Apprentissage à base de Noyaux Sémantiques pour le Traitement de Données Textuelles

Sujeevan Aseervatham

N° d'ordre :_____
N° attribué par la bibliothèque :_____

Université Paris 13 – Institut Galilée
Laboratoire d'Informatique de Paris Nord
UMR 7030 du CNRS

THÈSE

présentée pour l'obtention du titre de
Docteur en Sciences

spécialité
Informatique

par

Sujeevan ASEERVATHAM

Apprentissage à base de Noyaux Sémantiques pour le Traitement de Données Textuelles

---

dirigée par
Pr. Younès Bennani

Soutenue le 12 décembre 2007 devant le jury composé de :

M.	Massih-Reza Amini	Maître de Conférences, LIP6 – Université de Paris 6
M.	Younès Bennani	Professeur, LIPN – Université de Paris 13
M.	Christophe Fouqueré	Professeur, LIPN – Université de Paris 13
M.	Cyril Goutte	Agent de Recherche, CNRC – Canada
M.	Jean-François Marcotorchino	Directeur Scientifique, Thalès Land & Joint
M.	Alain Rakotomamonjy	Professeur, INSA – Université de Rouen
Mme	Michèle Sebag	Directrice de Recherche, CNRS – LRI (Paris 11)
M.	Emmanuel Viennet	Maître de Conférences, LIPN, Université de Paris 13

Résumé

Apprentissage à base de Noyaux Sémantiques pour le Traitement de Données Textuelles.

Depuis le début des années 80, les méthodes statistiques et, plus spécifiquement, les méthodes d'apprentissage appliquées au traitement de données textuelles connaissent un intérêt grandissant. Cette tendance est principalement due au fait que la taille des corpus est en perpétuelle croissance. Ainsi, les méthodes utilisant le travail d'experts sont devenues des processus coûteux perdant peu à peu de leur popularité au profit des systèmes d'apprentissage.

Dans le cadre de cette thèse, nous nous intéressons principalement à deux axes. Le premier axe porte sur l'étude des problématiques liées au traitement de données textuelles structurées par des approches à base de noyaux. Nous présentons, dans ce contexte, un noyau sémantique pour les documents structurés en sections notamment sous le format XML. Le noyau tire ses informations sémantiques à partir d'une source de connaissances externe, à savoir un thésaurus. Notre noyau a été testé sur un corpus de documents médicaux avec le thésaurus médical UMLS. Il a été classé, lors d'un challenge international de catégorisation de documents médicaux, parmi les 10 méthodes les plus performantes sur 44. Le second axe porte sur l'étude des concepts latents extraits par des méthodes statistiques telles que l'analyse sémantique latente (LSA). Nous présentons, dans une première partie, des noyaux exploitant des concepts linguistiques provenant d'une source externe et des concepts statistiques issus de la LSA. Nous montrons qu'un noyau intégrant les deux types de concepts permet d'améliorer les performances. Puis, dans un deuxième temps, nous présentons un noyau utilisant des LSA locaux afin d'extraire des concepts latents permettant d'obtenir une représentation plus fine des documents.

Abstract

Semantic Kernel-based Machine Learning for Textual Data Processing.

Since the early eighties, statistical methods and, more specifically, the machine learning for textual data processing have known a considerable growth of interest. This is mainly due to the fact that the number of documents to process is growing exponentially. Thus, expert-based methods have become too costly, losing the research focus to the profit of machine learning-based methods.

In this thesis, we focus on two main issues. The first one is the processing of semi-structured textual data with kernel-based methods. We present, in this context, a semantic kernel for documents structured by sections under the XML format. This kernel captures the semantic information with the use of an external source of knowledge e.g., a thesaurus. Our kernel was evaluated on a medical document corpus with the UMLS thesaurus. It was ranked in the top ten of the best methods, according to the F1-score, among 44 algorithms at the 2007 CMC Medical NLP International Challenge. The second issue is the study of the use of latent concepts extracted by statistical methods such as the Latent Semantic Analysis (LSA). We present, in a first part, kernels based on linguistic concepts from external sources and on latent concepts of the LSA. We show that a kernel integrating both kinds of concepts improves the text categorization performances. Then, in a second part, we present a kernel that uses local LSAs to extract latent concepts. Local latent concepts are used to have a more finer representation of the documents.

Table des matières

Table des figures

Liste des tableaux

Liste des Algorithmes

CHAPITRE 1

Introduction

1.1 Contexte

Les moyens de stockage d'information connaissent, depuis quelques années, une forte croissance. De nos jours, il n'est plus rare de voir des disques durs atteignant le téraoctet. Cette capacité a permis de stocker une quantité importante de données dans le but d'acquérir une meilleure connaissance de l'information pour des tâches telles que la prise de décision. Toutefois, face à un tel volume de données, la recherche de l'information pertinente est devenue complexe et coûteuse. Pour répondre à cette problématique, la fouille de données automatique est devenu un domaine de recherche très actif. Plus particulièrement, l'apprentissage artificiel s'est imposé comme un axe majeur de ce domaine. La spécificité des algorithmes d'apprentissage numérique réside dans le fait qu'ils sont capables d'extraire automatiquement, à partir d'outils statistiques, des relations entre les données et de les utiliser sur de nouvelles données pour en déduire des informations. Dès lors, de nombreux algorithmes de recherche et d'extraction d'information ont vu le jour. La famille des algorithmes connexionnistes est un exemple de méthodes suscitant un intérêt considérable dans cette problématique.

Si les données numériques ont fait l'objet d'une attention particulière dans la fouille de données en donnant naissance à des algorithmes d'apprentissage numérique performants, il n'en a pas été de même pour les données textuelles. En effet, il a fallu attendre les années 80 pour que les méthodes statistiques et, plus spécifiquement, les méthodes d'apprentissage appliquées au traitement de données textuelles connaissent un intérêt grandissant. Cette tendance a été principalement due à la taille des corpus en perpétuelle croissance. Ainsi, les méthodes basées sur le travail d'experts pour établir des règles de traitements sont devenues coûteuses perdant peu à peu de leur popularité au profit des systèmes d'apprentissages.

Les méthodes d'apprentissage supervisé permettent d'extraire automatiquement, à partir d'un échantillon d'apprentissage ("annoté"), des relations entre les données et le problème posé. Ces relations peuvent ensuite être généralisées à l'ensemble d'un corpus. Par exemple, dans le cas de la catégorisation, le problème consiste à attribuer à chaque document une catégorie. Les relations entre le document et sa catégorie sont extraites par un processus inductif appliqué à un ensemble d'apprentissage constitué d'une série de documents avec l'indication de leur catégorie respective.
Parmi les algorithmes d'apprentissage, les méthodes à noyaux connaissent un énorme succès depuis quelques années. Les noyaux sont des fonctions de similarité respectant certaines propriétés mathématiques. Ils peuvent être utilisés avec des algorithmes d'apprentissage linéaires tels que les Séparateurs à Vaste Marge (SVM) pour extraire des relations non-linéaires.
Le succès des noyaux pour le traitement des données textuelles s'explique par deux points importants :

1. la possibilité de définir des noyaux sur des données non-numériques complexes telles que des graphes et des arbres,
2. l'utilisation des noyaux comme fonctions de similarité dans des algorithmes numériques.

1.2 Motivation

Dans le cadre de cette thèse, nous nous intéressons principalement à la définition de noyaux sémantiques. Ces noyaux ont la particularité de se baser sur le sens porté par les documents pour calculer les valeurs de similarité. Le sens d'un document sera défini par les concepts qu'il contient. Les similarités calculées à partir de ces concepts permettent de tenir compte de détails très fins au niveau des documents. L'information caractérisant chaque document est alors plus riche.
Nos travaux sur les noyaux sémantiques peuvent se classer en deux groupes :

1. les noyaux sémantiques pour données semi-structurées : nous nous intéressons dans cette partie aux noyaux pour documents "non-linéaires" où le texte peut être mis en forme en structure telle que des arbres. En effet, les documents sont de plus en plus organisés en champs d'information notamment sous le format XML. Ce formatage permet non seulement d'avoir une information riche (et hétérogène) mais permet aussi aux systèmes de traitement automatique de gérer plus facilement ces données (notamment pour l'affichage). Parmi les documents semi-structurées, nous pouvons citer, entres autres, les formulaires, les documents textes organisés en section et les courriels.

2. les noyaux sémantique basés sur l'information latente : dans cette partie, nous étudions l'utilisation de concepts extraits par des méthodes statistiques, telles que l'Analyse Sémantique Latente [DDL+90], pour la définition de noyaux. L'objectif est d'étudier la possibilité de remplacer les concepts linguistiques provenant d'ontologies par des concepts

statistiques extraits automatiquement. Ainsi, nous souhaitons mettre au point des algorithmes autonomes ne nécessitant pas l'intervention d'expert.

Nous évaluerons nos noyaux sur une tâche de catégorisation de texte. Pour cela, nous utiliserons les Séparateurs à Vaste Marge (SVM) comme algorithmes d'apprentissage pour l'intégration de nos noyaux.

En outre, une partie de nos travaux s'inscrit dans le cadre du projet Infomagic du pôle de compétitivité Cap Digital. Ce projet consiste à mettre en place, sur une période de trois ans, un laboratoire industriel de sélection, de test, d'intégration et de validation de technologies franciliennes dans le domaine de l'ingénierie des connaissances. Ce laboratoire s'appuie sur une plate-forme commune d'interopérabilité. Cette plate-forme doit couvrir les grands domaines fonctionnels et techniques de l'analyse d'information que sont la recherche et l'indexation, l'extraction de connaissances, et la fusion d'informations multimédias, sur tous les types de sources : texte, données, images et son.
Les partenaires de ce projet sont : BERTIN, EADS, EUROPLACE, INA, INTUILAB, ODILE JACOB, ONERA, PERTIMM , SPIKENET, TEMIS, THALES, VECSYS, XEROX , CEA, ENST, Université Paris 6, Université Paris 8, Université Paris 13, Université Paris Dauphine, Université Orsay, Université Marne la Vallée et le CNRS.

1.3 Organisation de la thèse et contributions

Ce manuscrit est organisé en cinq principaux chapitres :

- **Chapitre 2 :** Ce chapitre permet de présenter la problématique de l'apprentissage numérique. Ainsi, il expose les principes généraux de l'apprentissage numérique. Les différents types d'apprentissage sont décrits. Nous exposons, entre autres, le principe d'induction de la minimisation du risque structurel qui est à l'origine des Séparateurs à Vaste Marge (SVM). Puis, un tour d'horizon des algorithmes d'apprentissage à vaste marge est proposé. Les SVM et le principe des noyaux sont détaillés. En effet, les noyaux sont notre thème principal et les SVM nous serviront d'algorithme d'apprentissage pour l'intégration de nos noyaux.

- **Chapitre 3 :** Le chapitre 3 présente, dans un premier temps, le problème de l'apprentissage des données textuelles. Les outils permettant d'appliquer les algorithmes d'apprentissage numérique aux textes y sont détaillés. Ensuite, dans un deuxième temps, nous nous intéressons aux noyaux pour données textuelles semi-structurées. En effet, de plus en plus de documents utilisent une structure pour "mettre en forme" l'information. Les documents XML font, ainsi, partie de cette famille de documents semi-structurées. Nous présentons les principaux noyaux, proposés dans la littérature, permettant de tenir compte de la structure des textes.

– **Chapitre 4 :** Nous proposons, dans ce chapitre, un nouveau noyau sémantique pour documents textuels exploitant une structure en sections indépendantes. L'aspect sémantique du noyau est donné par l'utilisation de concepts, provenant d'ontologies, pour la modélisation des documents. Le noyau a été appliqué au domaine médical pour un problème de catégorisation de dossiers patients au format XML dans le cadre d'une compétition internationale de catégorisation de textes médicaux. Une ontologie médicale provenant du système UMLS (*Unified Medical Language System*) a été utilisée pour l'annotation sémantique exploitée par le noyau. Notre noyau, décrit dans ce chapitre, a été classé lors de cette compétition dans les dix méthodes, parmi 44, obtenant les meilleures performances.

– **Chapitre 5 :** Dans ce chapitre, nous étudions l'utilisation de méthodes statistiques pour l'extraction de concepts sémantiques. L'avantage de cette approche est que l'algorithme d'apprentissage n'est plus dépendant de sources d'information externes. L'algorithme est alors dit agnostique. Ce chapitre est composé de trois grandes sections. La première section présente les principales méthodes à noyaux permettant d'exploiter l'information latente issue des données. L'Analyse Sémantique Latente (LSA) étant une méthode de référence dans ce domaine, une part importante de cette section y est consacrée. Dans la deuxième section, nous proposons un espace vectoriel sémantique basé sur des concepts linguistiques. Une étude empirique est alors menée pour comparer la LSA avec cet espace sémantique. Les résultats montrent que les concepts linguistiques restent plus expressifs que les concepts statistiques obtenus par la LSA. Toutefois, l'utilisation de concepts sémantique extraits par la LSA dans l'espace de concepts linguistiques permet d'obtenir les meilleurs résultats. Dans la dernière section, nous proposons un noyau sémantique enrichi par des concepts locaux. Pour cela, nous définissons un espace sémantique global composé de sous-espaces sémantiques orthogonaux. Chaque sous-espace est obtenu en appliquant une LSA localement à certaines régions du corpus. Les résultats des expériences sur ce noyau montrent que les concepts locaux permettent d'améliorer les performances notamment dans les espaces de faible dimension.

– **Chapitre 6 :** Le chapitre 6 expose nos travaux de recherche dans le domaine de l'apprentissage symbolique. Ces travaux concernent l'extraction de motifs de séquences fréquentes pour la génération de règles d'association. Ainsi, nous présentons brièvement la problématique de l'extraction de motifs. Puis, nous proposons un nouvel algorithme d'extraction de séquences. Cet algorithme combine deux algorithmes dont l'un est connu pour sa performance au niveau de l'utilisation de mémoire et l'autre pour sa performance en matière de temps d'exécution. Nous montrons expérimentalement que notre nouvel algorithme réussit à intégrer les points forts des deux algorithmes.

Nous concluons cette thèse en exposant les points forts de nos contributions et les perspectives de recherche dans ce domaine.

CHAPITRE 2

Apprentissage numérique

2.1 Introduction

Dans le cadre de nos travaux, l'apprentissage numérique a été choisi comme approche pour le traitement de données textuelles. Par conséquent, nous présentons, dans une première partie, l'apprentissage numérique dans sa généralité. Puis, nous nous focalisons sur les Séparateurs à Vaste Marge (SVM). Nous exposons les principales variantes des SVM et les différentes tâches qu'ils peuvent résoudre.

2.2 Principes généraux

L'apprentissage automatique consiste à découvrir des relations pertinentes dans un ensemble de données. Ces relations fournissent des connaissances sur la source ayant généré les données. De ce fait, ces relations peuvent être utilisées pour prédire (classer) des données générées par la même source. Ce processus est inductif, il tente de généraliser les relations extraites sur un échantillon de données (ensemble d'apprentissage) à tout l'espace de données de la source.

L'apprentissage permet de résoudre deux problèmes principaux :

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

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

Dans cette section, nous présentons très brièvement les principales approches de l'apprentissage numérique. Pour illustrer ces approches, nous nous intéresserons uniquement aux problèmes de catégorisation et de classification.

2.2.1 Apprentissage Supervisé

Les méthodes supervisées traitent des données de la forme $(\mathbf{x}, y)$ où $\mathbf{x}$ représente un vecteur et y une valeur (continue ou discrete) associée à x. Ainsi, pour la tâche de classement, y est une variable discrète finie indiquant l'étiquette associée à $\mathbf{x}$. Dans le cas d'un classement binaire (problème à deux classes) y prend généralement une valeur dans $\{-1, +1\}$. Dans la tâche de prédiction, y est une variable réelle.
L'objectif de l'apprentissage est d'extraire une relation reliant y à $\mathbf{x}$ représentée par la fonction $f(\mathbf{x}, \alpha)$ avec $\alpha \in \Lambda$ un vecteur paramétrant f tel que $f \in \mathcal{F} = \{f(., \alpha) : \alpha \in \Lambda\}$. Par exemple, si nous fixons $\mathcal{F}$ comme étant l'ensemble des fonctions linéaires alors $f(\mathbf{x}, \alpha) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ avec $\alpha = (\mathbf{w}, b)$.

La fonction f est extraite à partir d'un ensemble d'apprentissage $\mathcal{D} = \{(\mathbf{x_i}, y_i\}_{i=1}^{N}$ en utilisant un principe dit d'induction.

2.2.1.1 Principe d'induction

La Minimisation du Risque Théorique

Les paramètres α définissant la fonction f sont choisis pour minimiser le risque moyen d'un mauvais classement. Ce risque appelé risque théorique, risque fonctionnel ou encore erreur de généralisation est donné par la formule suivante :

$$R(\alpha) = \int_{\mathcal{X} \times \mathcal{Y}} \mathcal{L}(y, f(\mathbf{x}, \alpha)) dF(\mathbf{x}, y) \tag{2.1}$$

avec $F(\mathbf{x}, y)$ la fonction de répartition des données caractérisant la source et $\mathcal{L}(y, \hat{y})$ étant la fonction de perte de la prédiction $\hat{y}$ sachant que la valeur correcte est y.
Dans le cas d'une prédiction où y est à valeur réelle, la fonction de perte généralement utilisée est l'écart quadratique :

$$\mathcal{L}(y, \hat{y}) = (y - \hat{y})^2 \tag{2.2}$$

qui permet de pénaliser la fonction f selon l'importance de l'écart entre la valeur réelle et la valeur prédite.
Dans le cas d'un classement, la fonction de perte sera :

$$\mathcal{L}(y, \hat{y}) = \begin{cases} 1 & \text{si } y \neq \hat{y} \\ 0 & \text{sinon} \end{cases} \tag{2.3}$$

ce qui équivaut à une pénalisation uniquement dans le cas d'un mauvais classement.

Le problème de l'apprentissage se ramène donc à un problème d'optimisation. En effet, les paramètres α^* optimum sont donnés par le problème suivant :

$$\alpha^* = \text{argmin}_\alpha R(\alpha) \tag{2.4}$$

Toutefois, la fonction de répartition $F(\mathbf{x}, y)$ caractérisant la source étant généralement inconnue, le risque théorique n'est, par conséquent, pas calculable. Il est donc nécessaire de passer par des estimateurs du risque théorique basés sur un échantillon d'apprentissage $\mathcal{D}$.

La Minimisation du Risque Empirique (ERM)

En utilisant un ensemble d'apprentissage $\mathcal{D} = \{(\mathbf{x_i}, y_i)\}_{i=1}^N$, le risque théorique peut être estimé par le risque empirique :

$$R_{emp}(\alpha) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i, f(\mathbf{x_i}, \alpha)) \tag{2.5}$$

Après l'apprentissage de f, en utilisant un ensemble $\mathcal{D}' = \{(\mathbf{x_i}', y_i')\}_{i=1}^M$ de données indépendantes générées aléatoirement avec $F(\mathbf{x}, y)$, nous obtenons les variables aléatoires $(y_i', f(\mathbf{x_i}', \alpha))$. En utilisant la loi des grands nombres, nous obtenons :

$$\forall \epsilon > 0, \lim_{M \to +\infty} P(|\frac{1}{M} \sum_{i=1}^{M} \mathcal{L}(y_i', f(\mathbf{x_i}', \alpha)) - \mathbb{E}(\mathcal{L}(y_i', f(\mathbf{x_i}', \alpha)))| \geq \epsilon) = 0 \tag{2.6}$$

Ainsi le risque empirique tend en probabilité vers le risque théorique lorsque le nombre de données tend vers l'infini :

$$R_{emp}(\alpha) \xrightarrow[p]{M \to \infty} R(\alpha) \tag{2.7}$$

Il est à noter que le risque empirique n'est qu'une estimation du risque théorique. Les algorithmes employant ce principe sont souvent sujets au sur-apprentissage. Le sur-apprentissage a lieu lorsque la fonction f apprise ne commet quasiment aucune erreur sur les données d'apprentissage (risque empirique très faible) mais commet beaucoup d'erreur sur les autres données (risque théorique élevé). La fonction a alors "appris par coeur" les données d'apprentissage. Ce phénomène donne lieu à des fonctions oscillantes comme illustré dans la figure 2.1.
Pour éviter le sur-apprentissage, nous pouvons utiliser un terme de régularisation $\lambda\Omega(f)$, avec

$\lambda \in \mathbb{R}$ un coefficient de pénalisation et $\Omega(f)$ une fonction prenant en compte les oscillations de f (par exemple $\Omega(f) = ||f||^2$). L'expression à minimiser devient donc [Bis95] :

$$\frac{1}{M}\sum_{i=1}^{M}\mathcal{L}(y_i, f(\mathbf{x_i}, \alpha)) + \lambda\Omega(f) \tag{2.8}$$

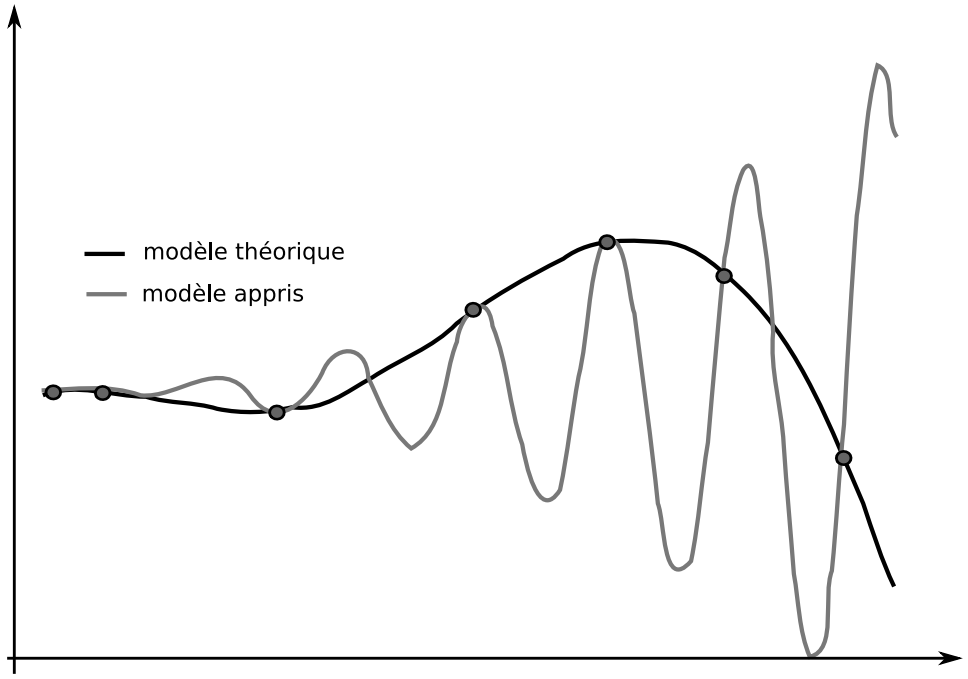


Fig. 2.1 – Phénomène de sur-apprentissage.

La Minimisation du Risque Structurel (SRM)

Vapnik et Chervonenkis [Vap95] ont montré que le risque empirique seul n'était pas un bon estimateur du risque théorique. En effet, le risque théorique peut être approché avec une meilleure borne. Dans le cas du classement, nous avons avec une probabilité $1 - \eta$, la borne suivante :

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\log(2M/h) + 1) - \log(\eta/4)}{M}} \tag{2.9}$$

avec h la VC-dimension de f. **La VC-Dimension** [Vap95] permet de quantifier la capacité de séparation d'une famille $\mathcal{F}$ de fonctions.

Définition 2.2.1 (Fonction de croissance)**.** *Soit une famille $\mathcal{F}$ de fonctions classifieurs $f : \mathcal{X} \rightarrow \{-1, 1\}$, la fonction de croissance $G(\mathcal{F}, m)$ est le nombre maximum d'étiquetages possibles d'une séquence de m points par les fonctions de $\mathcal{F}$:*

$$G(\mathcal{F}, M) = \max_{M} |\{(\mathbf{x_1}, f(\mathbf{x_1})), \ldots, (\mathbf{x_M}, f(\mathbf{x_M})) : f \in \mathcal{F}\}| \tag{2.10}$$

Theorème 2.2.2 (VC-Dimension [Vap95]). *Étant donné une famille $\mathcal{F}$ de fonctions classifieurs $f : \mathcal{X} \to \{-1, 1\}$, la VC-dimension de $\mathcal{F}$ est le plus grand entier h tel que la fonction de croissance $G(\mathcal{F}, m) = 2^h$. Si un tel entier n'existe pas, alors la VC-dimension h est dite infinie.*

Plus intuitivement, la VC-dimension h d'une famille de fonction est donnée par le nombre maximum de points pouvant être séparés par une fonction de cette famille quel que soit l'étiquetage des points. Par exemple, dans un espace à deux dimensions, un séparateur linéaire peut, au maximum, séparer 3 points. En effet, étant donné 3 points fixes, pour toutes combinaisons d'étiquetage de ces points, il existe un séparateur linéaire pouvant les séparer. La figure 2.2 illustre cet exemple. Plus généralement, la VC-dimension d'un hyperplan séparateur de points dans un espace à n dimension est $n + 1$ [Vap95].

