shapiro	$p(xy) - p(x*)p(*y)$
Certainty factor	$\max[\frac{p(y\|x)-p(*y)}{1-p(*y)}, \frac{p(x\|y)-p(x*)}{1-p(x*)}]$
One-way certainty factor	$\frac{p(y\|x)-p(*y)}{1-p(*y)}$
Added value	$\max[p(y\|x) - p(*y), p(x\|y) - p(x*)]$
One-way added value	$p(y\|x) - p(*y)$
Collective strength	$\frac{p(xy)p(\bar{x}\bar{y})}{p(x*)p(*y)+p(\bar{x}*)p(*\bar{y})} \cdot \frac{1-p(x*)p(*y)-p(\bar{x}*)p(*\bar{y})}{1-p(xy)-p(\bar{x}\bar{y})}$
Klosgen	$\sqrt{p(xy)}\max(p(y\|x) - p(*y), p(x\|y) - p(x*))$
One-way Klosgen	$\sqrt{p(xy)}(p(y\|x) - p(*y))$

GSS coefficient	$p(xy)p(\bar{x}\bar{y}) - p(x\bar{y})p(\bar{x}y)$
Specificity	$p(\bar{y}\|\bar{x})$
Leverage	$p(y\|x) - p(x*)p(*y)$
Relative risk	$p(y\|x)/p(y\|\bar{x})$
One-way support	$p(y\|x)\log_2 \frac{p(xy)}{p(x*)p(*y)}$
Two-way support	$p(xy)\log_2 \frac{p(xy)}{p(x*)p(*y)}$
Two-way support variation	$p(xy)\log_2 \frac{p(xy)}{p(x*)p(*y)} + p(x\bar{y})\log_2 \frac{p(x\bar{y})}{p(x*)p(*\bar{y})}$ $+p(\bar{x}y)\log_2 \frac{p(\bar{x}y)}{p(\bar{x}*)p(*y)+p(\bar{y}\|\bar{x})}\log_2 \frac{p(\bar{x}\bar{y})}{p(\bar{x}*)p(*\bar{y})}$
Loevinger	$1 - \frac{p(x*)p(*\bar{y})}{p(x\bar{y})}$
Sebag-Schoenauer	$\log \frac{p(xy)}{p(x*)p(*y)}$
Least contradiction	$\frac{p(xy)-p(x\bar{y})}{p(*y)}$
Odd multiplier	$\frac{p(xy)p(x\bar{y})}{p(*y)p(x\bar{y})}$
Example and counterexample rate	$1 - \frac{p(x\bar{y})}{p(xy)}$
Zhang	$\frac{p(xy)-p(x*)p(*y)}{\max(p(xy)p(*\bar{y}),p(*y)p(x\bar{y}))}$
Weighted log likelihood ratio	$p(x\|y)\log(\frac{p(x\|y)}{p(x\|\bar{y})})$
Document TFIDF	$\frac{f(xy)}{f(*y)}\log \frac{N}{f(x*)}$
Reverse-way document TFIDF	$\frac{f(xy)}{f(x*)}\log \frac{N}{f(*y)}$
Conditional probability variation 1	$p(y\|x)p(\bar{y}\|\bar{x})$
Conditional probability variation 2	$p(y\|x) + p(\bar{y}\|\bar{x})$
Conditional probability variation 3	$p(y\|x) + p(*y)$
Conditional probability variation 4	$\frac{p(y\|x)+p(\bar{y}\|\bar{x})+p(\bar{x}\|\bar{y})}{3}$
Conditional probability variation 5	$p(x\|y) - p(x\|\bar{y})$
Relevance frequency variation	$\log_2(2 + \frac{a}{\max(b,1)}) \times \log_2(2 + \frac{a}{\max(c,1)})$
Reverse-way BNS	$\|F^{-1}(p(y\|x)) - F^{-1}(p(y\|\bar{x}))\|$ [1]

TABLE 5.1: Nouvelles métriques utilisées pour la pondération de termes supervisée

5.4 Tests et résultats

5.4.1 Benchmarks

Afin de comparer expérimentalement les métriques, nous utilisons trois corpus : Reuters-21578, Ohsumed et 20 NewsGroup. Ces ensembles de données sont les benchmarks les plus largement utilisés pour la classification de textes.

1. F^{-1} est l'inverse de la fonction de répartition de la loi normale.

La distribution des classes du corpus Reuters-21578 est fortement déséquilibrée. Afin d'étudier les performances obtenues par chaque métrique de pondération dans des situations plus ou moins asymétriques nos résultats sur Reuters-21578 sont rapportés :

— pour les 115 classes avec au moins un exemple d'apprentissage,
— pour les 52 classes, avec au moins 16 exemples d'apprentissage,
— et pour l'ensemble des 10 classes avec le plus grand nombre d'exemples d'apprentissage.

Ohsumed est un corpus de résumés médicaux cardiovasculaires avec 23 classes de maladies. Le 20 Newsgroups est un corpus qui contient des articles pris à partir de 20 groupes de discussion de Usenet (classes). Les variations d'un terme peuvent affecter sa fréquence qui est un paramètre important dans le poids du terme, la solution consiste à remplacer chaque mot par sa racine. Dans tout le corpus, nous avons utilisé PorterStemmer [89] qui donne les meilleures performances dans nos expérimentations. Après la racinisation, nous avons segmenter les documents textuels. Pour chaque phrase dans un document, nous générons tous les n-grammes possibles (termes). Nous choisissons la taille des n-grammes selon les performances obtenues dans chaque corpus. Pour le corpus Reuters-21578 la taille de n a été fixée à 1, pour les corpus Ohsumed et 20 Newsgroups nous avons fixé $n \leq 2$. Nous avons utilisé le découpage proposé dans les corpus Reuters-21578 (split Mobapte) et Ohsumed. Il n'y a pas de découpage fixe dans la littérature pour les 20 Newsgroup. Il est généralement utilisé pour la validation croisée. Nous avons adopté la validation croisée (5 partitions) sur le corpus 20 NewsGroup afin d'évaluer la signification statistique des performances obtenues. Traditionnellement, la performance d'un classifieur sur un corpus est estimée par l'apprentissage de la classification sur les données d'apprentissage et l'évaluation de la qualité de prédiction obtenue sur les données d'évaluation. Les métriques d'évaluation utilisées sont la précision qui est la proportion de documents placés dans la classe qui sont vraiment dans la classe et le rappel qui est la proportion de documents de la classe qui sont effectivement placés dans la classe, et le F_1-score est définie comme suit :

$$\text{F}_1\text{-score} = 2 \times \frac{\text{précision} \times \text{rappel}}{\text{précision} + \text{rappel}}$$

La précision et le rappel sont calculés en comptant le nombre de classes vraies/fausses positives/négatives trouvés (voir la table 5.2 et l'équation 5.1).

$$\textbf{précision} = \frac{\textbf{TP}}{\textbf{TP+FP}} \ \textbf{rappel} = \frac{\textbf{TP}}{\textbf{TP+FN}} \tag{5.1}$$

La micro-moyenne du F_1-score est calculée globalement pour toutes les

		classe réelle	
		oui	non
classe prédite	oui	**VP** (vrai positif)	**FP** (faux positif)
	non	**VN** (vrai negatif)	**FN** (faux negatif)

TABLE 5.2 – Définitions des classes vraies/fausses positives/négatives.

classes, tandis que la macro-moyenne du F_1-score est la moyenne des F_1-scores calculée pour chaque classe. Cette dernière mesure la capacité d'un classifieur à bien classer lorsque la distribution des classes est déséquilibrée, alors que la micro-moyenne du F_1-score donne une vue globale de la performance de la classification des documents.

5.4.2 Apprentissage

Dans la classification multi-label de textes chaque document d appartient à une ou plusieurs catégories dans $C = \{c_1, c_2, \ldots, c_l\}$. Afin de faire de l'apprentissage pour la classification multi-label, nous avons utilisé la stratégie de pertinence binaire (Binary Relevance BR)[109, 73, 131], c'est une méthode de transformation classique, à savoir, un-versus-tous qui apprend $|C|$ classifieurs binaires indépendants, un pour chaque catégorie. Chaque classifieur binaire donne une probabilité que d appartienne ou non à la catégorie $y = c_i$.

Pondération

Pour chaque classifieur binaire associé à la catégorie y, chaque document d est transformé en un vecteur $W_d = (w(x_1, y, d), w(x_2, y, d), \ldots, w(x_n, y, d))$ où chaque descripteur x est pondéré par :

$$w(x, y, d) = w_{TF}(x, d) \times w_{DF}(x, y)$$

La pondération fréquence du terme w_{TF} dépend de la fréquence de x dans le document d. La pondération fréquence du document w_{DF} est l'une des métriques décrites dans la table 5.1.