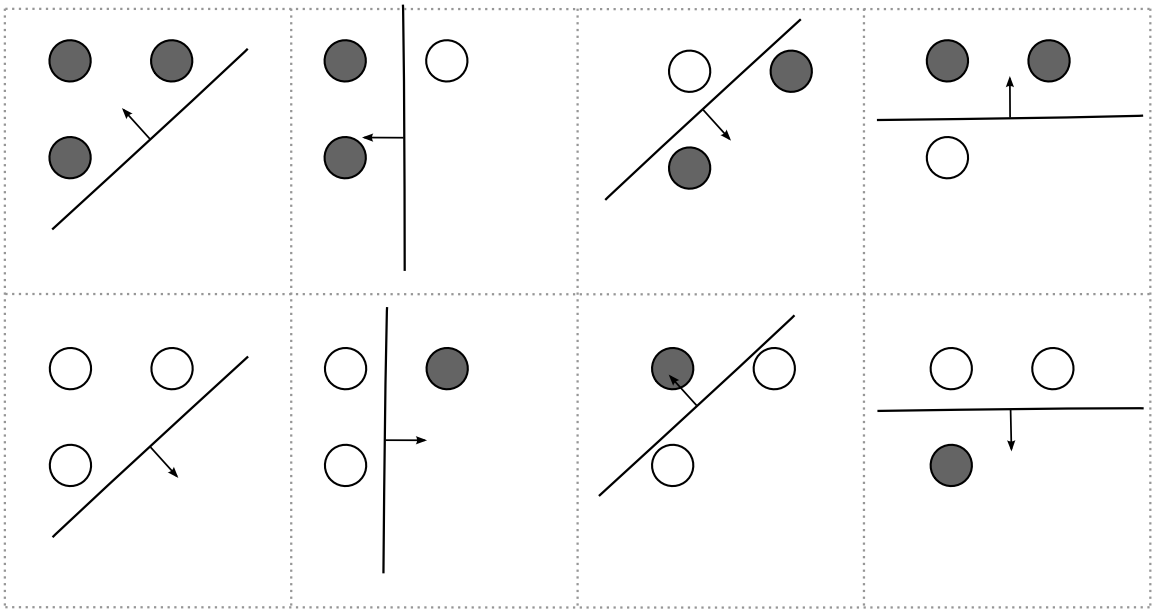


Fig. 2.2 – Nombre maximum de points fixes pouvant être séparés par des hyperplans quelque soit leurs classes.

D'après l'inégalité 2.9, une meilleure généralisation est obtenue en minimisant l'erreur empirique et la VC-confiance (second terme de droite). Pour minimiser la VC-confiance, la capacité du séparateur, à savoir la VC-dimension, doit être minimisée. Le principe de la minimisation du risque structurel (SRM) consiste à créer une structure dans l'ensemble d'hypothèses (ensemble de fonctions $\mathcal{F} = \{f_\alpha : \alpha \in \Lambda\}$) :

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \subset \mathcal{F}_n, \qquad \mathcal{F} = \cup_i \mathcal{F}_i \tag{2.11}$$

La structure est ordonnée selon les VC-dimensions croissantes : $h_1 \leq h_2 \leq \ldots \leq h_n$. Pour obtenir une fonction avec le risque de généralisation le plus faible, nous choisirons dans un premier temps le sous-ensemble $\mathcal{F}_i$ de fonctions avec la VC-dimension h_i minimisant la somme

du risque empirique et de la VC-confiance (terme à droite de l'inégalité 2.9). Puis dans un deuxième temps, la fonction $f_\alpha \in \mathcal{F}_i$ minimisant le risque empirique sera choisie pour être la fonction optimale. La figure 2.3 illustre le principe de la SRM.

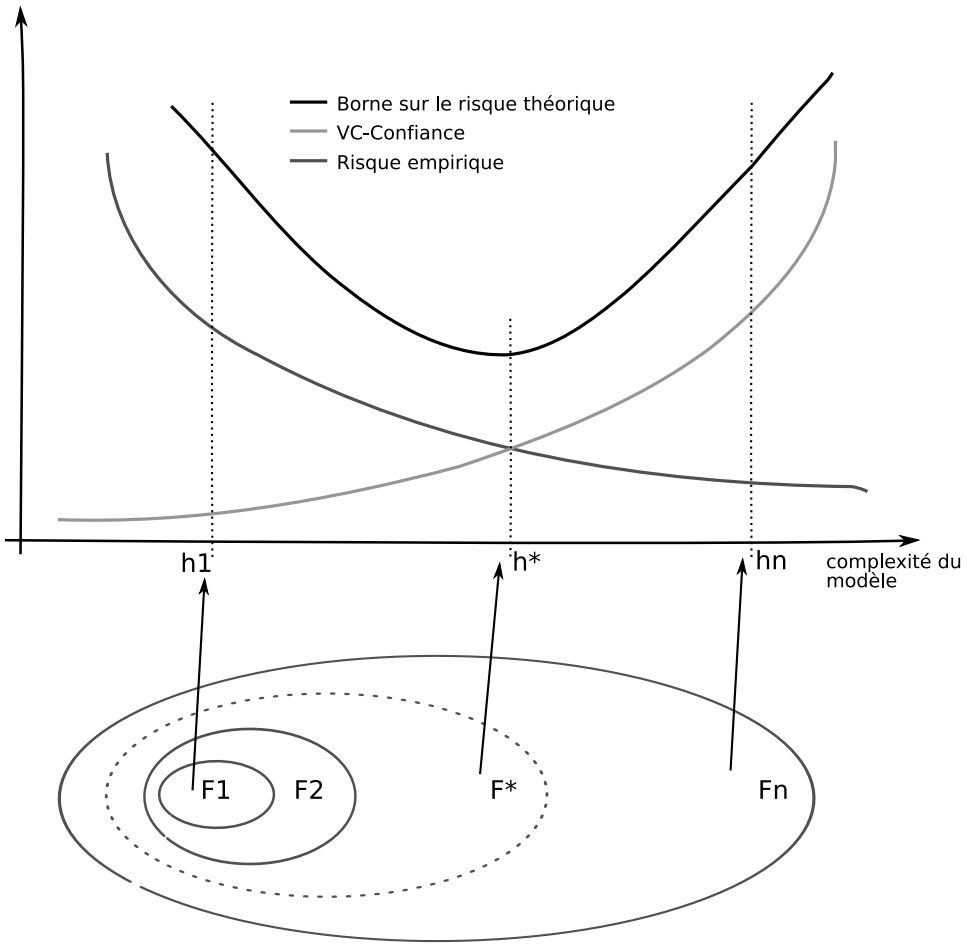


Fig. 2.3 – Minimisation du Risque Structurel.

Bien que la SRM permette d'obtenir des algorithmes avec un haut niveau de généralisation tels que les Séparateurs à Vaste Marge que nous étudierons dans la section 2.3, le calcul de la VC-dimension d'une famille de fonction n'est pas toujours évidente. Le calcul de VC-confiance peut dans ce cas être impossible. Lorsque la SRM ne peut être mise en oeuvre, la minimisation du risque empirique avec le terme de régularisation semble être un meilleur compromis.

2.2.1.2 **Approche discriminante**

Dans l'approche discriminante, les algorithmes d'apprentissage ne se soucient pas de la source ayant généré les données. Ainsi, la fonction de répartition $F(\mathbf{x}, y)$ apparaissant dans la formulation du risque théorique (équation 2.1) est laissée inconnue durant tout le processus d'apprentissage. La minimisation du risque empirique ou encore du risque structurel est utilisé

comme principe d'induction.

Les algorithmes discriminants tentent de segmenter l'espace en régions. Chaque région est délimitée pour ne contenir[1] que des données appartenant à une même catégorie. Pour la définition de ces algorithmes, il existe deux notions clés :

- la similarité : une fonction de similarité doit être définie pour permettre à l'algorithme d'identifier les groupes d'individus selon des critères induits par la fonction de similarité. L'algorithme sera ainsi capable "d'extraire" les éléments discriminants reliant les données et leurs classes. Ces éléments seront utilisés pour définir les paramètres de la fonction discriminante. Les fonctions de similarité les plus utilisés sont la distance ($||\mathbf{x} - \mathbf{y}||$) et l'orientation des données dans l'espace pouvant être obtenue par le produit scalaire.

- le principe d'induction : le critère d'induction permet "d'optimiser" les régions de l'espace en ajustant les frontières.

Parmi les approches discriminantes, nous citerons les algorithmes issus des Séparateurs à Vaste Marge [CV95, Vap95] que nous étudierons dans la section 2.3 et les méthodes connexionistes (les réseaux de neurones)[Bis95, Ben06].

2.2.1.3 **Approche Générative**

Les méthodes génératives tentent de caractériser la source ayant généré les données en modélisant leur distribution. Ainsi, avec une modélisation de la densité de probabilité $p(\mathbf{x}, y)$, le risque théorique devient :

$$R(f) = \int_{\mathcal{X}} \int_{\mathcal{Y}} \mathcal{L}(y, f(x)) P(x, y) dx dy \tag{2.12}$$

avec $\mathcal{L}(y, f(x))$ la fonction de coût définie par l'équation 2.3 et f la fonction de décision (fonction de classement).
La fonction de décision minimisant le risque théorique est la fonction de décision bayésienne :

$$f(x) = \text{argmax}_y P(x, y) = \text{argmax}_y P(y|x) \tag{2.13}$$

Il existe deux approches pour modéliser la distribution : l'approche paramétrique et l'approche non-paramétrique.

Méthodes paramétriques

Les méthodes paramétriques supposent connue la loi ayant généré les données. Ainsi pour

[1] Il peut toutefois y avoir une certaine tolérance. Notamment lorsque les données ne sont strictement séparable.

chaque catégorie y_i, une fonction de probabilité $g_i(x|\theta_\mathbf{i})$, suivant la même loi mais paramétrée par $\theta_\mathbf{i}$, lui est associé. Nous obtenons, ainsi, dans le cas discret :

$$P(x, y_i) = P(x|y_i)P(y_i) = g_i(x|\theta_\mathbf{i})P(y_i) \tag{2.14}$$

$P(y_i)$ est estimé empiriquement à partir de l'ensemble d'apprentissage : $P(y_i) = \frac{N_i}{N}$, N_i étant le nombre de données appartenant à y_i et le nombre de données dans l'ensemble d'apprentissage. L'expression de $P(y_i|x)$ nécessaire pour la fonction de décision est :

$$\begin{aligned} P(y_i|x) &= \frac{P(x|y_i)P(y_i)}{P(x)} \\ &= \frac{g_i(x|\theta_\mathbf{i})P(y_i)}{\sum_j g_j(x|\theta_\mathbf{j})P(y_j)} \end{aligned} \tag{2.15}$$

La modélisation des densités se fait en deux étapes :

1. choix d'un modèle (d'une loi),
2. estimation des paramètres $\theta_\mathbf{i}$.

L'un des modèles le plus utilisé est le modèle gaussien (loi normale) mais d'autres lois telle que la loi de Student peuvent aussi être utilisées.

L'estimation des paramètres s'effectue en général avec le critère du maximum de vraisemblance. Étant donné un ensemble d'apprentissage avec k classes $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, nous noterons : $\mathcal{D} = \cup_{i=1}^k \mathcal{D}_k$, avec $\mathcal{D}_k = \{(x_{i,k}, k)\}_{i=1}^{N_k}$, le critère du maximum de vraisemblance consiste à déterminer les paramètres qui maximisent :

$$\begin{aligned} l(\theta, \mathcal{D}) &= \prod_{j=1}^{k} \prod_{i=1}^{n_k} P(x_{i,k}, k|\theta) \\ &= \prod_{j=1}^{k} \prod_{i=1}^{n_k} g_i(x_{i,k}|\theta_\mathbf{i})P(y_i) \end{aligned} \tag{2.16}$$

Dans certains cas, comme le cas gaussien, il sera plus simple de maximiser la "log-vraisemblance" :

$$L(\theta, \mathcal{D}) = \sum_{i=1}^{k} \sum_{i=1}^{n_k} \log(g_i(x_{i,k}|\theta_\mathbf{i})) \tag{2.17}$$

Méthodes non-paramétriques

Les méthodes non-paramétriques ne font aucune supposition sur la loi sous-jacente aux données. Elles tentent de déterminer la densité en estimant les différentes distributions $P(x|y_i) = g_i(x)$ à partir des données d'apprentissage.
Il existe pour cela, essentiellement, trois méthodes :

– **Les K plus proches voisins (K-NN)** : cette méthode consiste à fixer un nombre k de données et à déterminer le volume minimal V, autour d'un point x, contenant au moins k données. La densité en x pour la classe i est alors donnée par la formule :

$$g_i(x) = \frac{1}{N_k}\frac{k}{V} \tag{2.18}$$

– **Les fenêtres de Parzen**[Par62] : dans cette méthode, pour un volume fixé, la densité est obtenue en déterminant le nombre de données se trouvant à l'intérieur du volume. Le volume est fixé à l'aide d'une fonction fenêtre $k(\frac{x-x_i}{h})$ autour d'un point x avec h la largeur de bande. Le volume est alors donné par : $V = h^d$ avec d la dimension de l'espace.

$$g_i(x) = \frac{1}{N_k(h^d)}\sum_{i=1}^{N_k} k(\frac{x-x_{i,k}}{h}) \tag{2.19}$$

La fenêtre la plus utilisée est la fenêtre gaussienne $k(\frac{x-x_i}{h}) = \frac{1}{\sqrt{2\pi}}\exp(-\frac{(x-x_i)^2}{2h^2})$.

– **Les modèles de mélange** : cette méthode est à la frontière des méthodes paramétriques et non-paramétriques. Elle repose sur l'utilisation d'une somme de densités simples pour modéliser des distributions complexes. L'un des mélanges les plus connus est le modèle de mélange gaussiens. Cette méthode est dite non-paramétrique dans le sens où aucune supposition n'est faite sur la loi sous-jacente aux données d'apprentissage. Pour un modèle de mélange à m composants, nous avons :

$$g_i(x|\theta_i) = \sum_{j=1}^{m} p_{i,j}g_{i,j}(x|\theta_{i,j}) \tag{2.20}$$

$p_{i,j}$ est la probabilité d'avoir le $j^{\text{ème}}$ composant du mélange de la classe i et $g_{i,j}$ est la densité modélisée par le $j^{\text{ème}}$ composant du mélange de la classe i.
Les paramètres $p_{i,j}$ et $\theta_{i,j}$ sont déterminés en utilisant l'algorithme EM (*Expectation-Maximization*) [MK97].

2.2.2 Apprentissage Semi-Supervisé

Bien souvent dans les problèmes d'apprentissage, les données étiquetées sont peu nombreuses alors que les données non-traitées sont disponibles en quantité. Nous avons vu, dans la section sur les principes d'induction, que plus la taille de l'échantillon est grande et plus l'estimateur obtenu par un principe d'induction est fiable. En outre, si l'échantillon d'apprentissage n'est pas suffisamment représentatif, les erreurs de généralisation des fonctions apprises risquent d'être importante. L'idée de l'apprentissage semi-supervisé est de limiter ces problèmes en utilisant les données non-étiquetées dans le processus d'apprentissage. Par exemple, la figure 2.4 illustre un cas où l'utilisation des données étiquetées (points en croix) permet d'affiner les régions des

instances positives et négatives.

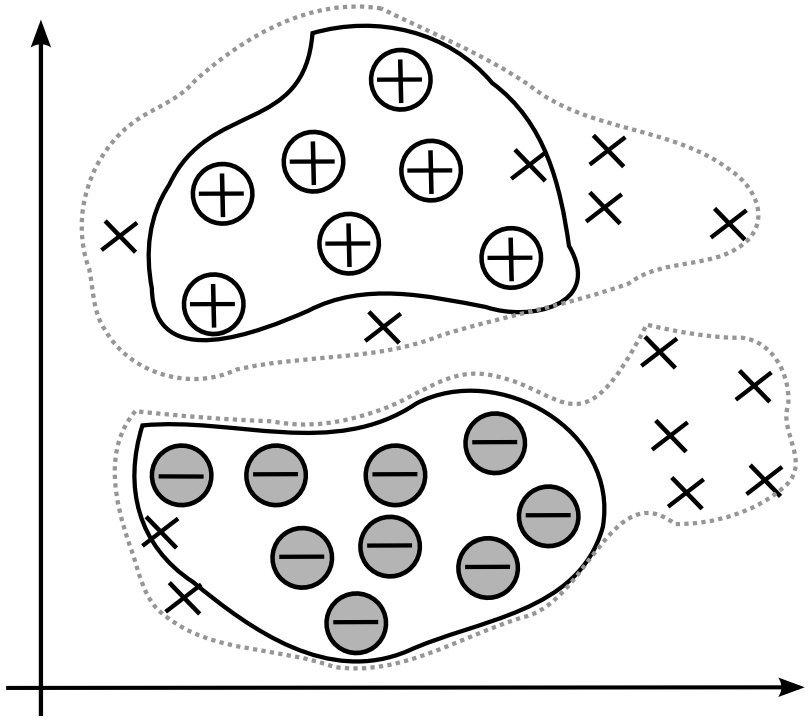


Fig. 2.4 – Amélioration de la zone de décision par apprentissage semi-supervisé.

Vapnik [Vap82, Vap95] défini l'apprentissage semi-supervisé comme étant un principe de transduction. L'apprentissage supervisé consiste à induire une fonction globale (principe d'induction) permettant de l'appliquer à toutes les données du domaines (principe de déduction). Le principe de transduction consiste à induire une fonction locale à partir des données d'apprentissage pour déduire des informations sur un ensemble de points d'interêts (ce sera dans le cas général l'ensemble de test). Les informations déduites seront, ensuite, ajoutées à l'ensemble d'apprentissage pour induire une fonction globale. L'utilisation de ce nouvel ensemble pour l'apprentissage est appelé le principe de minimisation du risque global (ORM-*Overall Risk Minimization*)[Vap82]. La figure 2.5 donne une illustration de ces principes.

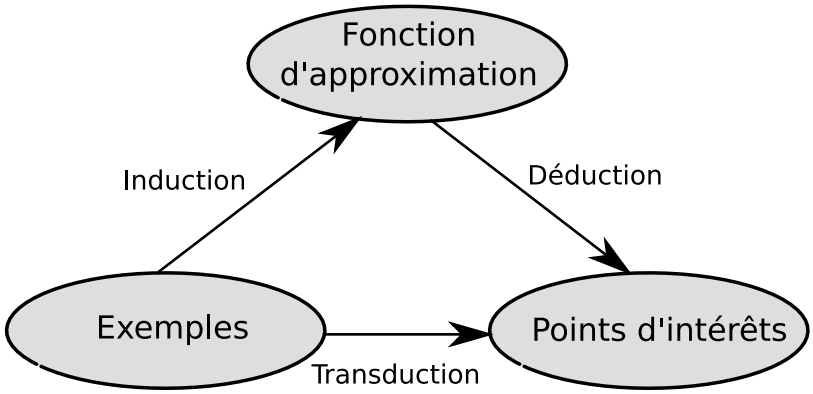


Fig. 2.5 – Principe de la transduction.

Dans la suite de cette section, nous exposerons quelques méthodes d'apprentissage semi-supervisé. Toutefois, nous ne présenterons pas, dans cette section, les Séparateurs à Vaste Marge semi-supervisés. Ils seront décrits plus loin dans la section 2.3.2.5.

2.2.2.1 **Les modèles génératifs**

Les modèles génératifs cherchent à estimer la distribution jointe des données et de leurs étiquettes à savoir $P(x, y)$. Pour cela, les méthodes paramétriques supposent que la distribution suit une loi connue. La méthode consiste alors à modéliser la densité jointe en utilisant cette loi paramétrée par un vecteur θ. Le problème consiste alors à déterminer θ afin de se rapprocher de la distribution inconnue.
Lorsqu'il existe des données non étiquetées, l'approche standard consiste à fixer θ à l'aide des données d'apprentissage puis à modifier ces paramètres afin de maximiser la probabilité que les données étiquetées et non étiquetées aient été générées par la même source [Haf05, See00, Zhu05]. Pour la phase d'optimisation, l'algorithme utilisé est l'Expectation-Maximisation (EM) [MK97]. L'algorithme EM est une méthode d'optimisation itérative constituée de deux étapes. Pour une itération t, la première étape est une étape d'estimation qui consiste à calculer la probabilité a posteriori des données inconnues (y) connaissant les données connues (x_i) :

$$p(y|x_i, \theta^t) = \frac{p(x_i, y|\theta^t)}{p(x_i|\theta^t)} = \frac{p(x_i, y|\theta^t)}{\sum_{y'} p(x_i, y'|\theta^t)} \tag{2.21}$$

La seconde étape consiste à maximiser, en fonction de θ, la probabilité que les données étiquetées et non étiquetées aient été générées avec θ :

$$\theta^{t+1} = \text{argmax}_\theta \sum_i \log p(x_i|\theta) - \log p(x_i|\theta^t)$$

avec :

$$\sum_i \log \frac{p(x_i|\theta)}{p(x_i|\theta^t)} \geq \sum_i \sum_y p(y|x_i, \theta^t) \log \frac{p(x_i, y|\theta)}{p(x_i, y|\theta^t)}$$

Ainsi nous avons :

$$\theta^{t+1} = argmax_\theta \sum_i \sum_y p(y|x_i, \theta^t) \log p(x_i, y|\theta) \tag{2.22}$$

Dans le cas de l'apprentissage supervisé, on ajoute une étape intermédiaire, appelée "classement", entre l'étape d'estimation et de maximisation. Cette variante est appelée l'algorithme CEM [CG92].
En effet, après l'étape d'estimation, on attribue à chaque donnée incomplète x_i l'étiquette y_i qui maximise la probabilité a posteriori $p(y|x_i, \theta^t)$. Ainsi, dans l'étape de maximisation, il n'est plus utile de marginaliser $p(x_i|\theta^t)$. L'équation 2.22 est modifiée en tenant compte des données complètes et des données nouvellement étiquetées :

$$\theta^{t+1} = argmax_\theta \sum_i p(y_i|x_i, \theta^t) \log p(x_i, y_i|\theta) \tag{2.23}$$

Dans [NMM06, NMTM00], les auteurs ont utilisé cette approche avec le classifieur naïf de Bayes utilisant la loi multinomiale. Ils ont aussi proposé des variantes en intégrant des poids pour accorder moins d'importance aux données nouvellement étiquetées utilisées dans la phase de maximisation (équation 2.23). De plus, un modèle de mélange a été introduit pour permettre de mieux modéliser les données.
Les expériences de catégorisation ont été menées par les auteurs sur des données textuels issus des corpus *20 Newsgroup* et *webkb*. Le corpus *20 Newsgroup* est constitué de 20017 messages issus de 20 différents groupes de discussion. Le corpus *webkb* contient 8145 pages du site internet de quatre départements de plusieurs universités américaines. Les pages sont classées en sept catégories dont seul quatre ont été utilisées pour les expériences.
Les résultats montrent que l'utilisation de données non étiquetées pour l'apprentissage améliore considérablement le taux de bon classement. En outre, l'utilisation du modèle de mélange, et dans un moindre sens les poids, améliore significativement les résultats comparés au modèle de base.

Dans [AG02], trois méthodes sont proposées pour résoudre un problème de résumé automatique. Il s'agit de classer chaque phrase comme "pertinent" ou "non pertinent" selon que la phrase doit faire partie ou non du résumé.
La première méthode utilise le classifieur naïf de bayes. La seconde est une méthode utilisant un mélange de deux gaussiennes optimisées avec l'algorithme CEM. La troisième méthode est une méthode discriminative dans le sens où elle estime $p(y|x)$ sans tenir compte de $p(x)$. Cette méthode utilise la fonction de régression logistique :

$$G(x) = \frac{1}{1 + \exp(-(\beta_0 + \beta^t.x))}$$

pour estimer la pertinence de x ($p(y = 1|x)$) et $1 - G(x)$ pour estimer la non-pertinence ($p(y = 0|x)$) de x. Le modèle logistique est initialement entraîné avec les données étiquetés.

Puis, l'algorithme CEM est utilisé pour optimiser $G(x)$ sur les données non étiquetées.
Les expériences ont été menées sur le corpus de résumé de Reuters, composé de 1000 documents, et sur le corpus *T*ipster Summac composé de 183 articles scientifiques. Les résultats montrent que la méthode logistique est légèrement plus performante que la méthode gaussienne. Cependant ces deux dernières méthodes obtiennent de meilleurs résultats que la méthode naïve de Bayes.

2.2.2.2 L'auto-apprentissage

L'auto-apprentissage (*self-training*) [Zhu05] consiste à entraîner un classifieur avec les données étiquetées (D_L). Le classifieur est, ensuite, utilisé pour étiqueter les données incomplètes (D_U). Les données étiquetées avec un haut degré de confiance sont ajoutées aux données d'apprentissage (D_L). Le classifieur est ré-entraîné sur les données de D_L et la procédure est répétée jusqu'à satisfaire un critère d'arrêt.
Parmi les variantes de l'auto-apprentissage, on peut citer la méthode qui consiste à éliminer de D_L des données ayant été mal classées. En effet, le fait d'utiliser des étiquettes erronées dans la phase d'apprentissage risque d'entraîner un renforcement de l'erreur. Toutefois, le problème est de détecter ces données étiquetées étant donné que l'étiquette a été attribuée avec un degré de confiance jugé suffisamment élevé. Ce problème est souvent résolu par l'utilisation d'heuristiques se basant sur le degré de confiance du classifieur. Ainsi, si la précision du classifieur chute en dessous d'un seuil, certaines données issues de D_U sont retirées de D_L pour être replacées dans D_U. Une autre heuristique, utilisée dans le cas de classifieur binaire, consiste à permuter les étiquettes des paires de données nouvellement étiquetées et obtenant de très faibles scores. Cette permutation a pour objectif d'augmenter le degré de confiance du classifieur. Une telle heuristique est notamment utilisée dans le cas des SVM transductives [Joa99b, Joa02] que nous verrons dans la section 2.3.2.5.

2.2.2.3 Le co-apprentissage

Le co-apprentissage (*co-training*) [BM98b] peut être perçu comme une extension de la méthode de l'auto-apprentissage.
Le co-apprentissage part du principe que s'il existe "deux vues" d'un même objet, indépendantes entre elles, alors deux classifieurs entraînés avec des vues différentes devront attribuer la même étiquette à l'objet. Ainsi, la méthode repose, pour des objets définis par un ensemble d'attributs tel que cet ensemble peut être divisé en deux sous-ensembles indépendants, sur l'utilisation de deux classifieurs. Chaque classifieur est entraîné dans un sous-espace avec les données d'apprentissage D_L. Les classifieurs sont ensuite utilisés pour étiqueter les données de D_U. Les données étiquetées avec un important degré de confiance sont ajoutées à D_L. La sélection et l'ajout des données dans D_L peuvent s'effectuer en tenant compte des proportions de chaque étiquette afin de rester proche de la distribution originale des données. Ensuite, la phase d'apprentissage des classifieurs est réitérée sur le nouvel ensemble d'apprentissage. Lorsque l'apprentissage est

terminé, les deux classifieurs peuvent être combinés pour classifier.

Dans [BM98b], un sous-ensemble $D'_U \subseteq D_U$ est utilisé en choisissant aléatoirement des éléments de D_U. Ainsi, pour chaque classifieur les p et n éléments étiquetés positivement, et respectivement, négativement et ayant les scores les plus importants sont ajoutés à D_L. La taille de D'_U est maintenue en transférant des données de D_U vers D'_U.
Cet algorithme a été évalué sur la base *webkb*. Chaque page P est représenté par un doublet (V_P, V_l). V_P est un sac de mot obtenu à partir des mots de la page et V_l est un sac de mot obtenu à partir des mots se trouvant sur les hyper-liens présents dans les autres pages et pointant vers la page P. Ainsi, les données se divisent naturellement en deux vues V_P et V_l. L'algorithme naïf de Bayes est utilisé comme classifieur pour chacune des deux vues. Les résultats montrent que le co-apprentissage diminue le taux d'erreur jusqu'à un facteur deux comparé à la méthode supervisée.

Dans [NG00], à chaque itération de l'algorithme de co-apprentissage, chaque classifieur choisit, pour chaque étiquette i, un élément de D_U ayant le score le plus élevé pour l'étiquette i et le place dans D_L. Ainsi, à chaque itération l'ensemble d'apprentissage sera augmenté de $2 * N$ éléments avec N correspondant aux nombres d'étiquettes (de classes). Ensuite, les classifieurs sont ré-entraînés avec le nouvel ensemble d'apprentissage. L'algorithme naïf de Bayes est, ici, aussi utilisé comme classifieur. En outre, les auteurs proposent un algorithme hybride mélangeant l'algorithme *EM* avec le co-apprentissage appelé *co-EM*. Les deux classifieurs sont initialisés par apprentissage sur les données étiquetées. Ensuite pour chaque classifieur, on effectue une étape d'estimation et une étape de maximisation avant de réitérer l'algorithme. Pour un classifieur f à l'itération i, l'étape d'estimation est obtenue en étiquetant les données de D_U avec l'autre classifieur f' de l'itération $i-1$ et l'étape de maximisation s'effectue en ré-entraînant f avec D_L et les données étiquetées par f'.
Il faut noter que l'algorithme *co-EM* est un algorithme itératif alors que le co-apprentissage initial [BM98b] est un algorithme incrémental. En effet, pour le co-apprentissage initial, l'ensemble D_L s'agrandit au fur et à mesure des itérations. Tandis que le *co-EM* utilise uniquement D_L pour l'initialisation et $D_L \cup D_U$ (D_U nouvellement étiqueté par l'autre classifieur) dans le reste des itérations.
Les expériences ont été menées pour comparer les méthodes de co-apprentissage, *co-EM*, auto-apprentissage et *EM*. L'algorithme de classification utilisé est toujours le classifieur naïf de Bayes.
Sur le corpus *webKB*, les méthodes semi-supervisées obtiennent les mêmes résultats que la méthode supervisée (naïf Bayes) en utilisant uniquement 1.5% des données étiquetés. Parmi les algorithmes semi-supervisés *EM* est légèrement plus performant que le co-apprentissage. Parmi les hypothèses expliquant cela, les auteurs pensent que les vues utilisées dans *webKB* ne sont pas complètement indépendantes. Sur des sous-ensembles du corpus *20 Newsgroup*, les méthodes de co-apprentissage et *co-EM* sont meilleures que les autres lorsque la condition d'indépendance est respectée dans le choix des vues. On notera que la méthode *co-EM* est légèrement plus

performante que le co-apprentissage.

Dans [DGLT03], le co-apprentissage a été utilisé en ayant uniquement des données positives et des données non étiquetées. Le classifieur utilisé est l'algorithme naïf de Bayes avec pour entrée $p(y = 1)$, la probabilité que la source génère un document positif. La probabilité d'avoir un document négatif est alors $p(y = 0) = 1 - p(y = 1)$.
L'algorithme de co-apprentissage est utilisé comme décrit dans [BM98b]. Cependant, les données de D_U étiquetées négativement sont ajoutées à l'ensemble des exemples négatifs initialement vide D_{NL}. La phase d'apprentissage prend alors en compte D_{NL} tout en lui accordant un poids inférieur :

$$p(y = 0|x) = (1 - \alpha).p(y = 0|x, D_L \cup D_U) + \alpha.p(y = 0|x, D_{NL})$$

Dans [BS04], les auteurs ont utilisé deux machines à vecteurs de supports (SVM) comme classifieurs pour l'algorithme *co-EM*.

2.2.3 **Apprentissage Non-Supervisé**

L'apprentissage non-supervisé est une analyse exploratoire des données. Contrairement aux tâches supervisées, les méthodes de cette catégorie ne requièrent pas que les données utilisées pour construire les modèles soient étiquetées.
La classification (*clustering*) fait partie des tâches non-supervisées. Son objectif consiste à découvrir des groupes homogènes d'individus. Les individus sont regroupés selon des critères de similarités tels que la distance. Chaque groupe est alors caractérisé par un individu moyen appelé centroïde ou prototype. Le classement d'une nouvelle donnée consiste à lui associer le groupe dont le prototype est le plus similaire à la donnée. Parmi les algorithmes de classification, on peut citer :

- K-Means [Mac67] qui permet de regrouper les données en K groupes représentés par K centres. Les centres sont fixés initialement de manière aléatoire. Puis, chaque individu est classé dans la classe dont le centre est le plus proche (similaire) de l'individu. Les nouveaux centres sont calculés en effectuant la moyenne des individus dans chaque classe et l'algorithme réitère l'opération de classement jusqu'à la stabilisation des centres.

- Les cartes auto-organisatrices (SOM : *Self Organizing Map*) de Kohonen [Koh97, CILR06] qui sont des réseaux bidimensionnels de neurones. À chaque neurone est associé un prototype de classe. Lors de l'apprentissage lorsqu'un individu est présenté, le neurone le plus proche de l'individu ainsi que les neurones voisins sont déplacés dans la direction de l'individu. Au fur et à mesure de l'apprentissage, le voisinage des neurones gagnants et le facteur de déplacement sont diminués. L'apprentissage se termine lorsque la carte s'est stabilisée.

Les méthodes non-supervisées peuvent aussi être utilisées pour effectuer de la réduction de dimension. L'objectif est de trouver une fonction de projection projetant les données dans un espace de dimension moindre où l'information contenue dans les données y est résumée. L'analyse en composante principale (ACP) [STC04] est un exemple d'algorithme. Cette méthode projette les données dans un sous-espace maximisant la variance des variables. Chaque axe du sous-espace représente une nouvelle variable obtenue par combinaisons linéaires de variables et est appelée composante principale.

2.3 Les Séparateurs à Vaste Marge (SVM)

Depuis un peu plus d'une dizaine d'année, une bonne partie de la recherche en apprentissage statistique s'est focalisée sur la famille des Séparateurs à Vaste Marge (SVM). Les SVM sont le résultat de l'application du principe de la Minimisation du Risque Structurel (SRM) proposé par Vapnik à la théorie bien étudiée des hyperplans séparateurs linéaires. L'intérêt suscité par les SVM est essentiellement dû à deux facteurs.

Le premier facteur est le fait que les SVM obtiennent des performances qui sont généralement parmi les meilleures dans l'apprentissage. Ces résultats proviennent du bon niveau de généralisation induit par la SRM. A la vue de ces performances, de nombreuses variantes du SVM ont été développées pour traiter différents type de problèmes.

Le second facteur, expliquant le succès des SVM, est l'utilisation des noyaux pour "transformer" le SVM en un algorithme non-linéaire pouvant être appliqué sur des données variées. Les noyaux et les SVM ont alors permis d'utiliser l'apprentissage numérique dans des problèmes traitant des données complexes telles que les données textuelles.

Dans la suite de cette section, nous ferons un point sur les principales variantes des SVM. Puis, nous terminerons par présenter les noyaux.

2.3.1 L'hyperplan Séparateur Optimal

L'application du principe d'induction de la Minimisation du Risque Structurel (SRM), vu à la section 2.2.1.1, au problème de classement binaire par un hyperplan a donné lieu à une famille d'algorithme puissant : les Séparateurs à Vaste Marge [CV95, Vap95].

Le problème du classement binaire par un hyperplan consiste à trouver un hyperplan séparant l'espace des données en un sous-espace d'instances positives et un sous-espace d'instances négatives. Plus formellement, étant donné un ensemble de données $\mathcal{D} = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_N}, y_N)\}$ avec $\mathbf{x} \in \mathbb{R}^n$ et $y \in \{-1, +1\}$, le problème consiste à déterminer un hyperplan de vecteur normal $\mathbf{w}$ tel que :

$$\forall(\mathbf{x_i}, y_i) \in \mathcal{D} : y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \geq 0 \tag{2.24}$$

La solution y pour le vecteur $\mathbf{x}$ est alors donnée par :

$$y = \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle + b) \tag{2.25}$$

La minimisation du risque empirique, pour ce problème, conduit à un espace de solutions où toutes les hypothèses commettent le même nombre d'erreur sur $\mathcal{D}$. La figure 2.6 illustre un cas avec plusieurs solutions.

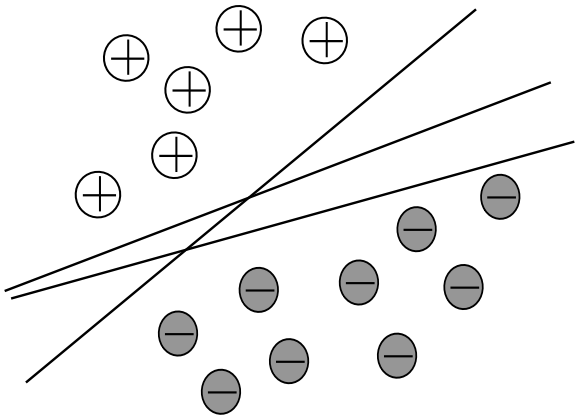


Fig. 2.6 – Exemples de séparateurs linéaires minimisant le risque empirique (cas séparable).

La SRM implique que la solution optimale doit non seulement minimiser le risque empirique, à savoir les erreurs sur $\mathcal{D}$, mais doit, aussi, être de capacité de séparation minimale. Elle doit avoir la plus faible VC-dimension parmi les solutions de l'espace d'hypothèse. Ainsi, la SRM permet de restreindre l'espace de solutions obtenus par l'ERM. Il faudra, ainsi, choisir la solution ayant la plus faible VC-dimension.
Nous avons vu que dans le cas des hyperplans séparateurs de points, la VC-dimension de ces fonctions étaient de $n + 1$ avec n étant la dimension de l'espace. Cependant le théorème 2.3.1 indique que l'hyperplan ayant la plus faible norme $||\mathbf{w}||$ aura la plus faible VC-dimension. La SRM implique donc de minimiser l'ERM et la norme de $\mathbf{w}$ pour obtenir l'hyperplan séparateur optimal.

Theorème 2.3.1 ([Vap95]). *Soit un ensemble, $\mathcal{D}$, de points de $\mathbb{R}^n$, une hypersphère de rayon R et de centre a tel que $\forall \mathbf{x} \in \mathcal{D}, |\mathbf{x} - a| \leq R$, un ensemble $\mathcal{S}$ d'hyperplans séparateurs paramétrés par $(\mathbf{w}, b)$ tel que $min_{\mathbf{x} \in \mathcal{D}} |\mathbf{w}.\mathbf{x} + b| = 1$, un sous-ensemble de $\mathcal{S}$ aura pour VC-dimension $h \leq \min(R^2||\mathbf{w}||^2, n) + 1$.*

La condition $min_{\mathbf{x} \in \mathcal{D}} |\mathbf{w}.\mathbf{x} + b| = 1$ du théorème 2.3.1 exprime une normalisation de $(\mathbf{w}, b)$ pour limiter l'espace des hypothèses aux hyperplans canoniques. Ainsi, les points x_{SV_i} les plus proches de l'hyperplan séparateur doivent vérifier l'équation :

$$|\langle \mathbf{w}, \mathbf{x_{SV_i}} \rangle + b| = 1 \tag{2.26}$$

Plus simplement, si $y_{SV_i} \in \{-1; +1\}$ est l'étiquette de x_{SV_i}, nous avons :

$$y_{SV_i}(\langle \mathbf{w}, \mathbf{x_{SV_i}} \rangle + b) = 1 \tag{2.27}$$

Il est aisé de montrer que les hyperplans déterminés par $(\mathbf{w}, b)$ vérifiant les équations ci-dessus minimisent l'ERM. En effet, pour une instance de $(\mathbf{w}, b)$, les x_{SV_i} sont les points de $\mathcal{D}$ les plus proches de l'hyperplan, par conséquent :

$$\forall(\mathbf{x_i}, y_i) \in \mathcal{D}, y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \geq 1 \tag{2.28}$$

Les vecteurs $\mathbf{x_{SV_i}}$ sont appelés les vecteurs supports de l'hyperplan et nous verrons plus tard que seuls ces vecteurs sont nécessaires pour déterminer l'hyperplan séparateur optimal.

Jusque là, nous avons caractérisé l'ensemble des hyperplans canoniques minimisant l'ERM. Il ne reste plus qu'à caractériser l'hyperplan optimal ayant la VC-dimension la plus faible. Pour cela, nous devons minimiser la norme de $\mathbf{w}$ ou encore maximiser $\frac{1}{||\mathbf{w}||}$. L'expression $\frac{1}{||\mathbf{w}||}$ désigne la distance entre l'hyperplan séparateur H d'équation $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ et l'hyperplan parallèle H_+ d'équation $\langle \mathbf{w}, \mathbf{x} \rangle + b = 1$. En effet, si nous désignons x un point appartenant à H_+ et x' la projection orthogonal de x sur H, la résolution du système :

$$\begin{cases} \langle \mathbf{w}, \mathbf{x} \rangle + b = 1 \\ \langle \mathbf{w}, \mathbf{x'} \rangle + b = 0 \end{cases} \tag{2.29}$$

amène à :

$$\text{dist}(x, x') = \frac{1}{||\mathbf{w}||} \tag{2.30}$$

Symétriquement, nous avons la même chose pour l'hyperplan H_- d'équation $\langle \mathbf{w}, \mathbf{x} \rangle + b = -1$. La distance entre H et H_- est $\text{dist}(H, H_-) = \frac{1}{||\mathbf{w}||}$.

Nous appellerons la marge de l'hyperplan séparateur la distance entre H_+ et H_- qui est : $M = \text{dist}(H_+, H_-) = \frac{2}{||\mathbf{w}||}$. La figure 2.7 illustre ce principe. Ainsi, minimiser la norme de $\mathbf{w}$ revient à maximiser la marge. L'hyperplan séparateur optimal sera donc celui qui aura la plus grande marge.

Selon le principe de la SRM, les hyperplans maximisant la marge ont un niveau de généralisation élevé, à savoir que les erreurs sur des données inconnues seront minimisées. Nous pouvons expliquer cela, intuitivement, par une tolérance au bruit. En effet, si nous remplaçons les points de $\mathcal{D}$ par des hypersphères s_i de rayon r et de centre x_i pour $x_i \in \mathcal{D}$, un hyperplan séparateur d'hypersphères aura un meilleur niveau de généralisation qu'un séparateur de point. D'une part, le séparateur d'hypersphères sera plus tolérant au bruit étant donné qu'il ne tient pas compte d'un point précis de l'espace mais d'une région sphérique, à savoir que le point peut se trouver n'importe où dans cette région sans altérer les performances du système. D'autre part, ce séparateur est moins sensible au sur-apprentissage pour la même raison qu'en cas de sur-apprentissage, ce sont des régions qui seront mémorisées et non des poins précis. Ainsi, plus le rayon r des hypersphères sera grand et plus les performances seront bonnes. Toutefois, il faudra ajouter les contraintes de séparabilité vu ci-dessus. Les hypersphères d'instances positives et négatives doivent être linéairement séparables. Par conséquent, les points les plus proches

de l'hyperplan doivent être à une distance minimale de r de l'hyperplan séparateur. Ainsi, $r = \frac{1}{||\mathbf{w}||}$. Par conséquent, maximiser la marge revient à maximiser le rayon r des hypersphères. La VC-dimension des séparateurs d'hypersphères est $h \leq \min(\frac{R^2}{r^2}, n) + 1$.

Nous soulignerons un dernier point qui est très important. Les séparateurs maximisant la marge seront moins influencés par les espaces de haute dimension. En effet, si $\frac{R^2}{r^2} < n$, la capacité de ces séparateurs ne dépendra pas de la dimension de l'espace.

Nous verrons dans les sous sections suivantes, des algorithmes appartenant à la famille d'algorithme minimisant l'ERM et $\mathbf{w}$, appelée famille des Séparateurs à Vaste Marge.

2.3.2 Les SVM pour le classement

Dans cette section, nous présentons les principaux algorithmes de classement basés sur les Séparateurs à Vaste Marge. Il est à noter que ces algorithmes peuvent aussi être utilisés pour d'autres objectifs que le classement. Par exemple, dans [Rak03], une méthode itérative de sélection de variables a été proposée. Cette méthode repose sur des critères utilisant les paramètres des SVM.

2.3.2.1 Les SVM pour le cas séparable

Dans cette section, nous décrivons la méthode pour déterminer l'hyperplan séparateur résultant de la minimisation du SRM dans le cas où les données sont linéairement séparables tel qu'illustré par la figure 2.7.

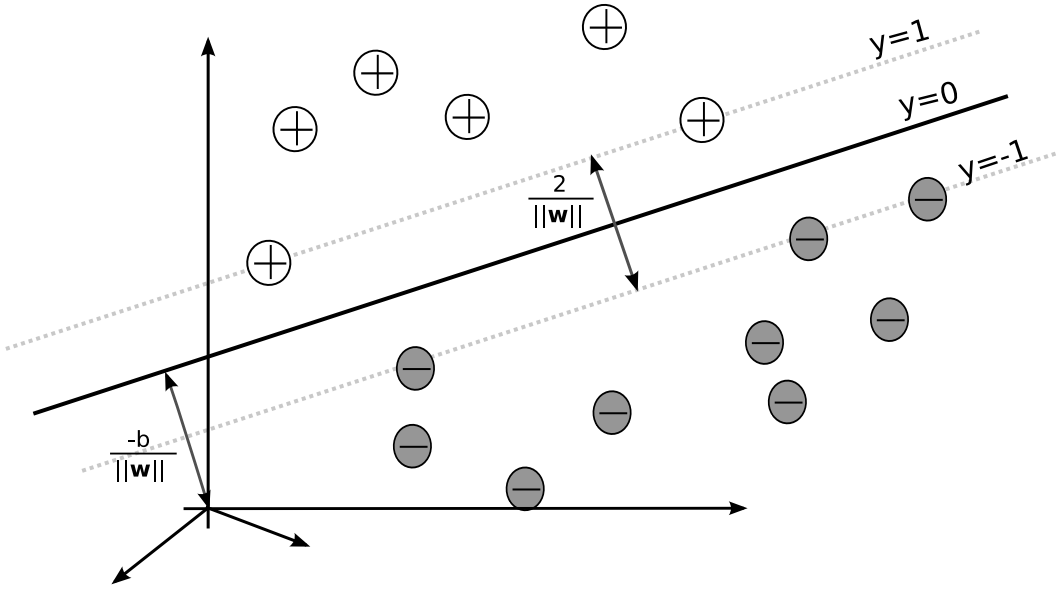


Fig. 2.7 – L'hyperplan séparateur dans le cas de données linéairement séparables.

La maximisation de la marge, vu dans la section précédente, nous amène au problème quadratique suivant :

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s. c.} \quad & y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \geq 1 \\ & i \in \{1, \ldots, N\} \end{aligned} \tag{2.31}$$

La formulation duale est en général plus utilisée pour résoudre le problème quadratique. En effet, la forme duale nous permet de passer d'un problème d'optimisation dans $\mathbb{R}^n$ (dépendant de la dimension de l'espace) à un problème d'optimisation dans $\mathbb{R}^N$ (dépendant du nombre d'exemples d'apprentissage). En utilisant, les multiplicateurs de Lagrange et les conditions de Karush-Kuhn-Tucker (KKT), nous arrivons à la formulation duale, dite de Wolfe [Fle87], suivante [Bur98] :

$$\begin{aligned} \min_{\alpha} \quad & -\sum_i \alpha_i + \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle \\ \text{s. c.} \quad & \sum_i y_i \alpha_i = 0 \\ & \forall i \in \{1, \ldots, N\}, \quad \alpha_i \geq 0 \end{aligned} \tag{2.32}$$

La résolution du problème amène à la solution optimale suivante :

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x_i} \tag{2.33}$$

$$b = y_{sv} - \langle \mathbf{w}, \mathbf{x_{sv}} \rangle \tag{2.34}$$

avec $(\mathbf{x_{sv}}, y_{sv}) \in \mathcal{S} = \{(\mathbf{x_i}, y_i) \in \mathcal{D} : \alpha_i \neq 0\}$. L'ensemble $\mathcal{S}$ est appelé ensemble des vecteurs supports. Il est à noter que de par les conditions de KKT :

$$\forall i, \alpha_i (y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) - 1) = 0 \tag{2.35}$$

Seuls les vecteurs appartenant à H_+ et H_-, vus plus haut, seront associés à des α_i non nuls.

2.3.2.2 Les C-SVM pour le cas non-séparable

Dans la majorité des cas, les données d'apprentissage, $\mathcal{D}$, ne sont pas linéairement séparables. Pour répondre à ce problème, des variables d'écarts ξ_i sont introduites. Ces variables permettent d'autoriser, pour chaque point x_i, un certain degré d'erreur dans le classement :

$$y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle) \geq 1 - \xi_i \tag{2.36}$$

Le nombre d'erreur de classement permis est contrôlé par une valeur prédéfinie C. Cette valeur doit être fixée de manière empirique. L'introduction de C et des variables d'écarts nous amènent à un nouveau problème quadratique dans le primal :

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s. c.} \quad & y_i(\langle \mathbf{w}, \mathbf{x_i}\rangle + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & i \in \{1, \ldots, N\} \end{aligned} \tag{2.37}$$

La formulation duale de ce problème est alors :

$$\begin{aligned} \min_{\alpha} \quad & -\sum_i \alpha_i + \frac{1}{2}\sum_{i,j} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j}\rangle \\ \text{s. c.} \quad & \sum_i y_i \alpha_i = 0 \\ & \forall i \in \{1, \ldots, N\}, \quad 0 \leq \alpha_i \leq C \end{aligned} \tag{2.38}$$

La solution est alors donnée par :

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x_i} \tag{2.39}$$

$$b = y_{usv} - \langle \mathbf{w}, \mathbf{x_{usv}}\rangle \tag{2.40}$$

avec $(\mathbf{x_{usv}}, y_{usv}) \in \mathcal{U} \subseteq \mathcal{S}$, $\mathcal{S}$ étant l'ensemble des vecteurs supports à savoir des vecteurs ayant un multiplicateur de lagrange tel que $0 < \alpha_i \leq C$ et $\mathcal{U}$ un sous-ensemble de vecteurs avec $0 < \alpha_i < C$.
Il peut arriver dans des cas exceptionnels que l'ensemble $\mathcal{U}$ soit vide. Dans ce cas, il existe plusieurs solutions optimales différentes par la valeur de b [BC99]. Dans ce cas, il est possible d'arriver à une solution unique en introduisant deux vecteurs artificiels $\mathbf{x'_1}$ et $\mathbf{x'_2}$, puis en ajoutant les deux contraintes suivantes au problème d'optimisation 2.37 : $\langle \mathbf{w}, \mathbf{x'_1}\rangle + b \geq 1$ et $-\langle \mathbf{w}, \mathbf{x'_2}\rangle + b \geq 1$. Ceci amène à un problème stabilisé [Joa02].