Chaque descripteur x peut être soit :

— un terme t dans le modèle classique,

— ou (t, n) ou (t, p) dans le modèle de représentation étendue comme définie dans la section 5.2.1.

Pour la classification classique, suivant [64], nous avons expérimenté trois pondérations fréquence du terme possibles (voir la table 5.3). Pour notre

Fréq. du terme	Valeur	Description
BIN(x,d)	1 si $tf(x,d) > 0$, 0 sinon	pondération binaire
RTF(x,d)	$tf(x,d)$	fréquence du terme brute
LTF(x,d)	$\log(1 + tf(x,d))$	logarithme de la fréquence du terme
ITF(x,d)	$1 - \frac{1}{1-tf(x,d)}$	inverse de la fréquence du terme

TABLE 5.3 – Les pondérations fréquence de terme expérimentées comme fonctions de fréquence $tf(x,d)$ de x dans d.

modèle, nous avons seulement utilisé la pondération fréquence du terme binaire ($w_{TF}(x,d) = \text{BIN}(x,d)$), car la fréquence du terme est déjà considérée dans la représentation étendue du terme $x = (t,n)$.

Classifieur SVM

Pour chaque catégorie, nous avons utilisé un classifieur binaire SVM qui apprend une combinaison linéaire des descripteurs afin de définir l'hyper plan décisionnel. Nous avons opté pour l'outil SVMLight [51] avec un noyau linéaire et utilisé les paramètres par défauts. Des études montrent que SVM-Light donne de bonnes performances pour la classification de textes [33].

Combinaison de classifieurs SVM

La méthode de combinaison de classifieurs [110] est une technique d'ensemble d'apprentissage qui utilise les prédictions de plusieurs classifieurs afin d'obtenir une meilleure performance de prédiction. Une manière de combiner des classifieurs est de considérer les sorties des différents classifieurs comme entrées d'un classifieur générique appelé classifieur secondaire.

Dans notre cas, pour chaque catégorie nous combinons les scores données par les différents classifieurs binaires avec un classifieur SVM. Chaque classifieur utilise l'une des 96 métriques de pondération fréquence de document. Les Classifieurs SVM de bases opèrent sur le même ensemble d'apprentissage. Les scores de classification obtenus pour chaque document de l'ensemble d'apprentissage sont utilisés comme descripteurs en entrées pour le classifieur secondaire SVM.

Nous avons aussi testé comme classifieur secondaire, les forêts aléatoires mais c'est l'approche SVM qui donne les meilleurs résultats.

Corpus	Représentation du terme Poids fréquence du terme	t RTF	t LTF	t ITF	(t,n) BIN	(t,n) &(t,p) BIN
Reuters 10 cat.	Poids fréquence du document					
1	Yulle's ω	0.912	0.921	0.926	0.927	**0.947**
2	Log odds ratio	0.913	0.922	0.927	0.929	**0.947**
3	Forman log odds ratio	0.913	0.922	0.927	0.929	**0.947**
4	Yulle's Q	0.909	0.919	0.924	0.925	**0.946**
5	Pointwise mutual information	0.909	0.920	0.923	0.929	**0.946**
6	Unigram subtuples	0.911	0.922	0.926	0.928	**0.946**
7	Bi-normal separation	0.915	0.922	0.927	0.928	**0.946**
8	δIDF	0.911	0.920	0.926	0.929	**0.945**
9	Zhang	0.908	0.918	0.921	0.923	**0.945**
10	Reverse-way BNS	0.911	0.920	0.924	0.924	**0.944**
...						
30	IDF	0.854	0.886	***0.892***	0.888	**0.932**
Reuters 52 cat.	Poids fréquence du document					
1	Bi-normal separation	0.843	0.861	0.867	0.878	**0.893**
2	Log odds ratio	0.837	0.855	0.860	0.872	**0.891**
3	Forman log odds ratio	0.837	0.855	0.860	0.872	**0.891**
4	Reverse-way BNS	0.834	0.853	0.855	0.870	**0.890**
5	Probability based term weight	0.821	0.847	0.853	0.862	**0.889**
6	δIDF	0.833	0.853	0.858	0.869	**0.889**
7	Pointwise mutual information	0.826	0.853	0.856	0.864	**0.889**
8	Yulle's ω	0.830	0.854	0.859	0.868	**0.888**
9	Collective strength	0.820	0.842	0.849	0.858	**0.887**
10	Relevance frequency variation	0.826	0.848	0.854	0.866	**0.886**
...						
76	IDF	0.750	0.797	***0.810***	0.820	**0.864**
Reuters 115 cat.	Poids fréquence du document					
1	Bi-normal separation	0.852	0.865	0.870	**0.883**	0.877
2	Log odds ratio	0.848	0.865	0.870	**0.880**	0.875
3	Forman log odds ratio	0.848	0.865	0.870	**0.880**	0.875
4	Relevance frequency$_{OR}$	0.834	0.851	0.854	**0.878**	0.869
5	Conviction	0.835	0.847	0.852	**0.876**	0.861
6	Yulle's ω	0.843	0.861	0.865	**0.875**	0.870
7	Relevance frequency	0.828	0.849	0.850	**0.875**	0.865
8	Reverse-way BNS	0.847	0.861	0.865	0.874	**0.875**
9	S cost	0.833	0.850	0.853	**0.874**	0.864
10	One-way conviction	0.831	0.844	0.849	**0.874**	0.854
...						
75	IDF	0.790	0.823	***0.833***	**0.849**	0.845
Ohsumed	Poids fréquence du document					
1	One-way Klosgen	0.587	0.604	0.609	0.631	**0.639**
2	Klosgen	0.586	0.600	0.605	0.626	**0.636**
3	z score	0.578	0.601	0.605	0.624	**0.634**
4	Pearson	0.577	0.600	0.604	0.624	**0.633**
5	Phi	0.577	0.600	0.604	0.624	**0.633**
6	Squared log likelihood ratio	0.558	0.585	0.593	0.621	**0.631**
7	Odds ratio	0.563	0.582	0.590	0.617	**0.629**
8	One-way Gini index	0.593	0.598	0.600	0.618	**0.629**
9	Pearson's χ^2 test	0.593	0.598	0.600	0.618	**0.629**
10	Sebag-Schoenauer	0.560	0.581	0.588	0.616	**0.628**
...						
81	IDF	0.296	0.363	***0.380***	0.417	**0.444**
20 Newsgroups	Poids fréquence du document					
1	One-way Klosgen	0.731	0.759	0.764	0.767	**0.790**
2	z score	0.713	0.755	0.762	0.767	**0.790**
3	Pearson	0.708	0.753	0.761	0.766	**0.788**
4	Phi	0.708	0.753	0.761	0.766	**0.788**
5	Bi-normal separation	0.664	0.737	0.747	0.749	**0.785**
6	Reverse-way BNS	0.666	0.737	0.747	0.750	**0.783**
7	One-way Laplace	0.671	0.733	0.742	0.748	**0.782**
8	Klosgen	0.706	0.742	0.750	0.758	**0.781**
9	Relative risk	0.692	0.743	0.751	0.756	**0.781**
10	Second Kulczynski	0.672	0.736	0.746	0.753	**0.781**
...						
78	IDF	0.396	0.589	***0.617***	0.628	**0.708**

TABLE 5.4 – La micro-moyenne du F_1-score de la méthode SVM avec différentes représentations des termes et métriques de pondération (top-10 des métriques et scores IDF) pour les corpus Reuters (10, 52 et 115 classes), Ohsumed et 20 Newsgroups