2.3.2.3 Les ν-SVM

Le choix du paramètre C, dans les C-SVM et les SVM pour la régression que nous verrons dans la section 2.3.5, est une tâche difficile. En effet, la constante C peut être perçue comme un paramètre de régularisation pour éviter les oscillations provoquées par les variables d'écarts, à savoir la tolérance aux erreurs dans la base d'apprentissage. Elle peut aussi être perçue comme une quantité d'erreur autorisé pour les vecteurs se trouvant dans la zone de marge ou dans la mauvaise zone. Il est ainsi difficile d'avoir une estimation de la valeur de C. Par conséquent,

elle doit être fixée empiriquement en fonction de la base d'apprentissage.
Le souhait de remplacer C par un paramètre ν ayant une signification, plus concrète, liée à l'apprentissage a donné lieu à la famille des ν-SVM [SSWB00]. Cette famille comprend le classifieur ν-SVM (ou ν-SVC) et l'algorithme ν-SVR pour la régression que nous verrons dans la section 2.3.5.

Dans le $\nu - SVM$, la constante C est remplacée par une constante $\nu \in [0, 1]$ dont nous verrons plus tard la signification et une variable ρ à optimiser. La variable ρ permettra de définir une marge plus souple. En effet, nous avons vu que la marge M est égal à $M = \frac{2}{||\mathbf{w}||}$. Une définition plus souple sera alors donnée par $M = \frac{2\rho}{||\mathbf{w}||}$. L'équation de l'hyperplan H_+ (resp. H_-) parallèle à l'hyperplan séparateur sera donnée par : $\langle \mathbf{w}, x \rangle + b = \rho$ (resp. $-\rho$). Ainsi, un meilleur niveau de généralisation sera obtenu en maximisant ρ et en minimisant $||\mathbf{w}||$. La constante ν permettra de doser l'importance de ρ dans la fonction objective. Le problème d'optimisation dans le primal est exprimé par :

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\rho} \quad & \frac{1}{2}||\mathbf{w}||^2 - \nu\rho + \frac{1}{N}\sum_{i=1}^{N}\xi_i \\ \text{s. c.} \quad & y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \geq \rho - \xi_i \\ & \xi_i \geq 0 \text{ et } \rho \geq 0 \\ & i \in \{1, \ldots, N\} \end{aligned} \tag{2.41}$$

La formulation duale, après introduction des multiplicateurs de Lagrange et des conditions de K.K.T., est alors :

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1,j=1}^{N}\alpha_i\alpha_j y_i y_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle \\ \text{s. c.} \quad & 0 \leq \alpha_i \leq \frac{1}{N}, \quad i \in \{1, \ldots, N\} \\ & \sum_i \alpha_i y_i = 0 \\ & \sum_i \alpha_i \geq \nu \end{aligned} \tag{2.42}$$

La solution de ce problème est alors :

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x_i} \tag{2.43}$$

$$b = -\frac{1}{2|S_+|}\sum_{\mathbf{x}\in S_+ \cup S_-}\langle \mathbf{w}, \mathbf{x} \rangle \tag{2.44}$$

$$\rho = \frac{1}{2|S_+|}\Big(\sum_{\mathbf{x}\in S_+}\langle \mathbf{w}, \mathbf{x} \rangle - \sum_{\mathbf{x}\in S_-}\langle \mathbf{w}, \mathbf{x} \rangle\Big) \tag{2.45}$$

avec S_+ (resp. S_-) un ensemble non vide de vecteurs supports $\mathbf{x_i}$ tel que $0 < \alpha_i < \frac{1}{N}$ et $y_i = 1$ (resp. y_i=-1) et $|S_+| = |S_-|$.

Le paramètre ν peut être caractérisé par les proposition suivantes :

Proposition 2.3.2. *Supposons que le produit scalaire $< ., . >$ utilisé dans ν-SVM est une fonction réelle et que nous obtenons un résultat avec $\rho > 0$ alors :*

1. *ν est la borne supérieur de la fraction d'erreurs de marge. Cette fraction est donnée par le nombre de points se trouvant dans la marge ou dans la mauvaise zone sur le nombre de points d'apprentissage.*
2. *ν est la borne inférieure de la fraction de vecteurs supports (nombre de vecteurs supports sur le nombre de points d'apprentissage).*
3. *Supposons que les données ont été générées indépendamment et identiquement distribuées par une distribution $P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$ telle que $P(\mathbf{x}, y = 1)$ et $P(\mathbf{x}, y = -1)$ ne contiennent aucune composante discrète, supposons de plus, que le produit scalaire utilisé soit analytique, non-constant, alors avec une probabilité 1, asymptotiquement, ν est égal à la fois à la fraction de vecteurs supports et à la fraction d'erreurs de marge.*

La preuve de cette proposition est donnée dans [SSWB00]. Ainsi, avec le paramètre ν, il est possible de contrôler le pourcentage de points pouvant être considérés comme des erreurs, lors de la phase d'apprentissage, tout en ayant une indication sur la borne inférieure du nombre de vecteurs supports.

2.3.2.4 Les SVM Probabilistes

La fonction de décision des SVM pour le classement est une fonction binaire indiquant la classe d'appartenance d'une donnée. Cette appartenance est déterminée par $(\langle \mathbf{w}, \mathbf{x} \rangle + b)$ qui indique non seulement le demi-espace dans lequel se trouve $\mathbf{x}$ mais aussi la distance qui sépare $\mathbf{x}$ de l'hyperplan séparateur. Cependant, il peut être plus avantageux d'avoir une probabilité d'appartenance à une classe. En effet, la connaissance de $P(y = 1|\mathbf{x})$ permettrait d'effectuer des traitements avec une transition douce. Cette probabilité devrait être définie de la façon suivante :

$$P(y = 1|\mathbf{x}) = \begin{cases} 1 & \text{si } \langle \mathbf{w}, \mathbf{x} \rangle + b \geq 1 \\ 0 & \text{si } \langle \mathbf{w}, \mathbf{x} \rangle + b \leq -1 \\ p_x \in]0, 1[& \text{sinon} \end{cases} \tag{2.46}$$

Pour modéliser la probabilité $P(y = 1|\mathbf{x})$ à la sortie d'un SVM, la fonction sigmoïde suivante a été proposée dans [Pla99] :

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + \exp\left(Af(\mathbf{x}) + B\right)} \tag{2.47}$$

avec $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ et (A, B) étant les paramètres de la sigmoïde.

L'apprentissage des paramètres de la sigmoïde se fait sur un ensemble d'apprentissage $\mathcal{D}'$ légèrement différent de l'ensemble d'apprentissage initial $\mathcal{D} = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_N}, y_N)\}$. En effet, nous définirons $\mathcal{D}'$ comme ceci : $\mathcal{D}' = \{(\mathbf{x_1}, t_1), \ldots, (\mathbf{x_N}, t_N)\}$, avec t_i étant la probabilité que $(\mathbf{x_i}, y_i) = (\mathbf{x_i}, 1)$. Une définition naïve de t_i consisterait à fixer $t_i = 1$ pour les instances positives ($y_i = 1$) et $t_i = 0$ pour les instances négatives. Toutefois, cette définition étant trop stricte, elle risquerait d'entraîner un sur-apprentissage. Il serait préférable d'introduire une possibilité de doute avec une valeur $\epsilon \in [0, 1]$ relativement faible telle que $t_i = 1 - \epsilon$ pour les instances positives et $t_i = \epsilon$ pour les instances négatives. En utilisant la correction de Laplace, on peut définir t_i comme ceci :

$$t_i = \begin{cases} \frac{N_+ + 1}{N_+ + 2} & \text{si } y_i = 1 \\ \frac{1}{N_- + 2} & \text{si } y_i = -1 \end{cases} \tag{2.48}$$

avec N_+ (resp. N_-) le nombre d'instance positive (resp. négative) dans l'ensemble d'apprentissage $\mathcal{D}'$.

Le problème d'apprentissage des paramètres revient à maximiser la probabilité de bon classement sur $\mathcal{D}'$, soit :

$$\max_{(A,B)} \prod_{(\mathbf{x_i}, t_i) \in \mathcal{D}'} P(y = 1|\mathbf{x_i})^{t_i} (1 - P(y = 1|\mathbf{x_i}))^{(1-t_i)} \tag{2.49}$$

En notant $p_i = P(y = 1|\mathbf{x_i})$ et en introduisant le logarithme, le problème devient :

$$\min_{(A,B)} - \sum_i (t_i \log(p_i) + (1 - t_i) \log(1 - p_i)) \tag{2.50}$$

Des algorithmes d'optimisation pour résoudre efficacement ce problème peuvent être trouvés dans [Pla99, LLW03].
Il est à signaler que l'apprentissage des paramètres devrait être effectué sur un sous-ensemble $\mathcal{D}$, qui n'aura pas été utilisé pour l'apprentissage du SVM, afin de diminuer le risque de sur-apprentissage. Il est, aussi, possible d'utiliser une validation croisée qui permettrait d'avoir un ensemble de taille convenable. Une fois, les paramètres (A, B) déterminés, l'apprentissage du SVM peut être effectué sur l'ensemble des données d'apprentissage.

2.3.2.5 **Les SVM Semi-Supervisés : l'approche transductive**

Dans la grande majorité des problèmes de la vie réelle, les données étiquetées sont rarement disponibles en quantité. En effet, l'étiquetage des données nécessite une intervention humaine pouvant être coûteuse. Lorsque les données d'apprentissage ne sont pas caractéristiques de la source alors la fonction apprise risque d'être fortement biaisée. L'idée de l'approche transductive proposée par Vapnik [Vap95] est de prendre en compte les données non-étiquetées pour induire

une fonction générale. Cette fonction devant non seulement minimiser le risque sur l'ensemble d'apprentissage mais aussi sur l'ensemble des données de test. Ce principe de minimisation du risque est connu sous le nom de *Overall Risk Minimization* [Vap82].

La figure 2.8 illustre un exemple de problème avec des données étiquetées (instances positives et négatives) et des données non-étiquetées (représentées par des croix noires). Lorsque seules les données étiquetées sont prises en compte, nous obtenons une marge contenant plusieurs points non-étiquetés. Cependant, lorsque les points non-étiquetés sont utilisés pour l'apprentissage, nous obtenons un SVM (représenté par des pointillés) de meilleur qualité.

La problématique de cette approche réside dans la manière d'intégrer l'information apportée par les données non-étiquetées dans la phase d'apprentissage. Dans cette section, nous présenterons quelques méthodes apportant une réponse à cette problématique.

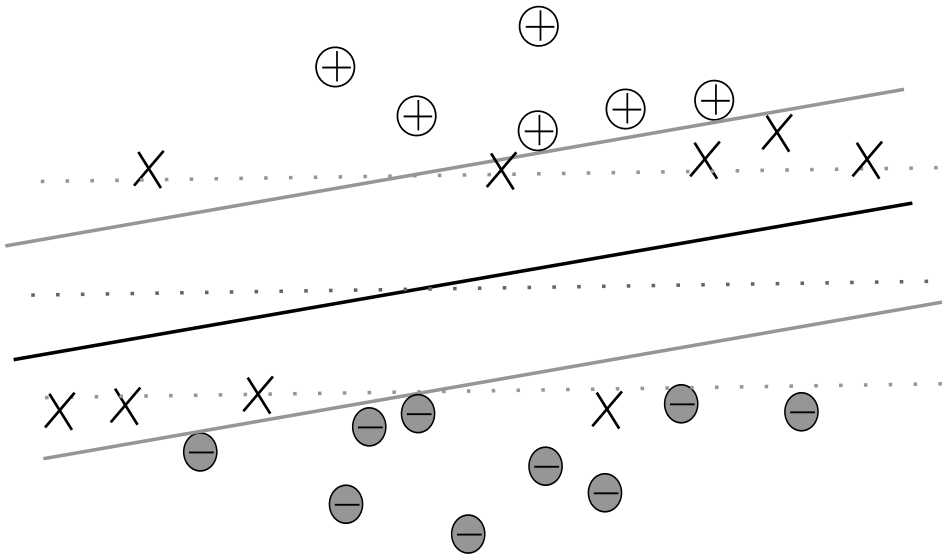


Fig. 2.8 – Illustration de l'apprentissage transductif.

Les S^{3}VMs

La première approche utilisant le principe de transduction (voir figure 2.5) proposé par Vapnik [Vap95] est le Séparateur Semi-Supervisé à Vaste Marge (S^{3}VM - *Semi-Supervised Support Vector Machine*) [BD98]. Dans cette approche, deux contraintes supplémentaires sont ajoutées au problème quadratique des SVM supervisés. Ces contraintes sont définies pour maintenir les données non-étiquetées à l'extérieur de la marge.
Pour simplifier l'écriture, nous supposerons que les l premières données sont étiquetées et les $N - l$ données suivantes non-étiquetées. En ajoutant les contraintes supplémentaires, le problème quadratique devient dans le cas non-séparable :

$$\begin{aligned}
\min_{\mathbf{w},b,\xi,\xi^{\mathbf{u}},\xi^{\mathbf{u}*}} \quad & \frac{1}{2}||\mathbf{w}||^2 + C(\sum_{i=1}^{l}\xi_i + \sum_{j=l+1}^{N}\min(\xi_j^u,\xi_j^{u*})\ ) \\
\text{s. c.} \quad & y_i(\langle \mathbf{w},\mathbf{x_i}\rangle + b) \geq 1-\xi_i\ , i\in\{1,\ldots,l\} \\
& (\langle \mathbf{w},\mathbf{x_j}\rangle + b) \geq 1-\xi_i^u\ , j\in\{l+1,\ldots,N\} \\
& (\langle \mathbf{w},\mathbf{x_j}\rangle + b) \leq -1+\xi_i^{u*} \\
& \xi_i,\xi_j^u,\xi_j^{u*}\geq 0
\end{aligned} \tag{2.51}$$

Pour faciliter la résolution de ce problème, il a été proposé, dans [BD98], de remplacer la norme 2 de $||\mathbf{w}||$ par la norme 1 ($||\mathbf{w}||_1$) afin d'obtenir un problème linéaire. En outre, une variable binaire est introduite $d_j \in \{0,1\}$ pour affecter la donnée non-étiquetée x_j à une classe. Le problème précédent devient alors :

$$\begin{aligned}
\min_{\mathbf{w},b,\xi,\xi^{\mathbf{u}},\xi^{\mathbf{u}*},\mathbf{d}} \quad & ||\mathbf{w}||_1 + C(\sum_{i=1}^{l}\xi_i + \sum_{j=l+1}^{N}(\xi_j^u+\xi_j^{u*})\ ) \\
\text{s. c.} \quad & y_i(\langle \mathbf{w},\mathbf{x_i}\rangle + b) \geq 1-\xi_i\ , i\in\{1,\ldots,l\} \\
& (\langle \mathbf{w},\mathbf{x_j}\rangle + b) + L.(1-d_j) \geq 1-\xi_i^u\ , j\in\{l+1,\ldots,N\} \\
& (\langle \mathbf{w},\mathbf{x_j}\rangle + b) - L.d_j \leq -1+\xi_i^{u*} \\
& \xi_i,\xi_j^u,\xi_j^{u*}\geq 0
\end{aligned} \tag{2.52}$$

La constante L est choisi suffisamment grande pour que si $d_j = 1$ alors $\xi_i^{u*} = 0$ et que si $d_j = 0$ alors $\xi_i^u = 0$. Ainsi, l'une des deux contraintes sera neutralisée selon la région à laquelle la donnée sera affectée. Le problème peut ainsi être résolu par des solveurs linéaires classiques.

Les expériences dans [BD98] ont montré que le S^{3}VM a un niveau de généralisation tout aussi bon, voire meilleur, que le SVM classique.

Les T-SVM

L'approche la plus utilisée pour la transduction consiste à induire une fonction locale à l'aide des données d'apprentissage. La fonction est ensuite utilisée pour étiqueter les données de test. L'union des ensembles d'apprentissage et de test est, ensuite, utilisée pour induire une fonction globale. La première méthode utilisant cette approche est le SVM transductif (T-SVM) [Joa99a, Joa02]. Le T-SVM a été initialement utilisé pour la catégorisation de documents textuels. Dans ce type de problème, l'étiquetage étant un processus coûteux, le nombre de documents étiquetés pouvant servir à l'apprentissage est extrêmement faible comparé au nombre de documents non traités.

En reprenant les notations de la section précédente, à savoir en supposant que les l premières données sont étiquetées et que les $N-l$ données suivantes ne sont pas étiquetées, nous obtenons le problème quadratique suivant :

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\xi^*,\mathbf{y}^*} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{l}\xi_i + C^*\sum_{j=l+1}^{N}\xi_j^* \\ \text{s. c.} \quad & y_i(\langle \mathbf{w}, \mathbf{x_i}\rangle + b) \geq 1 - \xi_i \ , i \in \{1,\ldots,l\} \\ & y_j^*(\langle \mathbf{w}, \mathbf{x_j}\rangle + b) \geq 1 - \xi_i^* \ , j \in \{l+1,\ldots,N\} \\ & \xi_i, \xi_j^* \geq 0 \end{aligned} \tag{2.53}$$

Pour résoudre ce problème, les différentes combinaisons devront être testées. Nous noterons l'analogie avec les variables de décision d_j du S^{3}VM dans le problème linéaire 2.52.
Afin de réduire la complexité de calcul, l'approche adoptée dans [Joa99a, Joa02] est d'induire une fonction locale et d'étiqueter les données $(\mathbf{x_j}, y_j^*)$. Pour cela, le paramètre C^* de la fonction objectif est utilisé. En le fixant à 0, le processus d'apprentissage est purement inductif et en le fixant à 1, le processus devient entièrement transductif. L'algorithme T-SVM est un algorithme itératif qui consiste, à chaque itération, à minimiser la fonction objectif puis à réapprendre un SVM en augmentant C^*. Lors d'une itération, la fonction objectif est minimisée en cherchant un couple de données non étiquetées (x_m, x_l), avec $y_m^* \neq y_l^*$, se situant dans la marge ou dans la mauvaise zone tel que $\xi_m^* + \xi_l^* > 2$. Les étiquettes du couple sont permutées pour replacer les données dans une zone plus plausible, minimisant $(\xi_m^* + \xi_l^*)$. Si un couple d'étiquettes a été permuté, un nouveau SVM est appris et la procédure est répétée. Si, par contre, aucun couple n'a été trouvé, $C*$ est augmenté et l'algorithme passe à la prochaine itération. Les itérations s'arrêtent lorsque C^* a atteint un seuil fixé a priori. L'algorithme 2.3.1 résume les principales étapes du T-SVM.

Algorithme 2.3.1 Algorithme de T-SVM

1: Apprendre un SVM sur les données d'apprentissage
2: Étiqueter les données de test avec le SVM appris
3: **pour** C^* allant de C^*_{min} à C^*_{max} en multipliant C^* par 2 **faire**
4: Réapprendre le SVM selon le problème quadratique 2.53
5: **tant que** $\exists(x_m, x_l) : y_l^* \neq y_m^*$ et $\min(\xi_m^*, \xi_l^*) > 0$ et $(\xi_m^* + \xi_l^* > 2)$ **faire**
6: Permuter les valeurs de y_m et y_l
7: Réapprendre le SVM (problème 2.53)
8: **fin tant que**
9: **fin pour**

Il est, en outre, possible de décomposer le terme $C^* \sum_{j=l+1}^{N} \xi_j^*$ en :

$$C_+^* \sum_{j:y_j^*=1} \xi_j^* \quad \text{et} \quad C_-^* \sum_{j:y_j^*=-1} \xi_j^*$$

dans la fonction objective du problème 2.53. Cette décomposition permet de définir des pénalisations prenant en compte la proportion d'instances positives et d'instances négatives. Les

instances positives étant généralement largement minoritaires, une pénalisation différente permet de diminuer l'influence des instances négatives sur la marge.

Les expériences menées dans [Joa99a, Joa02] sur des corpus de documents ont montré que le T-SVM est très largement plus performant que le SVM classique. Cette différence se creuse lorsque les données étiquetées ne sont disponible qu'en très faible quantité.

2.3.2.6 les SVM Multi-catégories

Les Séparateurs à vaste marge ont été développés pour traiter des problèmes binaires. Toutefois, les bases de données réels soulèvent des problèmes avec plusieurs classes. Nous verrons quelques méthodes permettant de traiter ces problèmes.

Stratégies standards

Pour traiter ces problèmes multi-classes, les deux stratégies les plus utilisées sont :

- **un-contre-tous** ou encore appelée **un-contre-le-reste** : dans cette stratégie, le problème multi-classes, avec k classes, est décomposé en k sous-problèmes binaires. Ainsi, un sous-problème pour la classe c, peut être ramené à un problème binaire avec les données de la classe c comme instances positives et les autres données comme instances négatives. Pour chaque problème, un hyperplan séparateur, $(\mathbf{w^i}, b^i)$, est "appris". Le classement d'une nouvelle donnée $\mathbf{x}$ est donné par :

$$y = \text{argmax}_i(\langle \mathbf{w^i}, \mathbf{x} \rangle + b^i) \tag{2.54}$$

  Pour le multi-étiquetage, à savoir les problèmes où une donnée peut avoir plusieurs classes, on affecte à la donnée x toutes les classes correspondantes à $(\langle \mathbf{w^i}, \mathbf{x} \rangle + b^i) \geq 0$.
  Le problème majeur avec cette stratégie est que, pour chaque sous-problème, les instances négatives sont largement plus nombreuses que les instances positives. Cette inégalité peut affecter l'apprentissage de l'hyperplan séparateur.

- **un-contre-un** : dans cette stratégie, le problème de k classes est décomposé en $\frac{k(k-1)}{2}$ sous-problèmes binaires. Chaque problème (i, j) consiste à déterminer un SVM séparant les instances de la classe i des instances de la classe j. Le classement s'effectue ensuite selon le schéma du vote majoritaire. Pour classer une donnée x, il faut ainsi traiter les $\frac{k(k-1)}{2}$ sous-problèmes et affecter à x, la classe à qui aura été affectée x le plus de fois lors du traitement des sous-problèmes. Le multi-étiquetage est traité, en général, en définissant un seuil.
  Une alternative au vote majoritaire est l'utilisation d'un graphe acyclique orienté de décision (DDAG) [PCST00]. La figure 2.9 illustre un exemple de DDAG pour 4 classes. Chaque noeud N_{ij} de ce graphe est associé à un SVM séparant les instances de la classe i des instances de la classe j. Pour l'affectation d'une donnée à une classe, nous procédons

par le noeud racine jusqu'à descendre à une feuille représentant la classe finale. Lors de l'évaluation d'un noeud N_{ij}, la classe qui n'aura pas été sélectionnée sera complètement éliminée de l'espace de solution pour la donnée. Ainsi, à chaque niveau, une classe est éliminée de l'espace de solution. L'ordre dans lequel les noeuds sont placés n'est pas pertinent pour le problème tant que le graphe final reste un DDAG.
Dans [BB06], les auteurs proposent une méthode de décomposition dont la structure classificatoire ne s'appuie pas sur un DDAG mais sur un dendogramme : D-SVM. Le dendogramme est construit en calculant les centres des sous-ensembles de l'ensemble d'apprentissage associés aux différentes classes et en appliquant à ces centres une classification ascendante hiérarchique (CAH). Le classifieur global s'obtient alors en plaçant sur les noeuds du dendogramme des SVM bi-classes. La figure 2.10 illustre un exemple de dendogramme de décision pour un problème à 6 classes.

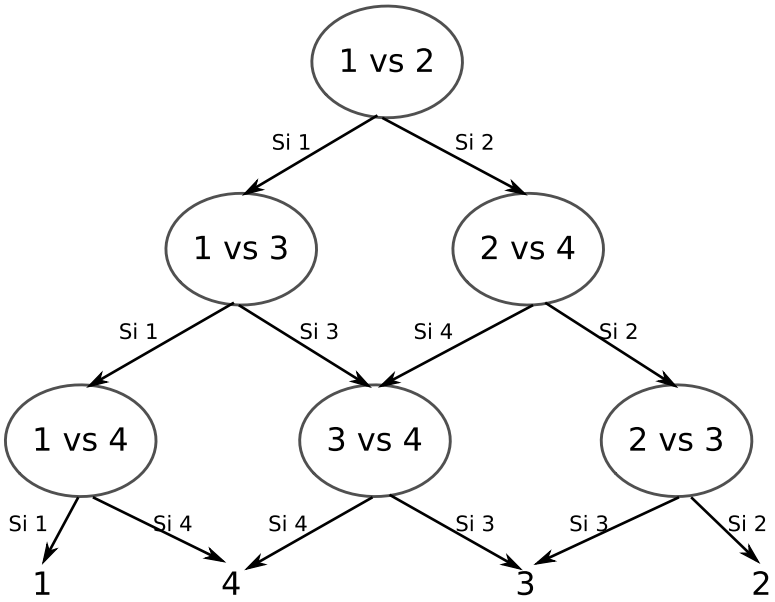


Fig. 2.9 – Graphe orienté acyclique de décision pour 4 classes en utilisant la stratégie un-contre-un.