Corpus	Représentation des termes Poids fréquence du terme	t RTF	t LTF	t ITF	(t,n) BIN	(t,n) &(t,p) BIN
Reuters 10 cat.	Poids fréquence du document					
1	Unigram subtuples	0.836	0.864	0.874	0.881	**0.898**
2	Bi-normal separation	0.857	0.868	0.876	0.882	**0.897**
3	Log odds ratio	0.851	0.868	0.876	0.878	**0.896**
4	Forman log odds ratio	0.851	0.868	0.876	0.878	**0.896**
5	Pointwise mutual information	0.844	0.864	0.869	0.880	**0.895**
6	Yulle's ω	0.849	0.867	0.874	0.880	**0.894**
7	Yulle's Q	0.840	0.860	0.871	0.873	**0.892**
8	Zhang	0.834	0.857	0.862	0.869	**0.892**
9	δIDF	0.846	0.864	0.871	0.878	**0.891**
10	Collective strength	0.825	0.850	0.856	0.867	**0.890**
...						
73	IDF	0.743	0.807	***0.817***	0.814	**0.862**
Reuters 52 cat.	Poids fréquence du document					
1	Kappa	0.615	0.645	0.661	0.706	**0.765**
2	Normalized expectation	0.612	0.647	0.657	0.710	**0.763**
3	One-way Klosgen	0.585	0.620	0.639	0.691	**0.762**
4	Bi-normal separation	0.645	0.686	0.686	0.721	**0.762**
5	Jaccard	0.614	0.646	0.655	0.706	**0.761**
6	Poisson significance	0.630	0.660	0.670	0.723	**0.760**
7	Log likelihood ratio	0.627	0.651	0.658	0.692	**0.760**
8	J measure	0.627	0.654	0.659	0.693	**0.759**
9	One-way J measure	0.626	0.647	0.661	0.705	**0.759**
10	One-way Gini index	0.593	0.624	0.637	0.695	**0.759**
...						
86	IDF	0.380	0.500	***0.520***	0.576	**0.625**
Reuters 115 cat.	Poids fréquence du document					
1	Relevance frequency$_{OR}$	0.443	0.498	0.493	**0.574**	0.489
2	Bi-normal separation	0.474	0.514	0.533	**0.566**	0.508
3	Poisson significance	0.504	0.513	0.522	**0.565**	0.546
4	Log odds ratio	0.465	0.509	0.524	**0.563**	0.495
5	Forman log odds ratio	0.465	0.509	0.524	**0.563**	0.495
6	Pearson	0.482	0.509	0.521	**0.562**	0.539
7	Phi	0.482	0.509	0.521	**0.562**	0.539
8	Conviction	0.465	0.508	0.529	**0.561**	0.510
9	S cost	0.444	0.501	0.510	**0.560**	0.477
10	Mutual expectation	0.481	0.458	0.432	0.512	**0.557**
...						
81	IDF	0.362	0.417	***0.426***	**0.474**	0.380
Ohsumed	Poids fréquence du document					
1	One-way Klosgen	0.538	0.569	0.575	0.595	**0.602**
2	Squared log likelihood ratio	0.540	0.557	0.564	0.589	**0.601**
3	Klosgen	0.536	0.565	0.570	0.591	**0.598**
4	One-way Gini index	0.553	0.562	0.567	0.586	**0.598**
5	Pearson's χ^2 test	0.553	0.562	0.568	0.587	**0.598**
6	Odds ratio	0.520	0.545	0.553	0.576	**0.594**
7	z score	0.523	0.556	0.562	0.584	**0.592**
8	Pearson	0.522	0.556	0.562	0.583	**0.591**
9	Phi	0.522	0.556	0.562	0.583	**0.591**
10	J measure	0.537	0.552	0.555	0.561	**0.591**
...						
85	IDF	0.185	0.237	***0.255***	0.289	**0.319**
20 Newsgroups	Poids fréquence du document					
1	One-way Klosgen	0.737	0.764	0.770	0.773	**0.795**
2	z score	0.718	0.761	0.768	0.773	**0.795**
3	Pearson	0.713	0.758	0.766	0.771	**0.793**
4	Phi	0.713	0.758	0.766	0.771	**0.793**
5	Bi-normal separation	0.669	0.743	0.753	0.754	**0.789**
6	Reverse-way BNS	0.672	0.743	0.752	0.755	**0.788**
7	Klosgen	0.712	0.748	0.757	0.765	**0.787**
8	One-way Laplace	0.676	0.738	0.747	0.754	**0.786**
9	Relative risk	0.697	0.749	0.756	0.761	**0.786**
10	Second Kulczynski	0.678	0.741	0.752	0.759	**0.785**
...						
78	IDF	0.403	0.595	***0.623***	0.633	**0.712**

TABLE 5.5 – La macro-moyenne du F_1-score de la méthode SVM avec différentes représentations des termes et métriques de pondération (top-10 des métriques et scores IDF) pour les corpus Reuters (10, 52 et 115 classes), Ohsumed et 20 Newsgroups

5.4.3 Tests et résultats

Afin d'estimer, à la fois, la performance de notre modèle et les performances des 96 métriques, nous avons comparé le $F1$-Score du classifieur SVM sur les documents de Reuters-21578, Ohsumed et 20 NewsGroup avec les représentations classiques et étendues de terme à l'aide de schémas de pondérations différents.

a- Comparaison des métriques

Nous rappelons que pour une classe fixe y le poids $w(x, y, d)$ d'un descripteur x dans un document d est :

$$w(x, y, d) = w_{TF}(x, d) \times w_{DF}(x, y)$$

où le poids de la fréquence du terme w_{TF} (voir le tableau 5.3) dépend de la fréquence de x dans le document d et le poids de la fréquence du document w_{DF} sera l'une des métriques décrites dans le tableau 5.1. x représente soit un descripteur terme t, soit un descripteur fréquence des termes (t, n) ou un descripteur position des termes (t, p).

Pour chaque poids de fréquence de documents w_{DF}, nous avons expérimenté 5 schémas de pondérations :

- pondération avec la fréquence du terme brute (*raw term frequency*) ($w_{TF} = \text{RTF}$) avec les termes t comme descripteurs.
- pondération avec le logarithme de la fréquence du terme ($w_{TF} = \text{LTF}$) avec les termes t comme descripteurs.
- pondération avec l'inverse de la fréquence du terme ($w_{TF} = \text{ITF}$) avec les termes t comme descripteurs.
- pondération avec la fréquence binaire du terme ($w_{TF} = \text{BIN}$) avec les fréquences de termes (t, n) comme descripteurs.
- pondération avec la fréquence binaire du terme ($w_{TF} = \text{BIN}$) avec les fréquences de termes (t, n) et positions de terme (t, p) comme descripteurs.

La table 5.4 rend compte de la micro-moyenne du F_1-Score pour les corpus Reuters-21578 (10, 52 et 115 classes), Ohsumed et 20 Newsgroups. Après le calcul du F_1-Score pour chaque classifieur, les métriques sont classées par ordre décroissant des scores par rapport aux schémas de pondérations. La table montre seulement le top-10 des métriques. On observe clairement que les modèles de représentation proposés (t, n) et $(t, n)\&(t, p)$ sont nettement plus performants que la représentation classique et atteignent les meilleures performances dans toutes les expérimentations en termes de micro-moyenne

du F_1-scores avec toutes les métriques. Le modèle $(t, n)\&(t, p)$ est plus performant que le modèle (t, n), ce qui signifie que l'utilisation de la position améliore les performances. La seule exception est Reuters-21578 avec 115 classes. Nous pensons que cela est dû au fait qu'un nombre important de classes (40/115) sont représentées par seulement au plus 3 documents d'apprentissage, l'influence de la position dans le document ne peut pas être apprise correctement.

Ces observations confortent notre intuition que l'inclusion de la fréquence du terme dans la formule de la fréquence de documents (comme une caractéristique) est plus pertinent que de multiplier ces quantités.

Dans le modèle de représentation de termes classique, le poids de l'inverse de la fréquence de termes (ITF) donne de meilleurs F_1-scores que le poids de la fréquence des termes brutes et logarithme (RTF et LTF). Nous remarquons aussi que la métrique de base TFIDF avec la représentation classique, précisément ITF.IDF, effectue un F_1-score sensiblement plus mauvais que d'autres métriques. En utilisant d'autres métriques que TFIDF et avec la représentation de terme étendue, cela donne une amélioration significative de la classification. Par exemple, le F1-score augmente de 0,892 pour TFIDF à 0,947 pour Yulle's ω avec la représentation $(t, n)\&(t, p)$ dans le corpus Reuters-21578 avec 10 classes. La meilleure performance est obtenue avec le corpus Ohsumed, le F1- Score augmente de 0,380 à 0,639 avec la métrique one-way Klosgen et la représentation étendue de terme. Cependant, nous remarquons que les meilleures métriques sont très différentes selon le corpus utilisé et selon que la distribution des étiquettes est équilibrée ou pas pour le corpus Reuters-21578.

La table 5.5 montre les macro-moyennes du F_1-score des top-10 des métriques parmi tous les schémas de pondérations. Nous observons également que les modèles de représentation proposés (t, n) et (t, n) & (t, p) parviennent à des meilleurs macro-moyennes du F_1-score. Toutefois, le top-10 des métriques, lorsque nous considérons les macro-moyennes du F_1-score, sont généralement différentes du top-10 des métriques des micro-moyennes du F_1-score. Nous remarquons aussi que, dans ce travail, beaucoup de métriques sont proposées pour la première fois pour la pondération de termes et qu'elles donnent de meilleurs résultats que les métriques précédemment utilisées pour ce problème.

b- Combinaison de classifieurs

Comme aucune métrique ne donne les meilleurs résultats dans toutes les situations, nous avons testé la combinaison de classifieurs.

La table 5.6 et la table 5.7 montrent les F_1-Scores obtenus par différentes métriques de pondération et leurs combinaisons lorsque nous utilisons la

Reuters 10 categories			
Poids fréquence du document	MiF	Poids fréquence du document	MaF
Combinaison	**0,952**	**Combinaison**	**0,910**
Yulle's ω	0.947	Unigram subtuples	0.898
Log odds ratio	0.947	Bi-normal separation	0.897
Forman log odds ratio	0.947	Log odds ratio	0.896
Yulle's Q	0.946	Forman log odds ratio	0.896
Pointwise mutual information	0.946	Pointwise mutual information	0.895
Unigram subtuples	0.946	Yulle's ω	0.894
Bi-normal separation	0.946	Yulle's Q	0.892
δIDF	0.945	Zhang	0.892
Zhang	0.945	δIDF	0.891
Reverse-way BNS	0.944	Collective strength	0.890
Reuters 52 categories			
Poids fréquence du document	MiF	Poids fréquence du document	MaF
Combinaison	**0.905**	**Combinaison**	**0.772**
Bi-normal separation	0.893	Kappa	0.765
Log odds ratio	0.891	Normalized expectation	0.763
Forman log odds ratio	0.891	One-way Klosgen	0.762
Reverse-way BNS	0.890	Bi-normal separation	0.762
Probability based term weight	0.889	Jaccard	0.761
δIDF	0.889	Poisson significance	0.760
Pointwise mutual information	0.889	Log likelihood ratio	0.760
Yulle's ω	0.888	J measure	0.759
Collective strength	0.887	One-way J measure	0.759
Relevance frequency variation	0.886	One-way Gini index	0.759
Reuters 115 categories			
Poids fréquence du document	MiF	Poids fréquence du document	MaF
Combinaison	**0.888**	**Combinaison**	**0.570**
Bi-normal separation	0.877	Mutual expectation	0.557
Log odds ratio	0.875	One-way Gini index	0.554
Forman log odds ratio	0.875	Second Sokal-Sneath	0.553
Reverse-way BNS	0.875	Pearson's χ^2 test	0.553
Probability based term weight	0.873	Jaccard	0.552
Pointwise mutual information	0.873	First Kulczynski	0.551
Relevance frequency variation	0.872	R cost	0.551
z score	0.872	T combined cost	0.551
Pearson	0.871	One-way Klosgen	0.551
Phi	0.871	One-way J measure	0.547
Ohsumed			
Poids fréquence du document	MiF	Poids fréquence du document	MaF
Combinaison	**0.679**	**Combinaison**	**0.647**
One-way Klosgen	0.639	One-way Klosgen	0.602
Klosgen	0.636	Squared log likelihood ratio	0.601
z score	0.634	Klosgen	0.598
Pearson	0.633	One-way Gini index	0.598
Phi	0.633	Pearson's χ^2 test	0.598
Squared log likelihood ratio	0.631	Odds ratio	0.594
Odds ratio	0.629	z score	0.592
One-way Gini index	0.629	Pearson	0.591
Pearson's χ^2 test	0.629	Phi	0.591
Sebag-Schoenauer	0.628	J measure	0.591

TABLE 5.6 – Micro et macro-moyennes du F_1-score de la méthode SVM et de la combinaison des SVM avec la représentation de terme étendue (t, n) & (t, p) avec différentes métriques de pondération pour les corpus Reuters (10, 52 and 115 classes), Ohsumed et 20 Newsgroups

20 Newsgroups			
Poids fréquence du document	MiF	Poids fréquence du document	MaF
Combinaison	**0.861 ± 0.008**	**Combinaison**	**0.866 ± 0.006**
One-way Klosgen	0.790 ± 0.007	One-way Klosgen	0.795 ± 0.006
z score	0.790 ± 0.008	z score	0.795 ± 0.005
Pearson	0.788 ± 0.009	Pearson	0.793 ± 0.006
Phi	0.788 ± 0.009	Phi	0.793 ± 0.006
Bi-normal separation	0.785 ± 0.012	Bi-normal separation	0.789 ± 0.010
Reverse-way BNS	0.783 ± 0.010	Reverse-way BNS	0.788 ± 0.009
One-way Laplace	0.782 ± 0.009	Klosgen	0.787 ± 0.004
Klosgen	0.781 ± 0.006	One-way Laplace	0.786 ± 0.007
Relative risk	0.781 ± 0.009	Relative risk	0.786 ± 0.007
Second Kulczynski	0.781 ± 0.007	Second Kulczynski	0.785 ± 0.006

TABLE 5.7 – Performances de la validation croisée (moyenne et écart type) des méthodes SVM et leur combinaison avec la représentation de termes étendue (t, n) & (t, p) pour différentes métriques de pondération avec les corpus 20 Newsgroups

représentation étendue de terme (t, n) & (t, p) sur les corpus Reuters (10, 52 et 115 classes), Ohsumed et 20 Newsgroups. Nous pouvons voir que les performances de la combinaison de classifieurs est toujours meilleure en tenant compte de tous les critères : micro et macro-moyennes du F_1-score pour tous les corpus. Cela confirme qu'en combinant les prédictions de plusieurs classifieurs utilisant différentes métriques, on obtient une meilleure performance prédictive que celle qui pourrait être obtenue avec tous les classifieurs qui utilisent une seule métrique. La signification statistique des performances réalisées sur le corpus 20 Newsgroups est donnée dans le tableau 5.7 . Outre le fait que le F_1-Score obtenu par la combinaison de classifieurs est plus (dans 20 Newsgroups) ou moins (dans Reuters avec 10 classes) meilleure que la meilleure métrique pour chaque corpus, la combinaison est la seule méthode qui donne de bons résultats pour tous les corpus, tout nombre de classes et avec les deux types de F1-Scores (micro et macro).

c- Temps de calcul et complexité

Le tableau 5.8 donne le nombre de descripteurs pris en compte dans l'ensemble d'apprentissage selon les représentations étendues de termes que nous avons considéré dans nos expérimentations. Il montre une croissance modérée du nombre de descripteurs, par un facteur de trois, si l'on considère les descripteurs fréquence du terme et position du terme par rapport au descripteur terme traditionnel. Il est intéressant de noter dans le même tableau que le temps de calcul de SVM (sur un processeur d'un Intel (R) Core (TM) i7-3520M à 2,90 GHz) est à peu près proportionnel au nombre de descripteurs. En effet, Thorsten Joachims a montré que l'apprentissage linéaire SVM peut

	Type descr.	Nombre desc.	Temps (en sec.)
Reuters-21578			
Terme	t	20 767	0.203
Frequence du terme	(t, n)	32 690	0.283
Frequence du terme et position	(t, n) & (t, p)	61 096	0.627
Ohsumed			
Terme	t	175 803	2.254
Frequence du terme	(t, n)	223 589	3.236
Frequence du terme et position	(t, n) & (t, p)	500 474	8.082
20 Newsgroups			
Terme	t	393 797	8.306
Frequence du terme	(t, n)	462 370	21.083
Frequence du terme et position	(t, n) & (t, p)	861 749	33.028

TABLE 5.8 – Nombre de descripteurs considérés dans l'ensemble d'apprentissage des corpus Reuters-21578, Ohsumed et 20 Newsgroups et la moyenne du temps de calcul de l'apprentissage avec SVM pour une métrique et une classe pour chaque corpus

être réalisé en temps $O(ns)$, où s est le nombre non nul de descripteurs (termes) dans chaque exemple (document) et n le nombre d'exemples [52], avec l'hypothèse que $s \ll N$, N étant le nombre de descripteurs dans l'ensemble du corpus. Cette dernière hypothèse est vérifiée pour un corpus de documents textuels.

Comme nous l'avons déjà indiqué dans les sections 5.2.2 et 5.2.3 le nombre de descripteurs de fréquence et de position associés à un terme t dans chaque document augmente en logarithme de s. Cela signifie que la complexité en temps pour l'apprentissage d'un corpus de documents contenant au plus s termes est $O(ns \log s)$. Dans la pratique, $\log s$ est faible, comme nous pouvons le constater, à la fois par le nombre de descripteurs et le temps de calcul présenté dans le tableau 5.8. Notre méthode de combinaison de classifieurs implique l'utilisation de 96 SVM dans la première étape, puis un autre SVM pour le classement final. La complexité du temps théorique n'est pas affectée, car 96 est une valeur constante. Toutefois, en pratique, cela signifie qu'il multiplie le temps de calcul par 96. Dans de nombreux problèmes de classification de textes on peut se permettre une telle croissance constante de temps de calcul pour obtenir une amélioration significative de la qualité de la classification (à partir d'une micro-moyenne du F_1-score de 0,444 avec IDF à 0,679 en combinant 96 métriques dans le cas du corpus Ohsumed).