La stratégie la plus employée est la stratégie "un-contre-tous". En outre, nous avons utilisé cette stratégie pour traiter les problèmes multi-classes dans nos travaux.

Les M-SVM

La décomposition en sous-problème affecte l'apprentissage de l'hyperplan. En effet, les SVM permettent de déterminer l'hyperplan optimal pour un problème binaire mais cette optimalité n'est pas garantie pour les problèmes multi-classes. Pour répondre, à ce problème les M-SVM ont été proposés [WW98, BB99, GEPM00].

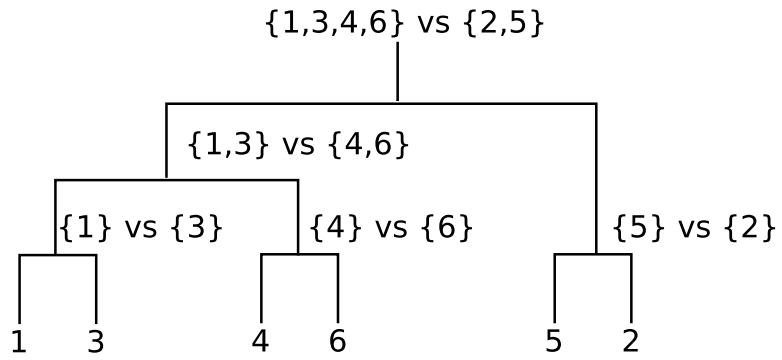


Fig. 2.10 – Exemple de dendogramme de décision construit par le D-SVM pour un problème à 6 classes.

Le M-SVM intègre les contraintes d'appartenance des données aux différentes classes directement dans le problème d'optimisation quadratique [BB99]. L'objectif est d'obtenir des hyperplans séparateurs optimaux. Pour cela, nous introduisons la définition de la séparabilité linéaire par morceau :

Définition 2.3.3. *Un ensemble de données,* $\mathcal{D} = \cup_{i=1}^{k} \mathcal{D}_k$ *avec* $\mathcal{D}_j = \{(\mathbf{x}_1^j, y_1^j), \ldots, (\mathbf{x}_{N_j}^j, y_{N_j}^j)\}$ *et* $y_i^j = j$*, est dit linéairement séparable par morceau s'il existe* $(\mathbf{w^i}, b^i)$ *pour* $i \in \{1, \ldots, k\}$ *tel que :*

$$\forall i, j \in \{1, \ldots, k\}, i \neq j, \ \forall \mathbf{x^i} \in \mathcal{D}_i, \ (\langle \mathbf{w^i}, \mathbf{x^i} \rangle + b^i) > (\langle \mathbf{w^j}, \mathbf{x^i} \rangle + b^j) \tag{2.55}$$

La définition 2.3.3 implique qu'il n'existe aucun chevauchement entre les différentes classes. Nous avons ainsi pour tout i, j $(i \neq j)$ et pour tout $\mathbf{x^i} \in \mathcal{D}_i$:

$$\begin{aligned} \langle \mathbf{w^i}, \mathbf{x^i} \rangle + b^i) - ((\langle \mathbf{w^j}, \mathbf{x^i} \rangle + b^j)) &> 0 \\ (\mathbf{w^i} - \mathbf{w^j})\mathbf{x^i} + (b^i - b^j) &> 0 \end{aligned} \tag{2.56}$$

En se limitant aux hyperplans canoniques, nous obtenons :

$$(\mathbf{w^i} - \mathbf{w^j})\mathbf{x^i} + (b^i - b^j) \geq 1 \tag{2.57}$$

À l'instar du problème du SVM classique, le problème d'optimisation dans le primal est donné par :

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{1}{2}||\mathbf{w^i}-\mathbf{w^j}||^2+\sum_{i=1}^{k}\frac{1}{2}||\mathbf{w^i}||^2 \\ \text{s. c.} \quad & (\mathbf{w^i}-\mathbf{w^j})\mathbf{x_l^i}+(b^i-b^j)\geq 1 \\ & l\in\{1,\ldots,N_i\} \\ & i,j\in\{1,\ldots,k\}, i\neq j \end{aligned} \tag{2.58}$$

Dans [WW98, GEPM00], seul le terme $\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{1}{2}||\mathbf{w^i}-\mathbf{w^j}||^2$ est utilisé dans la fonction objectif ci-dessus.
Le problème peut être résolu dans le dual en introduisant les multiplicateurs de Lagrange α_l^{ij} associé à chaque contrainte : $(\mathbf{w^i}-\mathbf{w^j})\mathbf{x_l^i}+(b^i-b^j)\geq 1$. La solution de l'hyperplan pour la classe c est donnée par :

$$\mathbf{w^c}=\sum_{i=1,i\neq c}^{k}\sum_{l=1}^{n_k}\alpha_l^{ci}\mathbf{x_l^c}-\sum_{i=1,i\neq c}^{k}\sum_{l=1}^{n_k}\alpha_l^{ic}\mathbf{x_l^i} \tag{2.59}$$

La fonction de classement (d'affectation) d'une donnée $\mathbf{x}$ à une classe y est :

$$y=f(\mathbf{x})=\text{argmax}_i\langle\mathbf{w^i},\mathbf{x}\rangle+b^i \tag{2.60}$$

Les M-SVM peuvent aussi être utilisés dans le cas où les données ne sont pas linéairement séparables. Il faudra, pour cela, autoriser une certaine quantité d'erreur en introduisant les variables d'écart ξ_i [WW98].

Les K-SVCRs

Une autre approche, appelée K-SVCR, pour traiter les problèmes multi-classes est présentée dans [AC00, APC03]. L'idée est d'utiliser un hyperplan séparateur pour diviser l'espace en trois régions. Une région doit contenir les instances d'une classe c_1, une seconde région doit contenir les instances d'une classe c_2 et la troisième région comprise entre les deux premières doit contenir les instances des autres classes. La figure 2.11 illustre cette idée. L'hyperplan séparateur étant donné par l'équation $\langle\mathbf{w},\mathbf{x}\rangle+b=0$, la fonction de décision pour deux classes c_1 et c_2 est alors :

$$f_{c_1,c_2}(\mathbf{x})=\begin{cases} +1 & \text{si } \langle\mathbf{w_{c_1,c_2}},\mathbf{x}\rangle+b_{c_1,c_2}\geq\epsilon \\ -1 & \text{si } \langle\mathbf{w_{c_1,c_2}},\mathbf{x}\rangle+b_{c_1,c_2}\leq\epsilon \\ 0 & \text{sinon} \end{cases} \tag{2.61}$$

avec $\epsilon\in[0,1[$ une constante définie a priori contrôlant la région des instances autres celle de c_1 et de c_2.

Il est à noter que la séparation en trois régions ainsi décrites est rarement possible dans l'espace d'origine. Toutefois, l'utilisation des noyaux permet d'utiliser les SVM dans un espace de description dans lequel cette séparation est possible. Les noyaux seront présentés plus tard dans la section 2.3.6.

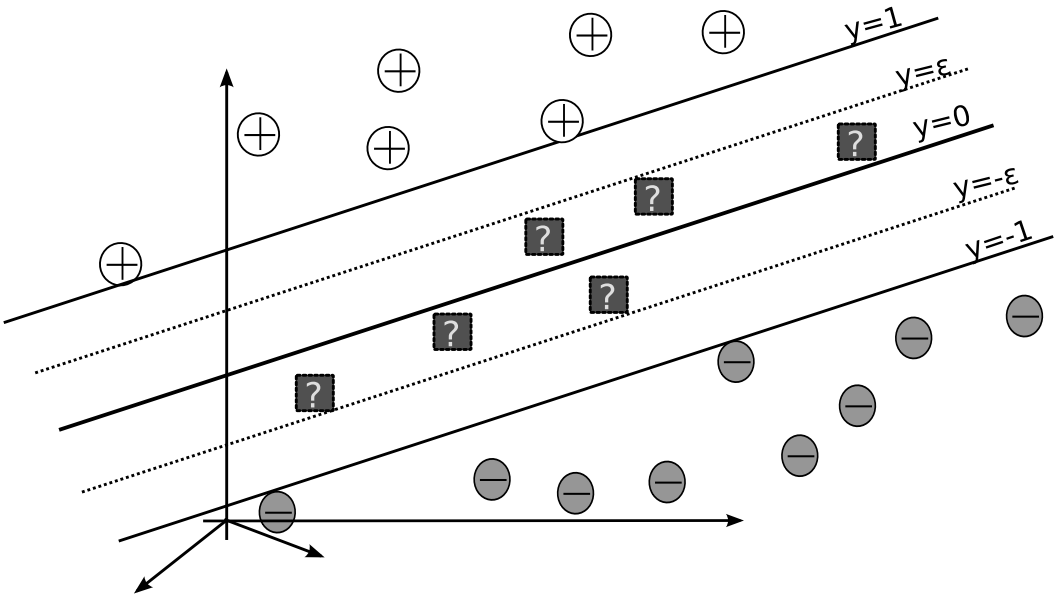


Fig. 2.11 – Séparation de trois classes par un hyperplan séparateur.

Le problème du K-SVCR peut être formalisé ainsi, étant donné un ensemble de données, $\mathcal{D} = \cup_{i=1}^{k} \mathcal{D}_k$ avec $\mathcal{D}_j = \{(\mathbf{x}_1^j, y_1^j), \ldots, (\mathbf{x}_{n_j}^j, y_{n_j}^j)\}$ et $y_i^j = j$, une constante ϵ, le séparateur à vaste marge pour les classes i et j, paramétré par $(\mathbf{w_{i,j}}, b_{i,j})$ doit vérifier les propriétés suivantes :

$$\begin{cases} \langle \mathbf{w_{i,j}}, \mathbf{x} \rangle + b_{i,j} \geq 1 & \forall \mathbf{x} \in \mathcal{D}_i \\ \langle \mathbf{w_{i,j}}, \mathbf{x} \rangle + b_{i,j} \leq -1 & \forall \mathbf{x} \in \mathcal{D}_j \\ |\langle \mathbf{w_{i,j}}, \mathbf{x} \rangle + b_{i,j}| \leq \epsilon & \forall \mathbf{x} \in \mathcal{D} - (\mathcal{D}_i \cup \mathcal{D}_j) \end{cases} \tag{2.62}$$

Pour simplifier l'écriture, nous fixerons $y_i = 1$ pour tout x_i appartenant à la classe i et $y_j = -1$ pour tout x_j appartenant à la classe j. En outre, pour le problème de séparation des classes i et j, nous simplifierons l'écriture de $(\mathbf{w_{i,j}}, b_{i,j})$ par $(\mathbf{w}, b)$. De plus, pour un ensemble contenant N données, nous considérons que les N_1 premières données sont des instances positives $(y = 1)$, les N_2 suivantes sont des instances négatives $(y = -1)$ et le reste étant des instances d'autres classes. Nous définirons $N_{12} = N_1 + N_2$. En autorisant les erreurs pour le cas non linéairement séparable, nous introduirons les variables d'écart ξ_i contrôlées par la constante C et (φ_i, φ_i^*) contrôlées par la constante D. Le problème du K-SVCR peut être formalisé par le problème quadratique dans le primal suivant :

$$\min_{\mathbf{w},b,\xi,\varphi,\varphi^*\rho} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i + D\sum_{i=1}^{N}(\varphi_i + \varphi_i^*) \tag{2.63}$$

$$\begin{aligned} \text{s. c.} \quad & y_i(\langle \mathbf{w}, \mathbf{x_i}\rangle + b) \geq 1 - \xi_i,\ \forall i \in \{1,\ldots,N_{12}\} \\ & \langle \mathbf{w}, \mathbf{x_i}\rangle + b \leq \epsilon + \varphi_i,\ \forall i \in \{N_{12}+1,\ldots,N\} \\ & \langle \mathbf{w}, \mathbf{x_i}\rangle + b \geq -\epsilon - \varphi_i^*,\ \forall i \in \{N_{12}+1,\ldots,N\} \\ & \xi_i, \varphi_i, \varphi_i^* \geq 0 \end{aligned} \tag{2.64}$$

Nous pouvons remarquer que le K-SVCR se comporte à la fois comme le C-SVM pour le classement des instances positives et négatives, et comme le SVR, défini plus bas à la section 2.3.5, pour les instances des autres classes. En effet, la région contenant les instances des autres classes est définie comme étant un "tube" ϵ-insensible autour du séparateur optimal (voir la section 2.3.5 pour plus de détail). La solution de ce problème est obtenue par une re-formulation dans le dual. Les détails sont décrits dans [AC00, APC03].
En outre, une re-paramétrisation de K-SVCR en introduisant la constante ν est proposée pour obtenir ν-K-SVCR dans [ZF06].

Le classement d'une nouvelle donnée s'effectue en évaluant les fonctions de décisions $f_{i,j}$ sur x tant que $f_{i,j}(\mathbf{x}) = 0$. On affecte alors à $\mathbf{x}$ la classe i si $f_{i,j}(\mathbf{x}) = 1$ ou la classe j si $f_{i,j}(\mathbf{x}) = -1$.

2.3.3 Les SVM pour l'estimation de densité

Bien qu'initialement, les Séparateurs à Vaste Marge aient été introduits pour le problème de classement binaire, ils peuvent aussi être utilisés pour l'estimation de densité dans des petites régions d'espace de haute dimension [SPST+99, SPST+01]. Ces SVM sont souvent appelés SVM à une classe. Cependant, l'information concernant la classe n'est pas utilisée. Le problème de l'estimation de densité peut être défini ainsi : étant donné un ensemble $\mathcal{D}$ de points générés par une source selon une distribution $P(x)$ inconnue, nous souhaitons estimer P par $\hat{P}$ tel que $\hat{P}(x) = 1$ si x appartient à une région de forte densité et $\hat{P}(x) = 0$ dans le cas contraire.

2.3.3.1 Estimation de la densité par un hyperplan

La région de forte densité est définie, dans le cas d'une estimation par des SVM, comme étant le sous-espace délimité par l'hyperplan H_+ parallèle à l'hyperplan "séparateur" H et tel que la distance entre H_+ et H soit maximale (maximisation de la marge). Ainsi, la zone de forte probabilité sera définie par :

$$\langle \mathbf{w}, \mathbf{x}\rangle \geq \rho \tag{2.65}$$

avec $(\mathbf{w}, \rho)$ les paramètres de l'hyperplan H_+. L'hyperplan séparateur est donné par $\langle \mathbf{w}, \mathbf{x}\rangle = 0$
L'appartenance de x à la région de forte densité sera définie par la fonction de décision :

$$f(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \mathbf{x}\rangle - \rho) \tag{2.66}$$

La figure 2.12 donne une illustration de l'estimation de densité par un hyperplan.

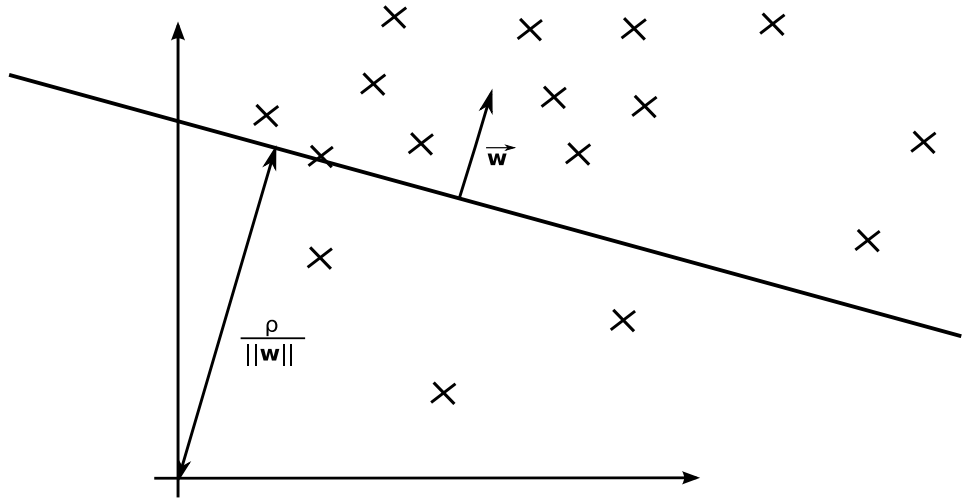


Fig. 2.12 – Estimation de densité par un hyperplan séparateur.

En autorisant un certain nombre d'erreur, par l'introduction des variables d'écart ξ_i, tout en contrôlant ces erreurs par la constante $\nu \in]0,1]$ (la section 2.3.2.3 concernant les ν-SVM décrit le rôle de ν), nous obtenons le problème quadratique primal suivant :

$$\begin{aligned} \min_{\mathbf{w},\rho,\xi} \quad & \frac{1}{2}||\mathbf{w}||^2 + \frac{1}{\nu N}\sum_{i=1}^{N}\xi_i - \rho \\ \text{s. c.} \quad & \langle \mathbf{w}, \mathbf{x_i} \rangle \geq \rho - \xi_i \\ & \xi_i \geq 0 \\ & i \in \{1, \ldots, N\} \end{aligned} \tag{2.67}$$

La formulation dans le dual est :

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1,j=1}^{N}\alpha_i\alpha_j\langle \mathbf{x_i}, \mathbf{x_j} \rangle \\ \text{s. c.} \quad & 0 \leq \alpha_i \leq \frac{1}{\nu N}, \quad i \in \{1, \ldots, N\} \\ & \sum_i \alpha_i = 1 \end{aligned} \tag{2.68}$$

La solution de ce problème est :

$$\mathbf{w} = \sum_{i=1}\alpha_i\mathbf{x_i} \tag{2.69}$$

$$\rho = \frac{1}{|S|}\sum_{\mathbf{x}\in S}\langle \mathbf{w}, \mathbf{x} \rangle \tag{2.70}$$

avec S un ensemble non vide de vecteurs supports $\mathbf{x_i}$ tel que $0 < \alpha_i < \frac{1}{\nu N}$.

2.3.3.2 **Estimation de la densité par une hypersphère**

Dans ce cas, la région de forte densité est délimitée par une hypersphère de rayon R et de centre c [SPST$^+$99, SPST$^+$01, SBV95, TD99b, TD99a, TD04]. L'application de la SRM consiste à minimiser le rayon R de l'hypersphère englobant la majorité des données. L'hypersphère est déterminée par les vecteurs supports minimisant R. La région de forte densité est définie par :

$$||\mathbf{x} - \mathbf{c}||^2 \leq R^2 \tag{2.71}$$

La fonction de décision devient alors :

$$f(\mathbf{x}) = \text{sign}(R^2 - ||\mathbf{x} - \mathbf{c}||^2) \tag{2.72}$$

On notera la forme développée suivante :

$$f(\mathbf{x}) = \text{sign}(R^2 - \langle \mathbf{c}, \mathbf{c} \rangle + 2\langle \mathbf{c}, \mathbf{x} \rangle - \langle \mathbf{x}, \mathbf{x} \rangle) \tag{2.73}$$

La figure 2.13 illustre le principe de l'estimation par une hypersphère.

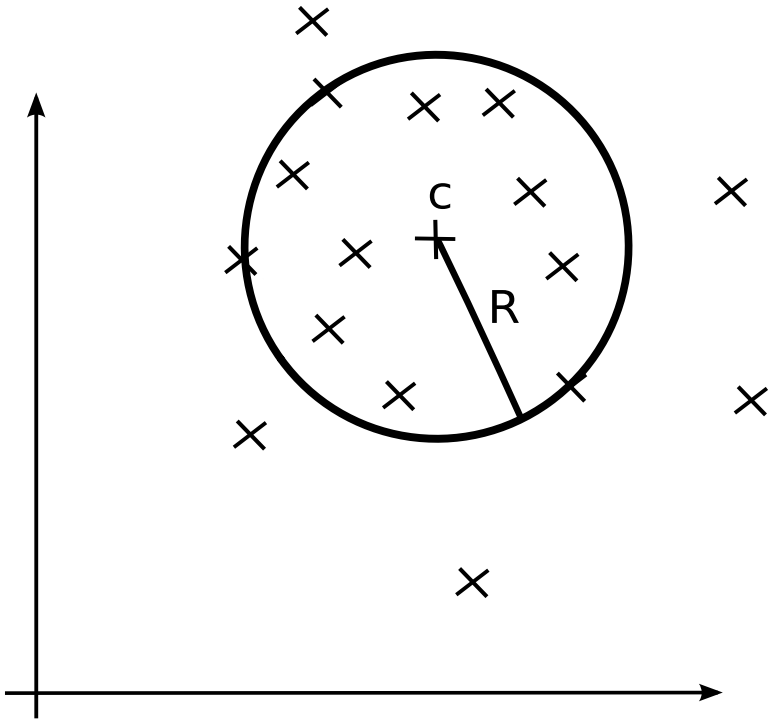


Fig. 2.13 – Estimation de densité avec une hypersphère.

De même que dans le cas de l'estimation de densité par un hyperplan, nous introduisons les variables d'écart pour permettre une certaine quantité d'erreur et le paramètre ν pour contrôler ces erreurs. Nous obtenons le problème quadratique primal suivant :

$$\begin{aligned} \min_{\mathbf{c},R,\xi} \quad & R^2 + \frac{1}{\nu N}\sum_{i=1}^{N}\xi_i \\ \text{s. c.} \quad & ||\mathbf{x_i} - \mathbf{c}||^2 \leq R^2 + \xi_i \\ & \xi_i \geq 0 \\ & i \in \{1, \ldots, N\} \end{aligned} \tag{2.74}$$

La formulation dans le dual est :

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i=1,j=1}^{N} \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle - \sum_{i=1}^{N} \alpha_i \langle \mathbf{x_i}, \mathbf{x_i} \rangle \\ \text{s. c.} \quad & 0 \leq \alpha_i \leq \frac{1}{\nu N}, \quad i \in \{1, \ldots, N\} \\ & \sum_i \alpha_i = 1 \end{aligned} \tag{2.75}$$

La solution de ce problème est :

$$\mathbf{c} = \sum_{i=1} \alpha_i \mathbf{x_i} \tag{2.76}$$

$$R = \sqrt{\frac{1}{|S|}\sum_{\mathbf{x}\in S} ||\mathbf{x} - \mathbf{c}||^2} \tag{2.77}$$

avec S un ensemble non vide de vecteurs supports $\mathbf{x_i}$ tel que $0 < \alpha_i < \frac{1}{\nu N}$.

2.3.4 Les SVM pour la classification (*clustering*)

L'objectif de la classification est de regrouper les données en plusieurs classes selon des critères de distances (ou de similarités). Ainsi, il s'agit d'une méthode non supervisée dans le sens où l'information sur les étiquettes n'est pas utilisée. Elle s'apparente ainsi à la méthode des SVM à une classe vue dans la section 2.3.3. La méthode de classification proposée dans [BHHSV01a, BHHSV01b] est relativement simple. Elle se base sur les SVM à une classe. L'idée est de trouver la région de haute densité à l'aide d'une hypersphère telle que décrite dans la section 2.3.3.2. Dans le cas de la méthode standard vue précédemment, seule une seule hypersphère serait découverte, comme illustrée dans la figure 2.14.

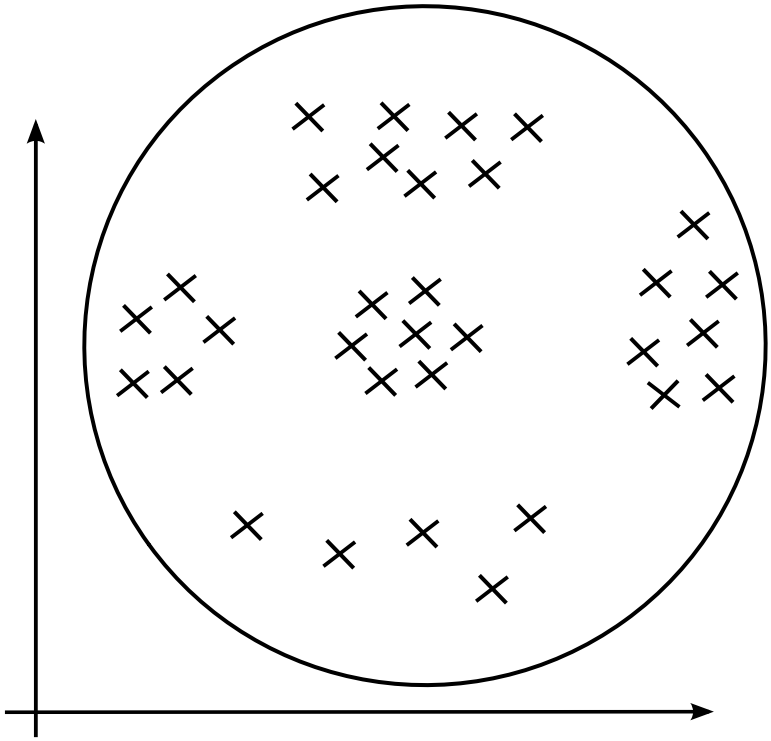


Fig. 2.14 – Estimation de densité avec une hypersphère et un produit scalaire dans l'espace d'origine.