5.5 Conclusion

Dans ce chapitre, nous avons étudié 96 métriques de pondération des termes, parmi elles 80 n'ont pas été utilisées, pour ce problème, dans la littérature. Beaucoup d'entre elles fournissent de meilleurs résultats que celles déjà utilisées pour la pondération de termes. Nous avons également proposé une représentation étendue où la fréquence du terme et sa position dans le document sont intégrées de manière adéquate à la fréquence du document. Comme aucune métrique ne donne les meilleurs résultats selon que la distribution des étiquettes est équilibrée ou pas, nous avons proposé une méthode de combinaison de classifieurs avec différentes métriques qui fonctionne bien à la fois pour les micro et macro-moyennes du F_1-score dans les différents cas de distribution des étiquettes.

Conclusion

Dans le cadre de cette thèse je me suis intéressée à deux problèmes de fouille de textes : l'extraction automatique de termes-clés dans des documents textuels et la classifications de textes. Étant donné un document d et un entier k, dans le premier problème, on recherche les k termes qui caractérisent le mieux le document d. Les termes ne sont donc pas prédéfinis et apparaissent tous dans d. Alors que dans le second problème, on cherche à classer le document d dans k catégories prédéfinies, k est variable selon le document et une catégorie n'apparait pas forcément dans d.

Pour le problème d'extraction automatique de termes-clés, j'ai proposé une nouvelle mesure le DPM-index qui discrimine les phrases (n-grammes) qui se chevauchent dans un document [34]. J'ai aussi développé un nouveau système d'extraction de termes-clés basé sur l'apprentissage supervisé qui combine 18 descripteurs statistiques parmi lesquels le DPM-index. J'ai montré expérimentalement, que la contribution de cette dernière est comparables aux descripteurs de niveau corpus pour l'extraction de 10 termes-clés. Les résultats de ce système montrent une amélioration par rapport aux autres systèmes sans utiliser aucune base de connaissance externe ou descripteurs structuraux, ce qui rend mon système, me semble t-il, plus souple et adaptable à d'autres types de corpus. Une phase de prétraitement des documents textuels qui consiste à filtrer les termes candidats du document avant de les soumettre au système d'extraction peut affecter de manière significative la précision de la méthode d'extraction, pour cela, j'ai amélioré le filtrage linguistique habituellement utilisé dans ce type de système [35].

Pour le problème de classification de textes, j'ai étudié 96 métriques de pondération de termes [36], parmi elles 80 n'ont pas été utilisées, pour ce problème, dans la littérature. Beaucoup d'entre elles fournissent de meilleurs résultats que celles déjà utilisées pour la pondération de termes en particulier dans la classification d'articles scientifiques biomédicaux [37]. J'ai également proposé une représentation étendue où la fréquence du terme et sa position dans le document sont intégrées de manière adéquate à la fréquence du document. Comme aucune métrique ne donne les meilleurs résultats selon que

la distribution des l'étiquettes du corpus est équilibrée ou non, j'ai proposé une méthode de combinaison de classifieurs avec différentes métriques qui donne de bons résultats en termes de performances (macro-moyenne et micro-moyenne du F_1-score) et pour différents cas de distribution des étiquettes.

La thèse comporte différentes contributions :

— La première contribution est la proposition de 6 nouvelles métriques, parmi elles le DPM-index qui permet de discriminer parmi les phrases qui se chevauchent (n-grammes) dans un document pour l'extraction automatique de termes-clés.

— La seconde contribution est la proposition d'un système d'extraction automatique de termes-clés basé sur l'apprentissage supervisé qui utilise 18 descripteurs statistiques, parmi eux la métrique DPM-index et cinq autres nouvelles métriques. Après la sélection des termes-clés candidats à l'aide d'information linguistique (étiquettes syntaxiques), le système offre la possibilité d'évaluer l'efficacité de la contribution de chaque descripteur statistique.

— La troisième contribution est la proposition d'un nouveau filtre linguistique au niveau de la phase de prétraitement des textes dans lesquels on veut extraire les terme-clés. Il s'agit d'un nouveau patron syntaxique pour la sélection des termes candidats.

— La quatrième contribution est la proposition de 80 métriques de pondérations des termes pour la classification de textes. Parmi elles, 9 sont une proposition originale et les autres ont été utilisées dans d'autres problèmes de recherche d'information. Ces métriques sont comparées à 13 autres métriques déjà utilisées dans la littérature.

— La cinquième contribution est la proposition d'une représentation étendue des termes pour le modèle d'espace vectoriel qui permet d'améliorer considérablement la prédiction du classifieur de textes. Afin de confronter les différents résultats obtenus, une étude expérimentale a été menée pour évaluer les performances des méthodes proposées en les comparant avec d'autres méthodes étudiées.

Les résultats obtenus ont donné lieu à deux publications dans des revues de rang A et de deux communications dans des conférences internationales avec actes et comité de lecture [34, 36, 35, 37].

Perspectives de recherche

Les perspectives de ce travail concernent les deux aspects de la thèse, extraction et classification. Au début de ma thèse je me suis intéressée au problème de l'extraction de termes-clés par le biais d'une application parti-

culière qui est la recommandation de tags dans le web basée sur le contenu [76, 91]. Au fur et à mesure de l'avancement de ma recherche bibliographique et expérimentale je me suis rendue compte que les solutions que je proposais peuvent être généralisées à l'extraction de termes dans tous types de textes. Après ce travail fondamental je compte revenir à l'étude plus spécifique de la recommandation de tags et aussi à d'autres applications liées au web et à la recherche documentaire. En particulier il me parait intéressant d'utiliser le DPM-index directement, ou au besoin de l'adapter pour ces applications dans lesquelles on ne tient pas compte actuellement du chevauchement entre les n-grammes.

Concernant la classification des textes il serait intéressant d'étudier le comportement des métriques de pondération des termes que j'ai proposé avec d'autres méthodes d'apprentissage. J'ai commencé à le faire avec une approche centroïde sur des résumés d'articles médicaux [37]. Par rapport à l'apprentissage par SVM, les méthodes utilisant les centroïdes permettent d'améliorer les performances en temps de calcul et d'envisager le traitement de grands volumes de données. Le classement automatique des résumés d'articles biomédicaux de la banque MEDLINE/PubMed avec des catégories MeSH est actuellement un challenge particulièrement intéressant [108]. Il y a plusieurs manières de calculer un centroïde (cf. Chapitre 3), j'ai exploré d'autres alternatives à la moyenne pour le calculer, ce qui a permis d'obtenir des performances qualitatives proches de celles obtenues par les SVM sur le corpus Ohsumed [38].

Par ailleurs, les métriques de pondération des termes pour la classification de textes dans des catégories prédéfinies peuvent être adaptées à l'analyse automatique des sentiments. Dans ce dernier problème on considère comme catégories la polarité du sentiment exprimé (positif ou négatif par exemple) dans les réseaux sociaux [17, 23].

Une autre direction de recherche serait d'explorer la possibilité de combiner les métriques que j'ai développé pour l'extraction avec celles de la classification. L'idée serait par exemple d'améliorer la classification de textes en combinant les pondérations des termes en vue d'extraire des termes-clés avec les pondérations supervisées de catégorisation en utilisant les méthodes d'apprentissage.

Bibliographie

[1] Charu C. Aggarwal and ChengXiang Zhai. A survey of text classification algorithms. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 163–222. Springer, 2012.

[2] Hakan Altinçay and Zafer Erenel. Analytical evaluation of term weighting schemes for text categorization. *Pattern Recognition Letters*, 31(11) :1310–1323, 2010.

[3] Hakan Altinçay and Zafer Erenel. Using the absolute difference of term occurrence probabilities in binary text categorization. *Applied Intelligence*, 36(1) :148–160, 2012.

[4] Dima Badawi and Hakan Altinçay. A novel framework for termset selection and weighting in binary text classification. *Eng. Appl. of AI*, 35 :38–53, 2014.

[5] Iyad Batal and Milos Hauskrecht. Boosting knn text classification accuracy by using supervised term weighting schemes. In Cheung et al. [13], pages 2041–2044.

[6] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms : Bagging, boosting, and variants. *Machine learning*, 36(1-2) :105–139, 1999.

[7] Seong-Yong Bong and Kyu-Baek Hwang. Keyphrase extraction in biomedical publications using mesh and intraphrase word co-occurrence information. In *Proceedings of the ACM 15th international workshop on Data and text mining in biomedical informatics, New York, NY, USA, October 24-28, 2011*, DTMBio 2011, pages 63–66. ACM, 2011.

[8] Flavien Bouillot, Pascal Poncelet, and Mathieu Roche. Classification of small datasets : Why using class-based weighting measures ? In Troels Andreasen, Henning Christiansen, Juan Carlos Cubero Talavera, and Zbigniew W. Ras, editors, *Foundations of Intelligent Systems - 21st International Symposium, ISMIS 2014, Roskilde, Denmark, June 25-27, 2014. Proceedings*, volume 8502 of *Lecture Notes in Computer Science*, pages 345–354. Springer, 2014.

[9] Leo Breiman. Bagging predictors. *Machine learning*, 24(2) :123–140, 1996.

[10] Leo Breiman. Random forests. *Machine learning*, 45(1) :5–32, 2001.

[11] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7) :107–117, 1998.

[12] Ding-Yi Chen, Xue Li, Jing Liu, and Xia Chen. Ranking-constrained keyword sequence extraction from web documents. In *Proceedings of the 20th Australasian Database Conference, ADC 2009, Wellington, New Zealand, 20-23 January, 2009*, volume 92 of *CRPIT 2009*, pages 161–169. ACS, 2009.

[13] David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy J. Lin, editors. ACM, 2009.

[14] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3) :273–297, 1995.

[15] Franca Debole and Fabrizio Sebastiani. Supervised term weighting for automated text categorization. Technical Report Technical Report 2002-TR-08, Istituto di Scienza e Tecnologie dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, IT, 2002.

[16] Franca Debole and Fabrizio Sebastiani. Supervised term weighting for automated text categorization. In *Proceedings of the 2003 ACM Symposium on Applied Computing (SAC), March 9-12, 2003, Melbourne, FL, USA*, pages 784–788. ACM, 2003.

[17] Zhi-Hong Deng, Kun-Hu Luo, and Hongliang Yu. A study of supervised term weighting scheme for sentiment analysis. *Expert Systems with Applications*, 41(7) :3506–3513, 2014.

[18] Zhi-Hong Deng, Shiwei Tang, Dongqing Yang, Ming Zhang, Liyu Li, and Kunqing Xie. A comparative study on feature weight in text categorization. In Jeffrey Xu Yu, Xuemin Lin, Hongjun Lu, and Yanchun Zhang, editors, *Advanced Web Technologies and Applications, 6th Asia-Pacific Web Conference, APWeb 2004, Hangzhou, China, April 14-17, 2004, Proceedings*, volume 3007 of *Lecture Notes in Computer Science*, pages 588–597. Springer, 2004.

[19] Kathrin Eichler and Günter Neumann. DFKI KeyWE : Ranking keyphrases extracted from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, Sweden, July 15-16, 2010*, SemEval 2010, pages 150–153. ACL, 2010.

[20] Samhaa R. El-Beltagy and Ahmed Rafea. KP-Miner : Participation in SemEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, Sweden, July 15-16, 2010*, SemEval 2010, pages 190–193. ACL, 2010.

[21] Gonenc Ercan and Ilyas Cicekli. Using lexical chains for keyword extraction. *Information Processing & Management*, 43(6) :1705–1714, 2007.

[22] Hugo Jair Escalante, Mauricio A. García-Limón, Alicia Morales-Reyes, Mario Graff, Manuel Montes-y-Gómez, Eduardo F. Morales, and José Martínez-Carranza. Term-weighting learning via genetic programming for text classification. *Knowl.-Based Syst.*, 83 :176–189, 2015.

[23] Mohamed Abdel Fattah. New term weighting schemes with combination of multiple classifiers for sentiment analysis. *Neurocomputing*, 167 :434–442, 2015.

[24] Usama M. Fayyad and Keki B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence. Chambéry, France, August 28 - September 3, 1993*, IJCAI 1993, pages 1022–1029. Morgan Kaufmann, 1993.

[25] George Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3 :1289–1305, 2003.

[26] George Forman. BNS feature scaling : an improved representation over tf-idf for svm text classification. In James G. Shanahan, Sihem Amer-Yahia, Ioana Manolescu, Yi Zhang, David A. Evans, Aleksander Kolcz, Key-Sun Choi, and Abdur Chowdhury, editors, *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*, pages 263–270. ACM, 2008.

[27] Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. Domain-specific keyphrase extraction. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, July 31 - August 6, 1999*, IJCAI 1999, pages 668–673. Morgan Kaufmann, 1999.

[28] Katerina Frantzi, Sophia Ananiadou, and Hideki Mima. Automatic recognition of multi-word terms : the C-value/NC-value method. *International Journal on Digital Libraries*, 3(2) :115–130, 2000.

[29] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1) :119–139, 1997.

[30] Luit Gazendam, Christian Wartena, and Rogier Brussee. Thesaurus based term ranking for keyword extraction. In *Database and Expert Systems Applications, International Workshops, Bilbao, Spain, August 30 - September 3, 2010*, DEXA 2010, pages 49–53. IEEE, 2010.

[31] Liqiang Geng and Howard J. Hamilton. Interestingness measures for data mining : A survey. *ACM Computing Surveys*, 38(3), 2006.

[32] Salton Gerard. The smart retrieval system—experiments in automatic document processing. 1971.

[33] Hu Guan, Jingyu Zhou, and Minyi Guo. A class-feature-centroid classifier for text categorization. In Juan Quemada, Gonzalo León, Yoëlle S. Maarek, and Wolfgang Nejdl, editors, *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, pages 201–210. ACM, 2009.

[34] Mounia Haddoud and Saïd Abdeddaïm. Accurate keyphrase extraction by discriminating overlapping phrases. *J. Information Science*, 40(4) :488–500, 2014.

[35] Mounia Haddoud, Aïcha Mokhtari, Thierry Lecroq, and Saïd Abdeddaïm. Accurate keyphrase extraction from scientific papers by mining linguistic information. In *Proc. of the Workshop Mining Scientific Papers : Computational Linguistics and Bibliometrics, 15th International Society of Scientometrics and Informetrics Conference (ISSI), Istanbul, Turkey : http ://ceur-ws. org*, 2015.

[36] Mounia Haddoud, Aïcha Mokhtari, Thierry Lecroq, and Saïd Abdeddaïm. Combining supervised term weighting metrics for SVM text classification with extended term representation. *Knowledge and Information Systems*, 2015. accepted on Dec 28, 2015.

[37] Mounia Haddoud, Aïcha Mokhtari, Thierry Lecroq, and Saïd Abdeddaïm. Supervised term weights for biomedical text classification. In *Proceedings of the 12th International meeting on Computational Intelligence methods for Bioinformatics and Biostatistics, CIBB 2015, Naples, Italy, September 10-12, 2015*, pages 55–60, 2015.

[38] Mounia Haddoud, Aïcha Mokhtari, Thierry Lecroq, and Saïd Abdeddaïm. Supervised term weights for biomedical text classification : Improvements in nearest centroid computation. volume 9874 of *Lecture Notes in Computer Science*, page CIBB. Springer, 2016.

[39] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software : an update. *SIGKDD Explorations*, 11(1) :10–18, 2009.

[40] Eui-Hong Han and George Karypis. Centroid-based document classification : Analysis and experimental results. In Djamel A. Zighed, Henryk Jan Komorowski, and Jan M. Zytkow, editors, *Principles of Data Mining and Knowledge Discovery, 4th European Conference, PKDD 2000, Lyon, France, September 13-16, 2000, Proceedings*, volume 1910 of *Lecture Notes in Computer Science*, pages 424–431. Springer, 2000.

[41] Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (10) :993–1001, 1990.

[42] Kazi Saidul Hasan and Vincent Ng. Conundrums in unsupervised keyphrase extraction : Making sense of the state-of-the-art. In *Proceedings of the 23rd international conference on Computational Linguistics, Beijing, China, 23-27 August 2010*, COLING 2010 : Posters, pages 365–373, 2010.

[43] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction : A survey of the state of the art. In *ACL (1)*, pages 1262–1273, 2014.

[44] Katja Hofmann, Manos Tsagkias, Edgar Meij, and Maarten de Rijke. The impact of document structure on keyphrase extraction. In Cheung et al. [13], pages 1725–1728.

[45] Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, Sapporo, Japan July 11-12, 2003*, EMNLP 2003, pages 216–223. ACL, 2003.

[46] Anette Hulth, Jussi Karlgren, Anna Jonsson, Henrik Boström, and Lars Asker. Automatic keyword extraction using domain knowledge. In *Proceedings of the 2nd International Conference on Computational Linguistics and Intelligent Text Processing, CICLing 2001, Mexico-City, Mexico, February 18-24, 2001*, volume 2004 of *Lecture Notes in Computer Science*, pages 472–482. Springer, 2001.

[47] David J Ittner, David D Lewis, and David D Ahn. Text categorization of low quality images. In *Symposium on Document Analysis and Information Retrieval*, pages 301–315. Citeseer, 1995.

[48] Paul Jaccard. *Distribution de la Flore Alpine : dans le Bassin des dranses et dans quelques régions voisines.* Rouge, 1901.