Afin d'avoir, une estimation plus fine de la densité, il est proposé, dans [BHHSV01a] et [BHHSV01b], d'utiliser un noyau gaussien.
Nous étudierons plus tard les noyaux. Toutefois, pour les besoins de cette méthode, nous définirons, un noyau $k(\mathbf{x}, \mathbf{y})$, comme étant un produit scalaire dans un espace de Hilbert appelé espace de description (ou encore espace d'attributs). Le noyau permet, implicitement, de transformer un vecteur $\mathbf{x}$ (resp. $\mathbf{y}$) en son vecteur image $\Phi(\mathbf{x})$ (resp. $\Phi(\mathbf{y})$) dans l'espace de description. Le noyau est alors exprimé par $k(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$. Dans le cas du noyau gaussien, on a : $k(\mathbf{x}, \mathbf{y}) = \exp(-\frac{||\mathbf{x}-\mathbf{y}||^2}{2\sigma^2})$. En remplaçant dans les algorithmes SVM précédents, le produit scalaire standard (dans l'espace des données) $\langle ., . \rangle$ par le noyau gaussien, on obtient alors des algorithmes linéaires dans l'espace de description mais non linéaires dans l'espace des données. Avec le noyau gaussien, l'estimation de la densité devient plus fine dans l'espace des données comme illustrée dans la figure 2.15. Nous obtenons, ainsi, plusieurs zones de forte densité. Il est à noter que dans l'espace de description seule une hypersphère, de rayon R et de centre $\Phi(c)$, existe. En effet, pour tout point x appartenant à une zone de densité, nous avons :

$$d_{\mathcal{H}}(x, c) = ||\Phi(\mathbf{x}) - \Phi(\mathbf{c})||^2 \leq R^2 \tag{2.78}$$

La distance peut être exprimée uniquement avec le noyau k :

$$\begin{aligned} d_{\mathcal{H}}(x,c) &= ||\Phi(\mathbf{x}) - \Phi(\mathbf{c})||^2 \\ &= \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}) \rangle + \langle \Phi(\mathbf{c}), \Phi(\mathbf{c}) \rangle - 2\langle \Phi(\mathbf{x}), \Phi(\mathbf{c}) \rangle \\ &= k(\mathbf{x},\mathbf{x}) + k(\mathbf{c},\mathbf{c}) - 2k(\mathbf{x},\mathbf{c}) \end{aligned} \tag{2.79}$$

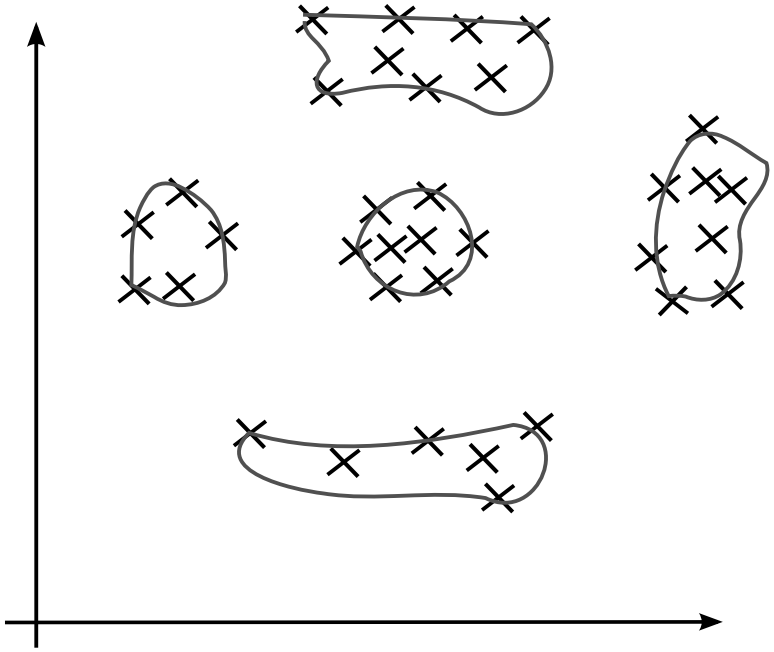


Fig. 2.15 – Estimation de densité avec une hypersphère et un produit scalaire dans un espace de description.

L'idée développée dans [BHHSV01a, BHHSV01b] se base sur l'observation qu'étant donné deux points x et y, ces points appartiennent à deux classes différentes si aucun des chemins connectant x et y n'est contenu dans l'hypersphère de rayon R résidant dans l'espace de description. Plus formellement si $p_{x,y}$ est un chemin connectant x et y, dans l'espace de description, x et y appartiennent à deux classes différentes si et seulement si :

$$\forall p_{x,y}, \exists z \in p_{x,y}, d_{\mathcal{H}}(z,c) > R \tag{2.80}$$

Par conséquent, s'il existe un chemin allant de x à y et contenu dans l'hypersphère de rayon R, alors x et y appartiennent à la même classe. On définit alors la matrice d'adjacence A tel que :

$$A_{ij} = \begin{cases} 1 & \text{si } \forall z \text{ appartenant au segment de droite reliant } x_i \text{ à } x_j \ d_{\mathcal{H}}(z,c) \leq R \\ 0 & \text{sinon} \end{cases} \tag{2.81}$$

Le graphe induit par la matrice d'adjacence A définit les classes. En effet, tous les points connectés dans le graphe appartiennent à une même classe. En outre, les algorithmes [YECC02, Lee05] apportent quelques modifications permettant d'améliorer les performances.

2.3.5 Les SVM pour la régression (SVR)

Pour le cas de la régression linéaire, la variable à prédire y_i est un nombre réel. L'utilisation des SVM pour la régression consiste à utiliser l'hyperplan séparateur optimal pour prédire y[DBK+97, VGS96], soit :

$$y = f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b \tag{2.82}$$

Pour traiter le cas où les données ne sont pas parfaitement linéaires, une marge d'erreur de plus ou moins ϵ est autorisée pour la prédiction. La figure 2.16 illustre le principe.

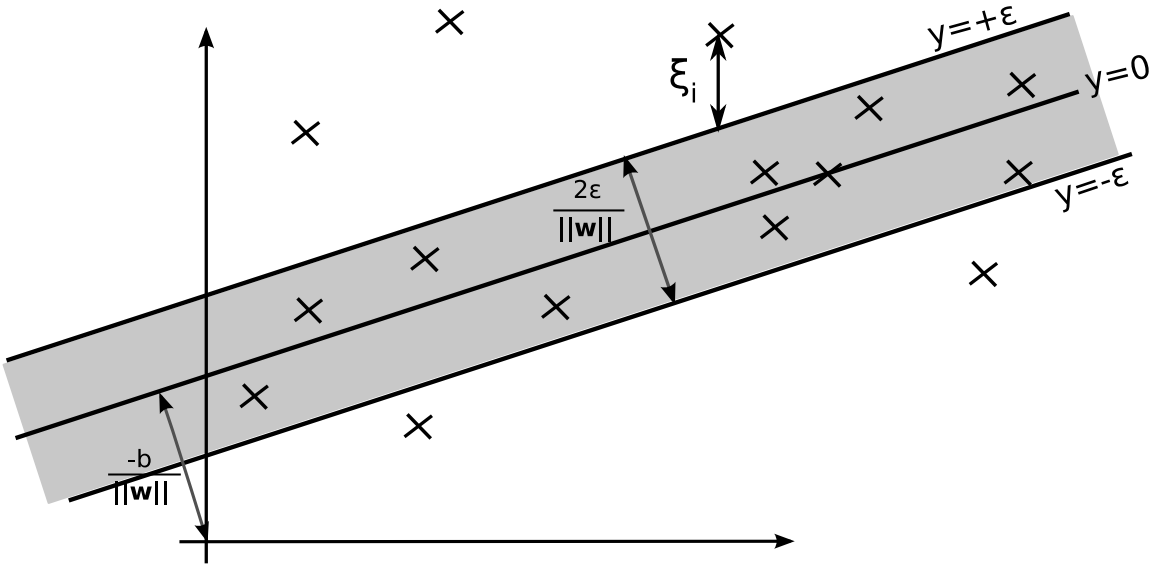


Fig. 2.16 – Régression SVR ϵ-insensible.

Ainsi, la fonction de coût pour une mauvaise prédiction dans le cas d'une zone de tolérance de $\pm\epsilon$ (ϵ-insensible) sera :

$$\mathcal{L}_\epsilon(y_i, f(x_i)) = \begin{cases} 0 & \text{si } |f(\mathbf{x_i}) - y_i| \leq \epsilon \\ |f(\mathbf{x_i}) - y_i| - \epsilon & \text{sinon} \end{cases} \tag{2.83}$$

Le problème des SVR consiste à minimiser :

$$R_{emp}(f_{\mathbf{w},b}) + ||\mathbf{w}|| = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_\epsilon(y_i, f_{\mathbf{w},b}(x_i)) + ||\mathbf{w}|| \tag{2.84}$$

Le problème C-SVR peut être formulé dans le primal comme :

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\xi^*} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \\ \text{s. c.} \quad & y_i - (\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \leq \epsilon + \xi_i \\ & (\langle \mathbf{w}, \mathbf{x_i} \rangle + b) - y_i \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0 \\ & i \in \{1, \ldots, N\} \end{aligned} \tag{2.85}$$

Le paramètre de régularisation C est introduit pour contrôler le niveau d'oscillation de la fonction.

La formulation duale de ce problème devient alors :

$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & \frac{1}{2}\sum_{i,j}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle \mathbf{x_i}, \mathbf{x_j} \rangle + \epsilon\sum_i(\alpha_i + \alpha_i^*) - \sum_i y_i(\alpha_i - \alpha_i^*) \\ \text{s. c.} \quad & \sum_i y_i(\alpha_i - \alpha_i^*) = 0 \\ & \forall i \in \{1, \ldots, N\}, \quad 0 \leq \alpha_i, \alpha_i^* \leq C \end{aligned} \tag{2.86}$$

la solution du problème dual est donnée par :

$$\mathbf{w} = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\mathbf{x_i} \tag{2.87}$$

$$b = \frac{1}{|\mathcal{U}|}\sum_{(\mathbf{x},y)\in\mathcal{U}} y - \langle \mathbf{w}, \mathbf{x} \rangle - \text{sign}(\alpha_i - \alpha_i^*)\epsilon \tag{2.88}$$

avec $\mathcal{U}$ l'ensemble des vecteurs supports tels que $0 < \alpha_i < C$ et $\alpha_i^* = 0$ ou $\alpha_i = 0$ et $0 < \alpha_i^* < C$.

Nous avons vu dans la section 2.3.2.3 qu'il pouvait être pratique d'introduire une constante ν pour contrôler l'apprentissage. ν doit ainsi définir une borne supérieure sur la fraction d'erreur et une borne inférieure sur la fraction de vecteurs supports.
Partant de ce principe, l'introduction de la constante ν permet d'obtenir la formulation primale du ν-SVR [SSWB00] :

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\xi^*,\epsilon} \quad & \frac{1}{2}||\mathbf{w}||^2 + C(\nu\epsilon + \frac{1}{N}\sum_{i=1}^{N}(\xi_i + \xi_i^*)\ ) \\ \text{s. c.} \quad & y_i - (\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \leq \epsilon + \xi_i \\ & (\langle \mathbf{w}, \mathbf{x_i} \rangle + b) - y_i \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0, \epsilon \geq 0 \\ & i \in \{1, \ldots, N\} \end{aligned} \tag{2.89}$$

La formulation duale du ν-SVR est alors :

$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & \frac{1}{2}\sum_{i,j}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\langle \mathbf{x_i},\mathbf{x_j}\rangle - \sum_i y_i(\alpha_i-\alpha_i^*) \\ \text{s. c.} \quad & \sum_i y_i(\alpha_i-\alpha_i^*)=0 \\ & \sum_i(\alpha_i+\alpha_i^*)\leq C\nu \\ & \forall i\in\{1,\dots,N\},\quad 0\leq\alpha_i,\alpha_i^*\leq\frac{C}{N} \end{aligned} \tag{2.90}$$

La solution est alors donnée par

$$\mathbf{w} = \sum_{i=1}^{n}(\alpha_i-\alpha_i^*)\mathbf{x_i} \tag{2.91}$$

$$b = \frac{1}{2|S|}\sum_{(\mathbf{x},y)\in S\cup S^*} y-\langle\mathbf{w},\mathbf{x}\rangle \tag{2.92}$$

$$\epsilon = \frac{1}{2|S|}\Big(\sum_{(\mathbf{x},y)\in S} y-\langle\mathbf{w},\mathbf{x}\rangle - \sum_{(\mathbf{x},y)\in S^*} y-\langle\mathbf{w},\mathbf{x}\rangle\ \Big) \tag{2.93}$$

avec S (resp. S^*) un ensemble non vide de vecteurs supports $(\mathbf{x_i},y)$ tel que $0<\alpha_i<\frac{C}{N}$ et $\alpha_i^*=0$ (resp. $0<\alpha_i^*<\frac{C}{N}$ et $\alpha_i=0$) et $|S|=|S^*|$.

En outre, dans [Rak07], des méthodes itératives sont proposées pour effectuer une sélection de variables pour les SVR.

2.3.6 Les noyaux (*kernels*)

Les SVM ne peuvent être appliqués que sur des données linéairement séparable ou pouvant être linéairement séparées avec une quantité raisonnable d'erreur. Toutefois, les problèmes répondant à ces contraintes sont rares. En effet, la majorité des problèmes est composée de données reliées à des classes, ou plus généralement à une variable cible, par des relations complexes ne pouvant être modélisées linéairement. Il existe deux approches pour traiter ces problèmes.

La première approche consiste à utiliser un espace d'hypothèse contenant des fonctions non-linéaires. L'inconvénient de cette approche est que si les algorithmes linéaires tels qu'Adaline, les perceptrons [Ben06] et les SVM ont été l'objet de vastes études, il n'en est pas de même avec les algorithmes non-linéaires. En effet, ces algorithmes sont apparus plus tard avec les perceptrons mulicouches (PMC) [Bis95, BLT06]. Le frein majeur est la difficulté de trouver un algorithme pouvant être utilisé pour optimiser des fonctions non-linéaires. Dans le cas des

réseaux de neurones, il a fallu attendre l'algorithme de la rétro-propagation du gradient pour pouvoir utiliser les PMC.

La seconde approche consiste à trouver un espace dans lequel les données pourront être linéairement séparables [Vap95]. Nous appellerons un tel espace, espace de description, espace de caractéristiques ou encore espace d'attributs. Une fois cet espace défini, via une fonction de passage Φ qui associe une donnée à son image dans l'espace de description, il devient possible d'utiliser tout les algorithmes linéaires dans le nouvel espace. La figure 2.17 donne une illustration de ce principe. La difficulté, dans cette approche, réside dans la définition de la fonction de passage qui peut nécessiter une bonne connaissance des données. Dans cette section, nous nous intéresserons à cette approche.

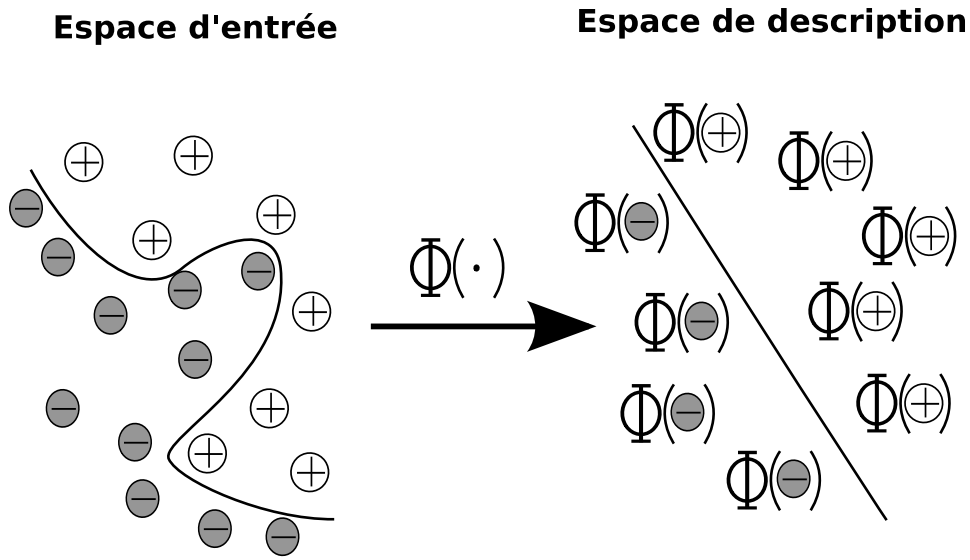


Fig. 2.17 – Passage des données de l'espace d'entrée vers un espace de description où les données sont linéairement séparables.

2.3.6.1 Définitions

Nous donnons d'abord la définition de l'espace de Hilbert que nous utiliserons pour la définition des espaces de description :

Définition 2.3.4 (Produit Scalaire)**.** *Soit $\mathcal{X}$ un espace vectoriel, l'application $k : \mathcal{X} \times \mathcal{X}$ est un produit scalaire dans $\mathcal{X}$, si k est une application symétrique bilinéaire strictement positive. k doit donc vérifier les propriétés suivantes :*

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

Définition 2.3.5 (Espace de Hilbert)**.** *Un espace de Hilbert $\mathcal{H}$ est un espace vectoriel complet doté du produit scalaire $\langle .,. \rangle_{\mathcal{H}}$ dont la norme découle du produit scalaire : $||f||_{\mathcal{H}}=\langle .,. \rangle^2_{\mathcal{H}}$.*

Nous définirons le noyau comme :

Définition 2.3.6 (Noyau)**.** *Un noyau est une fonction k tel que :*

$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{X},\ k(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle \tag{2.94}$$

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

Définition 2.3.7 (Matrice de Gram)**.** *Nous appellerons matrice de Gram, la matrice carrée G de taille $N \times N$ définie pour un ensemble de données $\{(\mathbf{x_i}, y_i)\}_{i=1}^N$ tel que $G_{ij} = k(\mathbf{x_i}, \mathbf{x_j})$.*

Les noyaux peuvent être utilisés dans les algorithmes basés sur le produit scalaire tels que les SVM pour résoudre des problèmes non-linaires [Vap95, STC04]. Par algorithmes basés sur le produit scalaire, nous désignons les algorithmes dépendant uniquement de la valeur du produit scalaire entre les données d'entrée. Ainsi, si un algorithme $\mathcal{A}(\{(\mathbf{x_i}, y_i)\}_{i=1}^N)$ peut-être représenté sous la forme $\mathcal{A}'(\{(\mathcal{S}_i, y_i)\}_{i=1}^N)$ avec $\mathcal{S}_i = \{\langle x_i, x_j \rangle\}_{j=1}^N$ alors $\mathcal{A}$ est un algorithme basé sur le produit scalaire.
Pour les SVM, il suffira de remplacer tous les produits scalaires standard $\langle .,. \rangle$ par des fonctions noyaux $k(.,.)$. Par exemple, la solution de classement pour le SVM défini dans l'équation 2.25 peut être réécrite comme ceci :

$$\begin{aligned} y &= sign(\langle \mathbf{w}, \mathbf{x} \rangle + b) \\ &= sign(\sum_{i=1}^{N} \alpha_i . y_i . k(\mathbf{x_i}, \mathbf{x}) + b) \end{aligned} \tag{2.95}$$

Ainsi, en définissant un noyau avec une fonction Φ non-linéaire, un algorithme linéaire peut devenir non linéaire. Le passage vers l'espace de description est effectué par le noyau et est donc implicite pour l'algorithme d'apprentissage. Ceci est appelé l'astuce du noyau.

Toutefois, l'inconvénient du noyau est la définition de la fonction Φ. Les inconvénients majeurs sont :

1. une bonne définition de Φ exige une connaissance de l'espace de description. Or, dans la majorité des cas, cette connaissance fait défaut. Il serait plus intuitif et avantageux de définir une fonction de similarité entre les données à l'instar du produit scalaire qui peut être perçu comme une mesure de similarité basée sur l'angle formé par les deux vecteurs.

2. lorsque l'espace de description est de haute dimension et/ou complexe, il peut être calculatoirement coûteux d'effectuer le passage vers l'espace de description avant d'effectuer le calcul du produit scalaire. Une expression optimisée du produit scalaire sans transformation explicite peut être plus adaptée.
3. dans le cas d'un espace de description de dimension infini comme celui induit par une fonction gaussienne (RBF-*Radial Basis Function*-), la définition de Φ est impossible.

Nous verrons dans la section suivante comment définir un noyau sans pour autant définir explicitement Φ.

2.3.6.2 Noyau de Mercer

Comme nous l'avons vu précédemment, il peut être plus aisé de définir une fonction de similarité entre données plutôt qu'une fonction de transformation. En effet, il est plus simple de concevoir une fonction de similarité, avec une connaissance a priori des données, pour obtenir une meilleure discrimination que de construire un espace de description approprié et la fonction de passage associée.
Dans cette section, nous verrons qu'une fonction $k(\mathbf{x}, \mathbf{y})$ peut être, sous certaines conditions, un noyau valide, à savoir un produit scalaire dans un espace de Hilbert, sans avoir à définir de fonction de transformations.

Définition 2.3.8 (Noyau Valide)**.** *Une fonction $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ est un noyau valide si et seulement si k est un produit scalaire dans un espace de Hilbert.*

Étant donné une fonction k continue semi-definie positive (condition 2.96), la condition suffisante de Mercer, issue du théorème 2.3.9, permet de conclure qu'il existe un espace de Hilbert dans lequel k est un produit scalaire. Par conséquent, les noyaux vérifiant le théorème de Mercer, appelés noyaux de Mercer, sont des noyaux valides.

Theorème 2.3.9 (Mercer)**.** *Soit $\mathcal{X}$ un sous-espace compact de $\mathbb{R}^n$. Supposons que k est une fonction symétrique continue tel que l'opérateur d'intégration $T_k : L_2(\mathcal{X}) \rightarrow L_2(\mathcal{X})$:*

$$(T_k f)(.) = \int_{\mathcal{X}} k(., \mathbf{x}) f(\mathbf{x}) d\mathbf{x}$$

est positive, à savoir :

$$\int_{\mathcal{X}} \int_{\mathcal{X}} k(\mathbf{x}, \mathbf{y}) f(\mathbf{x}) f(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0 \tag{2.96}$$

pour toute fonction $f \in L_2(\mathcal{X})$. Alors, $k(\mathbf{x}, \mathbf{y})$ peut être décomposé en séries uniformément convergentes, $\phi_j \in L_2(\mathcal{X})$, sur $\mathcal{X} \times \mathcal{X}$:

$$k(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{\infty} \lambda_j \phi_j(\mathbf{x}) \phi_j(\mathbf{y})$$

avec ϕ_j les fonctions propres de T_k tel que $||\phi_j||_{L_2} = 1$ et $\lambda_j > 0$ les valeurs propres associées.

De plus, les séries $\sum_{j=1}^{\infty} ||\Phi_j||^2_{L_2(\mathcal{X})}$ sont convergentes.

Bien qu'il soit possible, à l'aide du théorème de Mercer, de s'assurer qu'une fonction k soit un noyau valide, il n'est généralement utilisé que pour la construction de l'espace de Hilbert pour un noyau valide. En effet, démontrer la condition 2.96 peut s'avérer difficile. De plus, dans la majorité des cas, les noyaux sont définis sur des ensembles discrets. Ainsi, pour démontrer la validité d'un noyau, nous utiliserons de préférence les propriétés de l'Espace d'Hilbert à Noyau Reproduisant [STC04, CMR03, RC05].

2.3.6.3 Espace d'Hilbert à Noyau Reproduisant (RKHS)

Dans cette section, nous donnerons des outils permettant de démontrer aisément la validité des noyaux. Dans le cas des ensembles discrets, ces outils seront plus simples d'utilisation que le puissant théorème de Mercer.

Nous introduirons la définition d'un noyau semi-défini positif équivalente à la propriété 2.96 mais plus adaptée au cas discret :

Définition 2.3.10 (Noyau semi-défini positif)**.** *Une fonction $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est dite semi-définie positive sur $\mathcal{X}$ si k est symétrique :*

$$\forall(\mathbf{x}, \mathbf{y}) \in \mathcal{X}^2, k(\mathbf{x}, \mathbf{y}) = k(\mathbf{y}, \mathbf{x}) \tag{2.97}$$

et si $\forall N \in \mathbb{N}, (\mathbf{x_1}, \ldots, \mathbf{x_N}) \in \mathcal{X}^N$, $\forall (a_1, \ldots, a_N) \in \mathbb{R}^N$, la matrice de Gram K associée à k (avec $K_{ij} = k(\mathbf{x_i}, \mathbf{x_j})$) est semi-définie positive :

$$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j K_{ij} \geq 0$$

Définition 2.3.11 (Noyau reproduisant)**.** *Un noyau $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est dit noyau reproduisant s'il vérifie la propriété reproduisante suivante :*

$$\forall \mathbf{x} \in \mathcal{X}, f \in \mathcal{H}, \ f(\mathbf{x}) = \langle f, k_x \rangle_{\mathcal{H}}$$

en particulier :

$$\forall(\mathbf{x}, \mathbf{y}) \in \mathcal{X}^2, \langle k(\mathbf{x}, .), k(\mathbf{y}, .) \rangle_{\mathcal{H}} = k(\mathbf{x}, \mathbf{y})$$

Définition 2.3.12 (Espace de Hilbert à Noyau Reproduisant (RKHS))**.** *Soit $\mathcal{X}$ un espace d'entrée et $\mathcal{H}$ un espace de Hilbert de fonctions tel que $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$, $\mathcal{H}$ est un RKHS si et seulement si il existe un noyau reproduisant $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ tel que $\mathcal{H}$ contient toutes les fonctions $k_x : \mathcal{X} \to \mathbb{R}$:*

$$\forall(\mathbf{x}, \mathbf{y}) \in \mathcal{X}^2, k_x(\mathbf{x}) = k(\mathbf{x}, \mathbf{y}), \ k_x \in \mathcal{H}$$

Un noyau reproduisant est unique dans un RKHS [Aro50].

Theorème 2.3.13. *Noyau semi-defini positif et reproduisant [Aro50] Si un noyau* $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ *est semi-défini positif alors il existe un RKHS ayant* k *pour noyau reproduisant. Réciproquement, si* k *est un noyau reproduisant alors* k *est semi-défini positif.*

Le théorème 2.3.13 nous assure que si un noyau est semi-défini positif alors il existe un RKHS dans lequel le noyau pourra être exprimé sous la forme d'un produit scalaire. Par conséquent, un noyau semi-défini positif est un noyau valide.
Ainsi, une fonction de similarité bilinéaire symétrique pourra être utilisée comme noyau valide après s'être assuré qu'elle est semi-définie positive. Nous reformulerons la définition 2.3.8 du noyau valide par le théorème suivant :

Theorème 2.3.14 (Noyau valide)**.** *Une fonction symétrique continue ou a domaine fini* k : $\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ *est un noyau valide si et seulement si* k *est un noyau semi-defini positif.*

Il est important de noter qu'aucune contrainte n'a été imposée sur l'espace d'entrée $\mathcal{X}$. Par conséquent, un noyau peut être défini sur des données de type varié et complexe : vecteur, graphe, texte, arbre, etc. Nous étudierons les noyaux pour données structurées dans la section 3.3. Les algorithmes numériques tels que les SVM pourront être appliqués sur des données non-numériques. En outre, il est aussi possible d'intégrer des méthodes génératives dans les algorithmes discriminants en utilisant des noyaux génératives [Hof00, TKR+02, TMY04].

2.3.6.4 Définition de nouveaux noyaux valides

La définition d'un noyau peut s'avérer complexe notamment pour la vérification de sa validité. Pour simplifier la construction de nouveaux noyaux, il est possible de combiner des noyaux valides pour former un nouveau noyau. Étant donné deux noyaux k_1 et k_2 sur $\mathcal{X}^2$ ($\mathcal{X} \subset \mathbb{R}^n$), une fonction $f : \mathcal{X} \rightarrow \mathbb{R}$, une fonction $\Phi : \mathcal{X} \rightarrow \mathbb{R}^N$, un noyau $k_3 : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}$, $a \in \mathbb{R}^+$ et B une matrice $N \times N$ symétrique semi-définie positive, les fonctions suivantes sont des noyaux [STC04] :

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

Le tableau 2.1 donne quelques exemples de noyaux bien connus pouvant servir de base pour la définition de nouveaux noyaux.

	Noyau
Noyau Linéaire	$K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle$
Noyau Polynomial	$K(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y} \rangle + C)^d$
	$C \in \mathbb{R}^+, d \in \mathbb{N}$
Noyau Gaussien	$K(\mathbf{x}, \mathbf{y}) = exp(-\frac{\|\|\mathbf{x}-\mathbf{y}\|\|^2}{2\sigma^2})$
	$\sigma \in \mathbb{R}^+$

Tab. 2.1 – Quelques noyaux bien connus.

2.4 Conclusion

Dans la première partie de ce chapitre, nous avons brièvement présenté un état de l'art sur les différents types d'apprentissage numérique. Nous avons, notamment, vu les deux principaux critères de minimisation du risque utilisé comme principe d'induction pour l'extraction des relations entre les données.

Puis, dans la seconde partie de ce chapitre, nous avons présenté la famille d'algorithme des Séparateurs à Vaste Marge issus du principe de la minimisation du risque structurel (SRM). Ces algorithmes connaissent un large succès de part leurs performances. L'utilisation de la SRM assure aux SVM un bon niveau de généralisation et les rend peu sensible à la haute dimension.

De plus, nous avons vu que les noyaux permettent d'utiliser, implicitement, les SVM pour des problèmes non linéaires. Les noyaux permettent de plonger, implicitement, les données d'entrée dans un espace de description prévu pour que les données soient linéairement séparables. Ces noyaux peuvent être définis pour des données variées, élargissant la gamme des problèmes pouvant être résolus par les SVM.

Dans la suite de cette thèse, nous nous intéresserons aux noyaux pour les données textuelles.

CHAPITRE 3

Apprentissage à base de noyaux pour le texte

3.1 Introduction

Nous avons vu dans le chapitre précédent que les noyaux peuvent être définis sur des espaces d'entrée quelconques. Cette caractéristique des noyaux permet d'appliquer des algorithmes numériques[1] tels que les SVM sur des données complexes (complexe au niveau structurel) non-nécessairement numériques. Ainsi, l'utilisation des noyaux pour les données textuelles suscite un engouement certain. En effet, avec les noyaux, il devient possible d'utiliser des représentations adaptées aux textes, prenant en compte la nature des informations portées par de telles données.

Dans la première partie de ce chapitre, nous présentons un cadre d'application de l'apprentissage aux données textuelles. Nous exposons différents outils, relatifs aux textes, qui nous sont nécessaires pour le traitement de ce type de données. Puis, dans la deuxième partie, nous décrivons les principaux noyaux pour le traitement des documents semi-structurés tels que les fichiers XML. Ces noyaux ont la spécificité d'être définis sur des structures telles que les arbres et les graphes.

3.2 Apprentissage pour données textuelles

L'augmentation des ressources textuelles nécessite des systèmes de traitement de l'information de plus en plus sophistiqués. Extraire les informations pertinentes à partir de grandes bases de données à un coût raisonnable est devenu un enjeu important. Pour répondre à ce problème plusieurs axes de recherches ont été explorés. L'approche la plus populaire a été sans nul doute issue de l'ingénierie des connaissances. Un expert était chargé de définir des règles sur la manière de classer et d'extraire l'information. Toutefois avec l'augmentation de la quantité

[1] Nous parlons, ici, des algorithmes numériques dits *kernelizables* à savoir des algorithmes pouvant être exprimés uniquement à partir des produits scalaires entre les données.

de données, cette approche est devenue difficile, nécessitant de plus en plus de temps de la part de l'expert. Dans les années 90, la recherche s'est focalisée sur les approches par apprentissage numérique. L'apprentissage automatique permet de construire des modèles résolvant le problème par un processus inductif à partir d'un échantillon de données préalablement traité (par un expert) dans le cas de l'apprentissage supervisé ou de découvrir des structures en ne nécessitant aucune information préalable dans le cas de l'apprentissage non supervisé. De plus, les données étant souvent représentées par des vecteurs de mots, l'espace des documents est à haute dimension dans lequel le problème peut être résolu par des fonctions linéaires [Joa02]. Ces avantages ont permis l'essor de multiples techniques issues de l'apprentissage numérique dont les méthodes à noyaux. Ces méthodes permettent de traiter le problème dans un espace plus approprié que l'espace d'origine. Ainsi, la pertinence du modèle peut être améliorée en choisissant un espace caractérisant au mieux les données. La pertinence du modèle est alors étroitement liée au choix du noyau.

Dans la suite de cette section, nous ferons un point sur la problématique liée aux données textuelles et nous verrons que bon nombre de problèmes peuvent soit être ramenés à des problèmes de catégorisation soit faire appel à des tâches de catégorisation pour la résolution. Par conséquent, nous utiliserons la catégorisation comme cadre applicatif pour cette thèse. Toutefois, les noyaux qui y seront développés seront bien entendu utilisables dans un contexte plus général.

En outre, nous présenterons le pré-traitement des textes pour l'apprentissage qui nous servira d'armature pour le développement de nos systèmes. Puis, nous exposerons les critères d'évaluation des systèmes et nous terminerons en présentant quelques méthodes de catégorisation dont certaines seront utilisées comme base de comparaison pour évaluer nos méthodes.

3.2.1 Problématique liée aux textes

Avec l'avènement de l'Internet, le volume de données textuelles disponibles ne cesse de croître. La recherche de documents textuels pertinents et leurs exploitations constituent de nouveaux défis pour la fouille automatique de données. On distingue généralement deux grandes catégories de problèmes : la recherche d'informations (*Information Retrieval*) et l'extraction d'informations (*Information Extraction*).

3.2.1.1 La recherche d'informations

La recherche d'informations s'intéresse particulièrement à la catégorisation de textes. La catégorisation consiste à associer à chaque document une étiquette selon des critères de similarité ou de pertinence. Ainsi, les documents pourront être classés en catégories définies par leurs étiquettes [Seb02]. Cette technique permet de répondre à deux problèmes cruciaux rencontrés dans la recherche documentaire :

– L'indexation est utilisée pour classer un ensemble de documents en catégories dans le but

d'extraire rapidement des documents répondant à des critères donnés [DDL+90, Hof99b]. Les étiquettes correspondant aux catégories de documents n'ont pas forcément besoin d'être prédéfinies. En effet, les méthodes de classification (apprentissage non supervisé) peuvent découvrir des relations de similarité entre les documents et former des classes de documents. L'une des applications de l'indexation est le moteur de recherche notamment sur Internet.

– Le classement est utilisé pour attribuer des étiquettes à des documents jusqu'alors inconnus selon le principe de similarité [Seb02, Joa02]. Le classement fait partie des méthodes supervisées. Il utilise un échantillon de données textuelles correctement étiquetées afin de découvrir les relations reliant les étiquettes aux types de données et construire un modèle par induction. Le modèle est alors utilisé pour classer les documents (leur attribuer une étiquette). Les applications de la catégorisation sont nombreuses telles que le filtrage de courriers électroniques, l'identification de l'auteur d'après son style, la reconnaissance du langage.

3.2.1.2 **L'extraction d'informations**

L'extraction d'informations consiste à recueillir des connaissances contenues dans un document textuel. Il s'agit d'extraire des informations pertinentes répondant à des questions précises. L'extraction de connaissances fait souvent appel à une analyse linguistique et sémantique du texte comme dans le cas des systèmes de questions-réponses. Un exemple d'un tel système est le remplissage de formulaires prédéfini [Poi99b]. Le système doit remplir chaque champ du formulaire, à partir d'information émanant du document à analyser, en sachant qu'à chaque champ est associé une question.
Ce problème a fait surgir d'autres sous-problèmes tel que la reconnaissance et l'identification d'entités nommées [Poi99a]. Les entités nommées sont généralement des noms de personnes, d'organisations, des lieux géographiques, des dates mais peuvent aussi être étendues à d'autres domaines tel que la biologie moléculaire où chaque nom de molécule est construit sur la base d'un schéma conceptuel [TC02]. La reconnaissance et l'identification des entités nommées ont souvent fait appel aux techniques classique du TAL (Traitement Automatique des Langues) basées sur l'utilisation de dictionnaire. Toutefois, ces dernières années des systèmes hybrides sont apparus. L'extraction étant faite par des systèmes classiques et l'identification, qui peut être perçue comme de la catégorisation, est effectuée par des approches d'apprentissage numérique [TC02, SLL04, Sol04, ZPZ04, IK02].

Une autre application de l'extraction d'informations est la réalisation de résumé de documents. Cette tâche consiste à extraire, à partir d'un document, les phrases les plus caractéristiques du document [HIMM02, AG02, ZPZ04]. Le résumé automatique peut aussi être vu comme un problème de catégorisation où chaque phrase d'un document doit être étiquetée selon les étiquettes "pertinent" et "non pertinent" [HIMM02].

3.2.2 Pré-traitement pour l'apprentissage

Les algorithmes d'apprentissage numérique sont naturellement conçus pour être utilisés sur des données numériques. Par conséquent, les données textuelles doivent subir une phase de pré-traitement pour obtenir une représentation utilisable pour l'apprentissage. Dans le cadre de cette thèse, nous utiliserons la chaîne de pré-traitement illustrée par la figure 3.1.

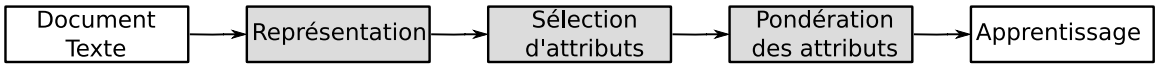


Fig. 3.1 – La chaîne de pré-traitement d'un document texte.

3.2.2.1 Représentation des données

La représentation des données est l'un des deux éléments clés dans les performances d'un système d'apprentissage, l'autre étant la mesure de similarité. En effet, les données doivent être préparées afin d'être utilisées en entrée d'un système d'apprentissage. Pour cela, il est nécessaire d'avoir une bonne connaissance a priori des données, pour pouvoir mettre l'accent sur les éléments pertinents tout en évitant les informations inutiles. Les éléments pertinents pourront, ensuite, être exploités lors de la phase d'apprentissage. L'essentiel des algorithmes d'apprentissage numérique exigent une représentation vectorielle des données. Toutefois, l'utilisation des noyaux dans les algorithmes numériques permet des représentations plus complexes, donc plus riches, telles que des structures en graphes ou arbres.

Les documents textes peuvent être représentés avec cinq niveaux de granularité :

1. **Représentation basée sur les caractères** : dans ce mode de représentation, l'accent est mis sur les séquences de caractères. L'un des modèles les plus connus est le modèle N-Gram. Dans ce modèle, le document est représenté par un ensemble de termes dont chaque terme est composé de N caractères. Le modèle N-Gram est souvent utilisé dans les domaines utilisant des termes composés comme le domaine biomédical. Par exemple, les noms de molécules sont souvent composés de séquences de caractères désignant une similarité (plus précisément une composition) avec d'autres molécules. Le N-Gram est alors capable de capturer ces similarités. De plus, les mots étant décomposés en sous-séquences de N caractères, le N-Gram est plus robuste aux erreurs d'écriture (orthographe). Toutefois, ce niveau est relativement primaire, puisque le texte est considéré comme une séquence de symboles dénuée de toutes significations linguistiques. Nous présenterons, dans la section 3.3.2, quelques noyaux utilisant cette représentation pour l'apprentissage.

2. **Représentation basée sur les mots** : la représentation d'un document en utilisant uniquement des mots clés définis dans un dictionnaire est probablement la plus utilisée. Le document est décomposé en un ensemble de mots. Il est possible d'ajouter une infor-

mation supplémentaire qui est la fréquence d'occurrence de chaque terme. Le document est ainsi caractérisé par la présence ou non de certains termes. Le modèle de l'espace vectoriel (VSM- *Vector Space Model*), que nous décrirons plus bas, utilise ce niveau de représentation. Le succès de ces modèles repose sur la simplicité d'encodage et sur les bonnes performances des systèmes utilisant ce modèle. En effet, en général la présence de certains termes, notamment des termes spécifiques, permettent de calculer des similarités pertinentes entre les documents.

3. **Représentation basée sur les séquences de mots** : ce niveau peut être perçu comme le mélange des deux premiers niveaux. Un terme, dans cette représentation, n'est plus un mot unique mais un groupe de mots composé de un ou plusieurs mots. L'objectif de cette représentation est de tenir compte de l'aspect linguistique des termes. Par exemple, le groupe de mots "pomme de terre" désigne un sens particulier qui peut être perdu si on le décompose. Les modèles reposant sur cette représentation nécessitent des informations linguistiques telles que des lexiques (*lexicon*) ou thésaurus. En outre, certains modèles ajoutent des informations morpho-syntaxiques pour affiner le calcul de la similarité en augmentant le degré de détail. Parmi ces modèles, nous citerons les arbres syntaxiques et les graphes que nous verrons dans la section 3.3. Il est à noter que ces trois premiers niveaux sont des niveaux basés sur l'information morphologique.

4. **Représentation Sémantique** : cette représentation tente de modéliser le sens induit par le document. Les modèles basés sur cette représentation utilisent des concepts pour-désigner les différents sens présents dans le document. Les modèles sont, d'une part, plus riches et précis au niveau informationnel que les modèles morphologiques et, d'autre part, indépendants de la langue. L'inconvénient majeur est essentiellement l'extraction des concepts. Plus précisément, le problème réside dans la définition des concepts et dans la définition des relations d'association entre concepts et termes linguistiques. Les deux approches pour tenter de répondre au problème sont soit d'utiliser des connaissances linguistiques a priori, soit d'extraire les concepts statistiquement.

   Dans la première approche, la représentation se base sur des thésaurus et des ontologies définies par des experts linguistes. Elle nécessite par là-même une intervention humaine qui peut être coûteuse. Bien que les informations fournies par les experts soient riches, cette approche doit faire face à deux problèmes. Premièrement, c'est une approche a priori donc fortement dépendante du domaine. En effet, les domaines spécialisés font appels à des ressources linguistiques très spécifiques. Deuxièmement, les relations entre concepts et termes entraînent bien souvent des ambiguïtés qui ont tendance à noyer l'information. Ces ambiguïtés sont inhérentes à la langue.

   Dans la seconde approche, les concepts sont extraits par des méthodes statistiques exploitant l'information latente aux données. Les concepts sont ainsi matérialisés par des relations entre les termes. La méthode la plus connue, utilisant, cette approche est l'analyse sémantique latente (LSA-*Latent Semantic Analysis*). Si l'avantage majeur est que

cette approche n'exige pas de connaissances a priori et est donc indépendante des domaines d'application, il n'en reste pas moins que les concepts extraits sont purement statistiques. Par conséquent, ces concepts peuvent avoir un sens linguistiquement difficile à interpréter ou même pire n'avoir aucune signification linguistique.

La représentation sémantique est très prometteuse du fait que le document est représenté par des concepts modélisant le sens du document : l'information est précise. De plus, si des informations sur les relations entre concepts, telles que les ontologies, sont disponibles, il devient alors possible de calculer des similarités entre documents de manière très pertinente. Dans notre thèse, nous nous intéresserons uniquement à cette représentation et plus particulièrement à l'approche basée sur les concepts statistiques.

5. **Représentation pragmatique** : ce niveau de représentation est, sans nul doute, le plus riche est le plus complexe. Nous avons vu que l'un des inconvénients de la représentation sémantique était les ambiguïtés générées dans les relations entre concepts et termes. Le niveau pragmatique permet de lever ces ambiguïtés en prenant en considération le contexte lors de l'extraction du sens du texte. Bien que cette représentation semble idéale pour l'extraction d'information, la gestion du contexte et la désambiguïsation sont des tâches difficiles qui font toujours l'objet de recherches actives.

Bien que nos recherches se concentrent sur le niveau sémantique, nous présentons très brièvement le modèle de l'espace vectoriel (VSM) qui est aussi appelé représentation en sac-de-mots. Le VSM nous permettra d'illustrer le reste des étapes du pré-traitement.

Le VSM est l'une des représentations la plus utilisée dans le domaine du traitement de données textuelles. Ce modèle a été introduit dans [SWY75] pour résoudre des tâches d'indexations pour la recherche documentaire. Dans le VSM, un dictionnaire de terme est utilisé. Le dictionnaire définit un espace vectoriel de haute dimension dans lequel chaque axe est associé à un terme du dictionnaire. Un document d peut alors être représenté par un vecteur $\mathbf{V}$. Chaque composant v_i de $\mathbf{V}$ détermine la fréquence d'occurrence du terme t_i, appartenant au dictionnaire, dans le document d. La fréquence peut aussi être remplacée par une valeur binaire indiquant la présence du terme dans le document mais le modèle est plus efficace lorsque l'information sur la fréquence est utilisée. La figure 3.2 illustre la représentation du VSM.

Le succès du VSM est essentiellement dû au fait que la représentation est vectorielle. Ainsi, tous les algorithmes d'apprentissage numérique peuvent être utilisés pour résoudre des problèmes textuelles avec la représentation VSM.

3.2.2.2 Sélection d'attributs

La sélection d'attributs s'apparente à la réduction de dimension dans un espace vectoriel. Il s'agit de choisir les termes les plus significatives ou, dans une autre formulation, d'éliminer les termes les moins pertinents. Cette étape est importante car les représentations obtenues à

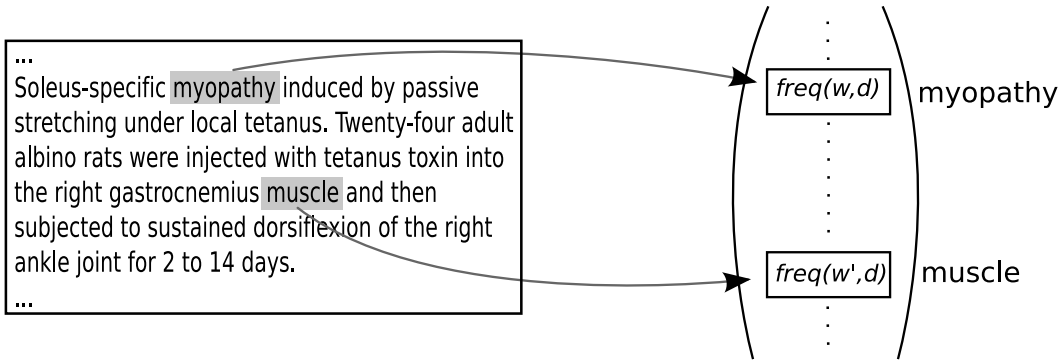


Fig. 3.2 – La représentation d'un document textuel avec le modèle de l'espace vectoriel.

partir de données textuelles sont souvent très riches. Dans le VSM, l'espace est à très haute dimension. Par conséquent, l'information pertinente risque d'être "noyée" dans la représentation.

En matière de document textuel, notamment dans la représentation VSM, les deux méthodes de réduction incontournables sont l'élimination des mots vides et la lemmatisation.

Les mots vides sont les mots fréquents qui n'apportent que très peu d'information et aucune information discriminative. Ces mots sont essentiellement les auxiliaires et les déterminants.

La lemmatisation permet de réduire les mots fléchis à une racine commune (lemme). Un verbe conjugué sera ramené à sa racine qui sera le verbe à l'infinitif (par exemple "discutions" deviendra "discuter"). Cette méthode permet de réduire considérablement la dimension de l'espace tout en conservant l'essentiel de l'information. Dans le cadre de cette thèse, nous utiliserons le terme *stemming* pour désigner la lemmatisation. En effet, nous n'utiliserons pas la lemmatisation au sens linguistique à savoir que nous utiliserons une forme commune aux mots fléchis et non le lemme proprement dit. Par exemple, pour le mot "discutions", la lemmatisation donnera "discuter" mais le *stemming* donnera "discut". Le *stemming* utilise des règles simples, propres à une langue, de flexion des mots pour trouver la base commune alors que la lemmatisation repose sur un lexique.

Bien que l'élimination des mots vides et le *stemming* permet de réduire l'espace de manière significative, il conservera toujours une dimension importante. Il est donc nécessaire de recourir à des méthodes statistiques permettant de quantifier l'apport d'information d'un terme.
Dans la suite de cette section, nous présentons quelques méthodes largement utilisés dans la catégorisation de texte [YP97, Seb02, Seb99]. Ces méthodes sont utilisées comme critère de dépendance entre un terme et la variable "classe". Les termes ayant des valeurs de dépendance faibles pourront être éliminés. En effet, dans ce cas, les termes et la classe sont considérés indépendants. Par conséquent, ces termes ne seront pas discriminatifs.

L'information mutuelle

L'information mutuelle permet de quantifier le degré de dépendance statistique entre un

terme et une classe. Elle est donnée par la formule suivante :

$$I(t_k, c_i) = \log(\frac{P(c_i, t_k)}{P(c_i)P(t_k)}) \tag{3.1}$$

Lorsque cette mesure est nulle, le terme t_k et la classe c_i sont considérés indépendants. Cela se traduit au niveau probabiliste par : $P(c_i, t_k) = P(c_i)P(t_k)$. Ainsi, la présence de t_k dans un document n'apporte aucune information sur le degré d'appartenance du document à la catégorie c_i.