[49] Xin Jiang, Yunhua Hu, and Hang Li. A ranking approach to keyphrase extraction. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Boston, MA, USA, July 19-23, 2009*, SIGIR 2009, pages 756–757. ACM, 2009.

[50] Taeho Jo. Neural based approach to keyword extraction from documents. In *Proceedings of the 2003 International Conference on Computational Science and Its Applications, Part I, ICCSA 2003, Montreal, Canada, May 18-21, 2003*, volume 2667 of *Lecture Notes in Computer Science*, pages 456–461. Springer, 2003.

[51] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA, 1999.

[52] Thorsten Joachims. Training linear SVMs in linear time. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 217–226, 2006.

[53] Amruta Joshi and Rajeev Motwani. Keyword generation for search engine advertising. In *Workshops Proceedings of the 6th IEEE International Conference on Data Mining, Hong Kong, China, December 18-22, 2006*, ICDM 2006 Workshops, pages 490–496. IEEE Computer Society, 2006.

[54] Seung-Shik Kang. Keyword-based document clustering. In *Proceedings of the 6th International Workshop on Information Retrieval with Asian Languages, Sappro, Japan, July 7, 2003*, pages 132–137. ACL, 2003.

[55] Su Nam Kim and Timothy Baldwin. Extracting keywords from multiparty live chats. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation, Bali, November 8-10, 2012*, PACLIC 2012, pages 199–208. Faculty of Computer Science, Universitas Indonesia, 2012.

[56] Su Nam Kim, Timothy Baldwin, and Min-Yen Kan. An unsupervised approach to domain-specific term extraction. In *Proceedings of the Australasian Language Technology Association Workshop, Sydney, Australia, 3-4 December 2009*, ALTA 2009, pages 94–98. ACL, 2009.

[57] Su Nam Kim, Timothy Baldwin, and Min-Yen Kan. Evaluating N-gram based evaluation metrics for automatic keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational*

Linguistics, Beijing, China, August 23-27, 2010, COLING 2010, pages 572–580, 2010.

[58] Su Nam Kim and Min-Yen Kan. Re-examining automatic keyphrase extraction approaches in scientific articles. In *Proceedings of the ACL 2009 Workshop on Multiword Expressions, Singapore, 6 August 2009*, MWE 2009, pages 9–16. ACL, 2009.

[59] Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. SemEval-2010 Task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, Sweden, July 15-16, 2010*, SemEval 2010, pages 21–26. ACL, 2010.

[60] Youngjoong Ko. A new term-weighting scheme for text classification using the odds of positive and negative class probabilities. *Journal of the Association for Information Science and Technology*, 2015.

[61] Mikalai Krapivin, Aliaksandr Autayeu, Maurizio Marchese, Enrico Blanzieri, and Nicola Segata. Keyphrases extraction from scientific documents : improving machine learning approaches with natural language processing. In *Proceedings of the 12th international conference on Asia-Pacific digital libraries, Gold Coast, Australia, June 21-25, 2010*, ICADL 2010, pages 102–111. Springer, 2010.

[62] Mikalai Krapivin, Maurizio Marchese, Andrei Yadrantsau, and Yanchun Liang. Unsupervised key-phrases extraction from scientific papers using domain and linguistic knowledge. In *Proceedings of the 3rd International Conference on Digital Information Management, London, UK, November 13-16, 2008*, ICDIM 2008, pages 105–112. IEEE, 2008.

[63] Anders Krogh, Jesper Vedelsby, et al. Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, 7 :231–238, 1995.

[64] Man Lan, Chew Lim Tan, Jian Su, and Yue Lu. Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4) :721–735, 2009.

[65] Zhenhui Li, Ding Zhou, Yun-Fang Juan, and Jiawei Han. Keyword extraction for social snippets. In *Proceedings of the 19th International Conference on World Wide Web, Raleigh, North Carolina, USA, April 26-30, 2010*, WWW 2010, pages 1143–1144. ACM, 2010.

[66] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations : Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1) :76–80, 2003.

[67] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of the 2009 Conference of the North American Chapter of the Association of Computational Linguistics, Boulder, Colorado, USA, May 31 - June 5, 2009*, NAACL 2009, pages 620–628. ACL, 2009.

[68] Jianyi Liu, Cong Wang, Zhengyang Liu, and Wenbin Yao. Advertising keywords extraction from web pages. In *Proceedings of the 7th International Conference on Web Information Systems and Mining, WISM 2010, Sanya, China, October 23-24, 2010*, volume 6318 of *Lecture Notes in Computer Science*, pages 336–343. Springer, 2010.

[69] Ying Liu, Han Tong Loh, and Aixin Sun. Imbalanced text classification : A term weighting approach. *Expert Systems with Applications*, 36(1) :690–701, 2009.

[70] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore, August 6-7, 2009*, EMNLP 2009, pages 257–266. ACL, 2009.

[71] Patrice Lopez and Laurent Romary. HUMB : Automatic key term extraction from scientific articles in grobid. In *Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, Sweden, July 15-16, 2010*, SemEval 2010, pages 248–251. ACL, 2010.

[72] Julie Beth Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11 :22–31, 1968.

[73] Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Saso Dzeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9) :3084–3104, 2012.

[74] Justin Martineau, Tim Finin, Anupam Joshi, and Shamit Patel. Improving binary classification on text problems using differential word features. In David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy J. Lin, editors, *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, pages 2019–2024. ACM, 2009.

[75] Yutaka Matsuo and Mitsuru Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(1) :157–169, 2004.

[76] Olena Medelyan, Eibe Frank, and Ian H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the*

2009 Conference on Empirical Methods in Natural Language Processing, Singapore, August 6-7, 2009, EMNLP 2009, pages 1318–1327. ACL, 2009.

[77] Olena Medelyan and Ian H. Witten. Measuring inter-indexer consistency using a thesaurus. In *Proceedings of the 6th ACM/IEEE Joint Conference on Digital Libraries, Chapel Hill, NC, USA, June 11-15, 2006*, JCDL 2006, pages 274–275. ACM, 2006.

[78] Rada Mihalcea and Paul Tarau. TextRank : Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, July 25-26, 2004*, EMNLP 2004, pages 404–411. ACL, 2004.

[79] Junichiro Mori, Yutaka Matsuo, and Mitsuru Ishizuka. Finding user semantics on the web using word co-occurrence information. In *Proceedings of the International Workshop on Personalization on the Semantic Web, Edinburgh, UK, July 24 - 30, 2005*, PerSWeb 2005 : Posters, pages 77–86, 2005.

[80] David Newman, Nagendra Koilada, Jey Han Lau, and Timothy Baldwin. Bayesian text segmentation for index term identification and keyphrase extraction. In *Proceedings of the 24th International Conference on Computational Linguistics, Mumbai, India, December 8-15, 2012*, COLING 2012, pages 2077–2092, 2012.

[81] Tam T. Nguyen, Kuiyu Chang, and Siu Cheung Hui. Supervised term weighting centroid-based classifiers for text categorization. *Knowledge and Information Systems*, 35(1) :61–85, 2013.

[82] Thuy Dung Nguyen and Min-Yen Kan. Keyphrase extraction in scientific publications. In *Proceedings of the 10th International Conference on Asian Digital Libraries, ICADL 2007, Hanoi, Vietnam, December 10-13, 2007*, volume 4822 of *Lecture Notes in Computer Science*, pages 317–326. Springer, 2007.

[83] Thuy Dung Nguyen and Minh-Thang Luong. WINGNUS : Keyphrase extraction utilizing document logical structure. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval 2010, pages 166–169, Uppsala, Sweden, July 2010. ACL.

[84] David Opitz and Richard Maclin. Popular ensemble methods : An empirical study. *Journal of Artificial Intelligence Research*, pages 169–198, 1999.

[85] T. Pal, H. Banka, P. Mitra, and B. Das. Linguistic knowledge based supervised key-phrase extraction. In *Proceedings of National Conference on Future Trends in Information & Communication Technology*

& Applications, Bhubaneswar, India, September 10-11, 2011, NCICT 2011, 2011.

[86] Girish Keshav Palshikar. Keyword extraction from a single document using centrality measures. In *Proceedings of the 2nd International Conference on Pattern Recognition and Machine Intelligence, PReMI 2007, Kolkata, India, December 18-22, 2007*, volume 4815 of *Lecture Notes in Computer Science*, pages 503–510. Springer, 2007.

[87] Patrick Paroubek, Pierre Zweigenbaum, Dominic Forest, and Cyril Grouin. Indexation libre et contrôlée d'articles scientifiques présentation et résultats du défi fouille de textes deft2012. *Actes du huitième DÉfi Fouille de Textes*, 2012.

[88] Pavel Pecina. Lexical association measures and collocation extraction. *Language Resources and Evaluation*, 44(1-2) :137–158, 2010.

[89] M. F. Porter. Readings in information retrieval. chapter An algorithm for suffix stripping, pages 313–316. Morgan Kaufmann, 1997.

[90] Nakov Preslav and Hearst Marti A. Solving relational similarity problems using the web as a corpus. In *ACL*, pages 452–460, 2008.

[91] Nirmala Pudota, Antonina Dattolo, Andrea Baruzzo, Felice Ferrara, and Carlo Tasso. Automatic keyphrase extraction and ontology mining for content-based tag recommendation. *International Journal of Intelligent Systems*, 25(12) :1158–1186, 2010.

[92] Minghui Qiu, Yaliang Li, and Jing Jiang. Query-oriented keyphrase extraction. In *Information Retrieval Technology - Proceedings of the 18th Asia Information Retrieval Societies Conference, AIRS 2012, Tianjin, China, December 17-19, 2012*, volume 7675 of *Lecture Notes in Computer Science*, pages 64–75. Springer, 2012.

[93] Abdur Rehman, Kashif Javed, Haroon A. Babri, and Mehreen Saeed. Relative discrimination criterion - A novel feature ranking method for text data. *Expert Syst. Appl.*, 42(7) :3670–3681, 2015.

[94] Fuji Ren and Mohammad Golam Sohrab. Class-indexing-based term weighting for automatic text classification. *Information Sciences*, 236 :109–125, 2013.

[95] Joseph John Rocchio. Review of classifier combination methods. In *The SMART Retrieval System - Experimens in Automatic Document Processing*, pages 313–323. Prentice Hall Inc., 1971.

[96] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents. In Michael W. Berry and Jacob Kogan, editors, *Text Mining : Applications and Theory*, pages 1–20. Wiley, 2010.

[97] Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11) :613–620, 1975.

[98] Kamal Sarkar, Mita Nasipuri, and Suranjan Ghose. A new approach to keyphrase extraction using neural networks. *CoRR*, abs/1004.3274, 2010.

[99] Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2) :197–227, 1990.

[100] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1) :1–47, 2002.

[101] Ilia Smirnov. Overview of stemming algorithms. http ://the-smirnovs.org/info/stemming.pdf. Unpublished, 2008.

[102] Min Song, Il-Yeol Song, Robert B. Allen, and Zoran Obradovic. Keyphrase extraction-based query expansion in digital libraries. In *Proceedings of the 6th ACM/IEEE Joint Conference on Digital Libraries, Chapel Hill, NC, USA, June 11-15, 2006*, JCDL 2006, pages 202–209. ACM, 2006.

[103] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3 :583–617, 2003.

[104] Stamatina Thomaidou and Michalis Vazirgiannis. Multiword keyword recommendation system for online advertising. In *Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining, Kaohsiung, Taiwan, 25-27 July, 2011*, ASONAM 2011, pages 423–427. IEEE, 2011.

[105] Kai Ming Ting and Ian H Witten. Issues in stacked generalization. *J. Artif. Intell. Res.(JAIR)*, 10 :271–289, 1999.

[106] Takashi Tomokiyo and Matthew Hurst. A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions, Sapporo, Japan, July 7-12, 2003*, MWE 2003, pages 33–40. ACL, 2003.

[107] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1, Edmonton, Canada, May 27-Jun 1, 2003*, HLT-NAACL 2003, pages 173–180, 2003.

[108] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R. Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artières, Axel Ngonga, Norman Heino, Éric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16 :138, 2015.

[109] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis P. Vlahavas. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook, 2nd ed.*, pages 667–685. Springer, 2010.

[110] Sergey Tulyakov, Stefan Jaeger, Venu Govindaraju, and David S. Doermann. Review of classifier combination methods. In Simone Marinai and Hiromichi Fujisawa, editors, *Machine Learning in Document Analysis and Recognition*, volume 90 of *Studies in Computational Intelligence*, pages 361–386. Springer, 2008.

[111] Peter D. Turney. Extraction of keyphrases from text : Evaluation of four algorithms. Technical Report ERB-1051, National Research Council. Institute for Information Technology, 1997.

[112] Peter D. Turney. Learning to extract keyphrases from text. Technical Report ERB-1057, National Research Council. Institute for Information Technology, 1999.

[113] Peter D. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4) :303–336, 2000.

[114] Peter D. Turney. Coherent keyphrase extraction via web mining. In *Proceedings of the 18h International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, IJCAI 2003, pages 434–442. Morgan Kaufmann, 2003.

[115] Peter D. Turney. Similarity of semantic relations. *Computational Linguistics*, 32(3) :379–416, 2006.

[116] Peter D. Turney and Patrick Pantel. From frequency to meaning : Vector space models of semantics. *J. Artif. Intell. Res. (JAIR)*, 37 :141–188, 2010.

[117] Xiaojun Wan and Jianguo Xiao. CollabRank : Towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics,*

Manchester, UK, August 18-22, 2008, COLING 2008, pages 969–976, 2008.

[118] Xiaojun Wan and Jianguo Xiao. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence, Chicago, Illinois, USA, July 13-17, 2008*, AAAI 2008, pages 855–860. AAAI Press, 2008.

[119] Xiaojun Wan and Jianguo Xiao. Exploiting neighborhood knowledge for single document summarization and keyphrase extraction. *ACM Transactions on Information and System Security*, 28(2) :1–34, 2010.

[120] Haizhou Wang and Mingzhou Song. Ckmeans.1d.dp : Optimal k-means clustering in one dimension by dynamic programming. *The R Journal*, 3 :29–33, 2011.

[121] Jiabing Wang and Hong Peng. Keyphrases extraction from web document by the least squares support vector machine. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, Compiegne, France, September 19-22, 2005*, WI 2005, pages 293–296. IEEE, 2005.

[122] Jiabing Wang, Hong Peng, Jing-Song Hu, and Jun Zhang. Ensemble learning for keyphrases extraction from scientific document. In *Advances in Neural Networks - ISNN 2006, Proceedings of the 3rd International Symposium on Neural Networks, Part I, Chengdu, China, May 28 - June 1, 2006*, volume 3971 of *Lecture Notes in Computer Science*, pages 1267–1272. Springer, 2006.

[123] Jinghua Wang, Jianyi Liu, and Cong Wang. Keyword extraction based on PageRank. In *Proceedings of the 11th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, PAKDD 2007, Nanjing, China, May 22-25, 2007*, volume 4426 of *Lecture Notes in Computer Science*, pages 857–864. Springer, 2007.

[124] Yang Wei. An iterative approach to keywords extraction. In *Proceedings of the 3rd International Conference, on Advances in Swarm Intelligence, Part II, ICSI 2012, Shenzhen, China, June 17-20, 2012*, volume 7332 of *Lecture Notes in Computer Science*, pages 93–99. Springer, 2012.

[125] Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. KEA : practical automatic keyphrase extraction. In *Proceedings of the 4th ACM conference on Digital libraries, Berkeley, California, USA, August 11-14, 1999*, DL 1999, pages 254–255. ACM, 1999.

[126] David H Wolpert. Stacked generalization. *Neural networks*, 5(2) :241–259, 1992.

[127] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In Douglas H. Fisher, editor, *Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee, USA, July 8-12, 1997*, pages 412–420. Morgan Kaufmann, 1997.

[128] Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho. Finding advertising keywords on web pages. In *Proceedings of the 15th international conference on World Wide Web, Edinburgh, Scotland, UK, May 23-26, 2006*, WWW 2006, pages 213–222. ACM, 2006.

[129] Wei You, Dominique Fontaine, and Jean-Paul A. Barthès. An automatic keyphrase extraction system for scientific documents. *Knowledge and Information Systems*, 34(3) :691–724, 2013.

[130] Hongyuan Zha. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Tampere, Finland, August 11-15, 2002*, SIGIR 2002, pages 113–120. ACM, 2002.

[131] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Trans. Knowl. Data Eng.*, 26(8) :1819–1837, 2014.

[132] Weinan Zhang, Dingquan Wang, Gui-Rong Xue, and Hongyuan Zha. Advertising keywords recommendation for short-text web pages using Wikipedia. *ACM Transactions on Intelligent Systems and Technology*, 3(2) :36, 2012.

[133] Yongzheng Zhang, Evangelos E. Milios, and A. Nur Zincir-Heywood. A comparative study on key phrase extraction methods in automatic web site summarization. *Journal of Digital Information Management*, 5(5) :323–332, 2007.

[134] Wayne Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li. Topical keyphrase extraction from Twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies, 19-24 June, 2011, Portland, Oregon, USA*, ACL 2010, pages 379–388. ACL, 2011.

[135] Zhi-Hua Zhou, Jianxin Wu, and Wei Tang. Ensembling neural networks : many could be better than all. *Artificial intelligence*, 137(1) :239–263, 2002.