Le Gain d'information

Le gain d'information permet de mesurer la quantité d'information apportée par la présence et l'absence d'un terme pour une catégorie :

$$IG(t_k, c_i) = \sum_{t\in\{t_k,\overline{t_k}\},c\in\{c_i,\overline{c_i}\}} P(c,t)\log(\frac{P(c,t)}{P(c)P(t)}) \tag{3.2}$$

Cette mesure ne diffère de l'information mutuelle que par la prise en compte de l'information apportée par un terme. En effet, l'information mutuelle suppose que seule la présence d'un terme est pertinente pour la "réalisation" d'une classe. Or, l'absence d'un terme précis dans certains documents peut aussi être liée à une classe.
Le gain d'information peut ainsi être perçu comme l'espérance de l'information mutuelle :

$$IG(t_k, c_i) = \mathbb{E}_{t\in\{t_k,\overline{t_k}\},c\in\{c_i,\overline{c_i}\}}(I(t,c)) \tag{3.3}$$

Dans le cadre de notre thèse, nous utiliserons un gain d'information limité à la présence et l'absence d'un terme, soit :

$$IG(t_k, c_i) = \sum_{t\in\{t_k,\overline{t_k}\}} P(c_i,t)\log(\frac{P(c_i,t)}{P(c_i)P(t)}) \tag{3.4}$$

La méthode χ^2

Le test du χ^2 permet de déterminer le degré d'indépendance entre un terme et une classe :

$$\chi^2(t_k, c_i) = \frac{N[P(t_k,c_i)P(\overline{t_k},\overline{c_i}) - P(t_k,\overline{c_i})P(\overline{t_k},c_i)]^2}{P(t_k)P(c_i)P(\overline{t_k})P(\overline{c_i})} \tag{3.5}$$

avec N le nombre total de documents utilisé pour l'estimation (la taille de l'échantillon).

Une valeur nulle pour cette mesure indique que le terme et la classe sont indépendants. Nous noterons que la méthode χ^2 est une technique purement statistique tandis que l'information mutuelle et le gain d'information sont issus de la théorie d'information.

Mesure globale à toutes les catégories

Certaines méthodes de sélection de variables se basent sur le degré de dépendance entre un terme et une classe. Toutefois, une méthode de sélection de variable doit déterminer la quantité d'information apportée par un terme pour toutes les classes. Il est, par conséquent, nécessaire d'estimer la quantité d'information qu'apportera un terme dans la catégorisation. En effet, un terme apparaissant uniquement dans certaines classes apportera plus d'information qu'un terme uniformément distribué sur les classes. Pour calculer une mesure globale à toutes les classes à partir d'une mesure locale $f(t_k, c_i)$, il existe principalement trois méthodes [Seb02] :

1. **la fonction max** :
$$f_{\max}(t_k) = \max_{c_i} f(t_k, c_i) \tag{3.6}$$
2. **la fonction somme** :
$$f_{\text{sum}}(t_k) = \sum_{c_i} f(t_k, c_i) \tag{3.7}$$
3. **la fonction espérance** :
$$f_{\text{e}}(t_k) = \mathbb{E}_{c_i}(f(t_k, c_i)) = \sum_{c_i} p(c_i) f(t_k, c_i) \tag{3.8}$$

3.2.2.3 Pondération

Dans un corpus de documents, les termes n'ont pas le même pouvoir expressif. Il est souvent plus efficace de donner un poids plus important aux termes expressifs qu'aux autres termes. Toutefois, la problématique de cette approche réside dans la définition du critère permettant de mesurer le pouvoir d'expressivité des termes. Il est à noter que la pondération est un cas particulier de la sélection d'attributs.

De part son importance cruciale dans les performances des systèmes de traitement de données textuelles, la pondération des termes est devenue un domaine très actif de la recherche notamment dans le cadre de la recherche documentaire [SB87, ZM98].

Pondération non-supervisée

Les méthodes de pondération non-supervisées n'utilisent aucune information autre que la donnée. Ainsi, l'information sur la classe des documents n'est pas utilisée.

De multiples méthodes de pondération ont été définies dont les principales sont reportées dans le tableau 3.1. Bien que ces méthodes semblent différentes, elles se basent sur trois observations [ZM98, DS03] :

1. les termes fréquents dans un document peuvent être caractéristiques du document ;
2. les termes n'apparaissant que dans très peu de documents d'un corpus peuvent être des termes spécifiques caractérisant un document ou un groupe de documents ;
3. la taille d'un document n'est pas forcément significative dans l'apport d'information.

La pondération la plus utilisée est la pondération TF-IDF [Jon88, Joa97]. La pondération TF (*Term Frequency*) pour un terme t d'un document d donne la fréquence d'occurrence de t dans le document d. Ainsi, les termes fréquents d'un document auront une influence supérieure aux autres termes du document. Cette pondération se base donc sur la première observation citée plus haut. La pondération IDF (*Inverse Document Frequency*), quant à elle, résulte de la seconde observation. Elle permet de mettre l'accent sur les termes rares au sein d'un corpus. Elle est définie pour un corpus $\mathcal{D}$ par :

$$\text{idf}(t_i, \mathcal{D}) = \log(\frac{|\mathcal{D}|}{|\{d'|w_i \in d', d' \in \mathcal{D}\}|}) \tag{3.9}$$

La combinaison TF-IDF permet de satisfaire les deux premières observations, la troisième est satisfaite en normalisant la représentation du document selon le nombre de terme. Les documents normalisés auront tous une taille unitaire.
La pondération TF-IDF permet d'obtenir d'excellents résultats [Joa97] dans le cas général. Les autres pondérations ont des performances fluctuantes selon le corpus et les méthodes d'apprentissage. Ainsi, le TF-IDF est, naturellement, devenu la pondération la plus communément utilisée.

Pondération	**Formule**	**Référence**
Binaire	$w_i = \begin{cases} 1 & \text{si } t_i \in d \\ 0 & \text{sinon} \end{cases}$	[SWY75]
TF	$w_i = \text{tf}(t_i, d)$	[SWY75]
TF-IDF	$w_i = \text{tf}(t_i, d)\text{idf}(t_i, \mathcal{D})$	[Jon88, Joa97]
log de TF	$w_i = \log(\text{tf}(t_i, d) + 1)$	[BSAS94]
log de TF-IDF	$w_i = \log(\text{tf}(t_i, d) + 1)\text{idf}(t_i, \mathcal{D})$	[BSAS94]
ITF	$w_i = 1 - \frac{1}{\text{tf}(t_i,d)+1}$	[LK02, DIR03]

Tab. 3.1 – Méthodes usuelles de pondération non-supervisée pour un terme t_i d'un document d.

Plus récemment, deux pondérations efficaces ont été proposées : la pondération ITF[LK02] et RW[HMB07]. L'ITF (*Inverse Text Frequency*), présentée dans le tableau 3.1, permet de réduire l'influence des termes excessivement fréquents. L'idée est qu'à partir d'une certaine fréquence, les termes ont approximativement la même importance. Par conséquent, le poids accordé par l'ITF va donc croître très rapidement pour les fréquences faibles et moyennes, et se stabiliser pour les fréquences élevées. Cette pondération s'est montrée prometteuse, conduisant à des performances légèrement supérieures que le TF-IDF d'après les expériences menées dans [HMB07, LK02, DIR03].

La pondération RW (*Random-Walk*)[HMB07] a été définie comme une alternative au TF. Elle se base sur l'algorithme PageRank [BP98] pour attribuer à chaque terme d'un document une pondération qui est le score PageRank. L'idée du RW est de tenir compte des co-occurrences d'ordre supérieur entre les termes. Pour cela, un graphe de termes est construit pour chaque document d. Chaque terme de d est matérialisé par un noeud dans le graphe. Un arc de t_1 vers t_2 est créé dans le graphe uniquement si t_1 et $t2$ apparaissent à proximité l'un de l'autre dans d. La proximité est définie par une fenêtre de taille prédéfinie. En outre, si un arc de t_1 vers t_2 est ajouté alors un arc symétrique sera ajouté. À partir du graphe obtenu, un score de PageRank peut être calculé pour un terme t par une marche aléatoire selon la formule récursive suivante :

$$\mathrm{rw}(t) = (1-p) + p \sum_{t_i \in I(t)} \frac{\mathrm{rw}(t_i)}{|O(t_i)|} \tag{3.10}$$

avec $I(t)$ l'ensemble de arcs entrants de t, $O(t)$ l'ensemble des arcs sortants de t et p la probabilité (prédéfinie) de continuer la marche ($1-p$ étant dans ce cas la probabilité de s'arrêter). Cette équation peut être résolue par la méthode de Jacobi. En effet, la convergence est assurée par la probabilité p de continuer.

La méthode RW de base ne prend pas en compte la fréquence des termes dans un document et la rareté des termes au sein du corpus. Toutefois, ces informations, sous la forme du TF-IDF, sont intégrées dans la méthode RWe [HMB07] en modifiant la probabilité p de continuer la marche. Ainsi, cette probabilité n'est plus constante mais varie selon l'arc. La probabilité sera importante pour les arcs reliant deux termes avec des TF-IDF importantes. Pour cela, un poids E_{t_1,t_2} sera affecté à chaque arc allant de t_1 vers t_2 tel que :

$$E_{t_1,t_2} = E_{t_2,t_1} = tf(t_1,d)idf(t_1,\mathcal{D})tf(t_2,d)idf(t_2,\mathcal{D}) \tag{3.11}$$

avec d le document modélisé par le graphe et $\mathcal{D}$ le corpus de travail.
La pondération RWe est donnée par la formule suivante :

$$\mathrm{rw}_e(t) = \frac{(1-p)}{N} + C * \sum_{t_i \in I(t)} \frac{P_{E_{t_i,t}} \mathrm{rw}_e(t_i)}{|O(t_i)|} \tag{3.12}$$

avec N le nombre de terme dans le document (de noeud dans le graphe), $C \in [0,1]$ une valeur préfixée de normalisation et $P_{E_{t_1,t_2}} = \frac{E_{t_1,t_2}}{max_{(t_i,t_j)} E_{t_1,t_2}}$.

Pondération supervisée

La pondération est une technique permettant de fixer un poids w_i à un terme t_i. La pondération supervisée utilise le degré de pertinence du terme t_i en fonction des classes pour fixer le poids w_i.

La sélection d'attributs que nous avons vu dans la section précédente est un cas particulier de la pondération. En effet, dans la sélection de variable, un poids binaire est affecté aux termes selon le souhait de conserver ou non certains termes. Ainsi, étant donné une fonction $f(t_i)$ indiquant le degré de dépendance entre le terme t_i et la variable "classe" (comme les fonctions vues dans la section 3.2.2.2), la sélection de variable revient à attribuer la pondération w_i à t_i tel que :

$$w_i = \begin{cases} 1 & \text{si } f(t_i) \geq S \\ 0 & \text{sinon} \end{cases} \tag{3.13}$$

avec S un seuil prédéfini.

Dans [DS03], la notion de pondération supervisée a été introduite en fixant simplement $w_i = f(t_i)$. Différentes fonctions f, telles que le χ^2 et le gain d'information, ont été utilisées pour les expériences d'évaluation. Les résultats n'ont montré qu'une légère amélioration par rapport à la pondération non supervisée TF-IDF.

D'autres travaux récents ont proposé des fonctions pour améliorer les performances. Dans [SM05], une fonction basée sur l'intervalle de confiance pour la proportion de documents contenant un terme a été proposée. Dans [LTL06, LTSL07], une fréquence de pertinence est définie :

$$rf(t, c_i) = \log(2 + \frac{N_{c_i}^+}{N_{c_i}^-}) \tag{3.14}$$

avec $N_{c_i}^+$ étant le nombre de documents de la classe c_i contenant le terme t et $N_{c_i}^-$ le nombre de documents contenant t_i et n'appartenant pas à c_i.

Bien que les méthodes supervisées apportent une certaine amélioration des performances, il n'a pas été clairement établit que ces méthodes sont plus adaptées que les méthodes non-supervisées. En effet, les performances obtenues par les méthodes de pondération supervisée et non-supervisée, fluctuent selon le corpus et la méthode d'apprentissage. Ainsi, le choix de la méthode de pondération doit être faite de manière empirique.

Les techniques de pondération sortant du cadre de notre thèse, nous utiliserons la pondération standard TF-IDF pour évaluer les performances de nos noyaux.
Toutefois, nous signalons que la pondération à un impact non-négligeable sur les performances. Par exemple, dans [SM05], certaines pondérations améliorent de 5% les performances par rapport au TF-IDF et cela pour un même système d'apprentissage.

3.2.3 Prise de décision pour le classement

Nous avons vu dans la section 2.3.2.6 des stratégies permettant de prendre des décisions de classement dans le cas multi-catégorie, notamment à partir de systèmes de classement binaire. Cependant, lorsqu'il s'agit d'un problème de multi-étiquetage, à savoir que les documents du problème peuvent appartenir à une ou plusieurs classes, la solution n'est pas triviale.

Dans la suite de cette section, nous présentons quelques stratégies permettant la prise de décision pour le multi-étiquetage [Yan01]. Pour cela, nous supposerons qu'un score $f(c_i|d)$ a été attribué pour chaque document d et chaque classe c_i.

3.2.3.1 **Stratégie** *RCut*

Dans cette stratégie, pour chaque document d, les t classes ayant obtenues les scores les plus élevés pour d sont sélectionnées pour étiqueter d. Le paramètre t est soit prédéfini, soit fixé par validation croisée.

3.2.3.2 **Stratégie** *PCut*

La stratégie *PCut* est définie comme ceci : pour chaque catégorie c_i, les k_i documents obtenant les meilleurs scores pour c_i seront étiquetés avec c_i, puis la procédure est répétée pour les autres classes. Il s'agit ici d'un tirage avec remise.
Les paramètres k_i sont définis par :

$$k_i = P(c_i) \times x \times m \tag{3.15}$$

avec m le nombre de classe, $P(c_i)$ la probabilité a priori d'avoir la classe c_i ($P(c_i)$ est estimée à partir de l'échantillon d'apprentissage) et x une valeur réelle prédéfinie ou à fixer par validation croisée.

3.2.3.3 **Stratégie** *SCut*

La stratégie *SCut* prévoit de traiter chaque catégorie séparément. Ainsi, pour chaque catégorie c_i, un seuil s_i est défini. Pour l'étiquetage d'un document d, d est affecté à toutes les classes c_k vérifiant $f(c_k|d) \geq s_k$.
Les seuils s_i doivent être optimisés individuellement sur un ensemble de test par validation croisée. Il est, aussi, efficace pour certains classifieurs, tels que les SVMs, de fixer les s_i à zéro.

3.2.4 **Les mesures des performances**

Les mesures de performance d'un classifieur se basent sur la matrice de contingence [Seb02]. Cette matrice, pour une classe c, indique le nombre d'erreur et de bon classement du système. Le tableau 3.2 donne un exemple de matrice de contingence avec :

- TP_c (vrai positif), le nombre de documents correctement classés dans la classe c,
- TN_c (vrai négatif), le nombre de documents correctement identifiés comme n'appartenant pas à la classe c,
- FP_c (faux positif), le nombre de documents négatif classés dans la classe c,
- FN_c (faux négatif), le nombre de documents positif classés comme étant négatif par le système.

	Positif	**Négatif**
Prédit Positif	TP_c	FP_c
Prédit Négatif	FN_c	TN_c

Tab. 3.2 – Matrice de contingence pour une classe c.

3.2.4.1 Précision

La précision est une mesure permettant d'évaluer la pertinence d'un ensemble de documents retournés par un système. Cette mesure est essentiellement utilisée dans le domaine de la recherche documentaire. Par exemple, un système d'indexation doit être capable d'extraire un ensemble de documents à partir d'un corpus selon des critères (requêtes). La précision permet alors de mesurer les performances de ce système, en évaluant le taux de documents pertinents (relativement à la requête) retournés par le système.
Pour un système de classement, pour une classe c, le taux de documents pertinents est donné par le rapport entre le nombre de documents bien classés dans c et le nombre total de documents classés, par le système, dans cette même classe. La précision est alors, en utilisant la matrice de contingence :

$$\text{précision}(c) = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FP}_c} \tag{3.16}$$

3.2.4.2 Rappel

La précision mesure uniquement le degré de pertinence des documents classés comme instances positives. Le rappel, quant à lui, mesure la complétude du système. Il mesure le taux du nombre de documents pertinents extraits par le système sur le nombre total de documents pertinents. Pour un système de classement, il s'agira du taux du nombre de documents correctement classés dans c sur le nombre de documents total appartenant à c. Plus formellement, nous avons :

$$\text{rappel}(c) = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FN}_c} \tag{3.17}$$

3.2.4.3 Mesure F_n

Dans le cas de données non séparables, les mesures de la précision et du rappel ne peuvent pas atteindre leurs valeurs optimales pour un même problème. En effet, si le système est paramétré pour extraire un grand volume de document pour une requête, alors le risque d'avoir des documents non pertinents sera important et, par conséquent, la précision risque de diminuer. Le rappel, quant à lui, augmentera car la probabilité d'avoir des documents pertinents croît avec le nombre de documents extraits. Ainsi, pour résumer, la précision diminue avec l'augmentation du nombre de documents extraits (nombre de documents classés positivement par le système) mais, à l'opposé, le rappel augmente avec le nombre de documents extraits.

Par conséquent, la précision ou le rappel ne permettent pas, isolément, de caractériser correctement un système de classement. Il est, donc, nécessaire d'utiliser une mesure qui sera un compromis entre la précision et le rappel. Pour cela, la mesure F_n est introduite. Cette mesure est une moyenne harmonique pondérée par n de la précision et du rappel :

$$\mathrm{F}_n(c) = \frac{(1+n) \times \text{précision}(c) \times \text{rappel}(c)}{n \times \text{précision}(c) + \text{rappel}(c)} \tag{3.18}$$

Le paramètre n permet de mettre l'accent sur la précision ou le rappel selon le problème à résoudre.

La mesure la plus utilisée est la mesure F_1 qui donne une même importance à la précision et au rappel :

$$\mathrm{F}_1(c) = \frac{2 \times \text{précision}(c) \times \text{rappel}(c)}{\text{précision}(c) + \text{rappel}(c)} \tag{3.19}$$

3.2.4.4 **Problème multi-catégorie**

Dans le cas d'un problème multi-catégorie, il existe deux types de moyennes pour combiner les mesures de performance pour chaque catégorie afin d'obtenir une mesure globale.

La première moyenne est la macro-moyenne. Elle consiste à effectuer la moyenne arithmétique des mesures. Nous obtenons les mesures suivantes :

$$\pi_{F_1} = \frac{2 \times \pi_{\text{précision}} \times \pi_{\text{rappel}}}{\pi_{\text{précision}} + \pi_{\text{rappel}}} \tag{3.20}$$

$$\pi_{\text{précision}} = \frac{1}{N_c} \sum_c \text{précision}(c) \tag{3.21}$$

$$\pi_{\text{rappel}} = \frac{1}{N_c} \sum_c \text{rappel}(c) \tag{3.22}$$

avec N_c le nombre de catégorie.

La macro-moyenne permet d'attribuer un même poids à toutes les catégories sans tenir compte des différences de taille entre les catégories. Les mesures macro-moyennées permettent d'obtenir des estimateurs moyens des performances du système pour la reconnaissance des documents d'une catégorie. Toutefois, il est à noter que les catégories faiblement représentées risquent d'avoir un impact négatif très important sur ces estimateurs. En effet, les performances d'un système sont dépendantes de la phase d'apprentissage et plus particulièrement de la qualité (et la quantité) de données disponibles par catégorie. Ainsi, un classifieur pour une petite catégorie aura des performances inférieur à un classifieur pour une catégorie importante. Il est ainsi important de tenir compte du biais induit par la taille lors de l'utilisation de la macro-moyenne.

La seconde moyenne est la micro-moyenne. Cette moyenne ne tient compte que des bons classements et des mauvais classements sans faire de distinction entre les classes. Pour cela,

une matrice de contingence globale est utilisée en effectuant une sommation sur les matrices de contingence de chaque classe. Le tableau 3.3 illustre une telle matrice.

	Positif	**Négatif**
Prédit Positif	$\mathrm{TP} = \sum_c \mathrm{TP}_c$	$\mathrm{FP} = \sum_c \mathrm{FP}_c$
Prédit Négatif	$\mathrm{FN} = \sum_c \mathrm{FN}_c$	$\mathrm{TN} = \sum_c \mathrm{TN}_c$

Tab. 3.3 – Matrice de contingence globale.

En utilisant la matrice de contingence, les mesures micro-moyennées sont données par :

$$\mu_{F_1} = \frac{2 \times \mu_{\text{précision}} \times \mu_{\text{rappel}}}{\mu_{\text{précision}} + \mu_{\text{rappel}}} \tag{3.23}$$

$$\mu_{\text{précision}} = \frac{TP}{TP + FP} = \frac{\sum_c TP_c}{\sum_c (TP_c + FP_c)} \tag{3.24}$$

$$\mu_{\text{rappel}} = \frac{TP}{TP + FN} = \frac{\sum_c TP_c}{\sum_c (TP_c + FN_c)} \tag{3.25}$$

La micro-moyenne permet d'attribuer un poids égal à chaque document. Ainsi, les catégories faiblement représentées n'auront qu'une influence proportionnelle à leurs tailles. Les mesures micro-moyennées sont particulièrement utiles pour évaluer les performances globales d'un système indépendamment des classes. Toutefois, le risque avec ces estimateurs est de sélectionner des modèles ayant d'excellentes performances micro-moyennées tout en ayant des performances médiocres sur les petites classes.

Le tableau 3.4 résume les mesures micro et macro-moyennées. La mesure F_n étant calculée de manière standard à partir de la précision et du rappel.

	Micro	**Macro**
Précision	$\frac{\sum_c \mathrm{TP}_c}{\sum_c (\mathrm{TP}_c + \mathrm{FP}_c)}$	$\frac{1}{N} \sum_c \text{précision}(c)$
Rappel	$\frac{\sum_c \mathrm{TP}_c}{\sum_c (\mathrm{TP}_c + \mathrm{FN}_c)}$	$\frac{1}{N} \sum_c \text{rappel}(c)$

Tab. 3.4 – Micro et macro moyennes pour N catégories.

3.2.5 Méthodes de référence pour la catégorisation de texte

3.2.5.1 Les SVM avec un noyau sac-de-mots

Le modèle de l'espace vectoriel (VSM) défini un espace vectoriel de très haute dimension. Or, dans ces espaces de haute dimension, les algorithmes d'apprentissage, et plus particulièrement ceux utilisant le principe de la minimisation du risque empirique, sont sujets au

sur-apprentissage. En effet, ces algorithmes ont d'excellentes performances sur l'ensemble d'apprentissage mais de très faibles performances sur des données nouvelles.
Les Séparateurs à Vaste Marge sont protégés contre le sur-apprentissage par l'utilisation du principe de la minimisation du risque structurel. En effet, nous avons vu que la capacité (VC-dimension) de ces systèmes, pour une valeur de marge supérieure à un certain niveau, n'était plus dépendante de la dimension de l'espace mais de la valeur de la marge. Par conséquent, les performances de généralisation des SVM sont moindrement affectées par la haute dimension de l'espace. Dans [Joa02, Joa98], un noyau linéaire, aussi appelé noyau sac-de-mots, a été présenté :

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle \tag{3.26}$$

avec x, y, deux documents et $\Phi(x)$, $\Phi(y)$ leurs représentations dans l'espace vectoriel VSM.

Il a été montré que les SVM avec ce noyau sac-de-mots permet d'obtenir d'excellentes performances pour la tâche de catégorisation de documents textuelles. Ces performances sont supérieures aux autres méthodes telles que les K-NN et le classifieur naïf de Bayes.

Cette méthode SVM avec le noyau sac-de-mots sert de méthode de référence pour les problèmes de catégorisation de documents par apprentissage.

3.2.5.2 Le classifieur multinomial naïf de Bayes

Dans cette section, nous présentons très brièvement le classifieur multinomial naïf de Bayes [MN98]. Étant donné un ensemble de classe $\mathcal{C}$, un espace d'attributs $\mathcal{F}$, la probabilité qu'un document d appartienne à la classe $c \in \mathcal{C}$ est donnée par la probabilité conditionnelle de Bayes :

$$P(c|d) = \frac{P(c, d)}{P(d)} = \frac{P(d|c)P(c)}{\sum_{c_i} P(d|c_i)P(c_i)} \tag{3.27}$$

Ce classifieur est théoriquement le classifieur idéal. Cependant, la densité de probabilité $P(c, d)$ est inconnue. Par conséquent, il est nécessaire de déterminer les probabilités $P(c)$ et $P(d|c)$. Pour le modèle naïf, ces probabilités sont estimées de "manière naïve" à savoir empiriquement sur la base d'apprentissage. Nous avons ainsi :

$$P(c) = \frac{N_c}{N} \tag{3.28}$$

avec N_c le nombre de document d'apprentissage dans la classe c et N le nombre de document total dans la base d'apprentissage.

Dans le modèle multinomial, un document est dépendant de la fréquence d'occurrence de ses termes contrairement au modèle de Bernouilli [MN98] qui lui assume qu'un document est caractérisé par la présence ou l'absence des termes. En capturant les fréquences d'occurrence,

le modèle multinomial est plus performant que le modèle de Bernouilli. Ainsi, nous nous intéresserons, ici, uniquement au modèle multinomial.

En supposant que les termes d'un document sont générés de manière indépendante et indépendamment de la taille des documents, la probabilité conditionnelle d'avoir un document d connaissant la classe c est donnée par l'hypothèse "naïve" suivante :

$$P(d|c) = \prod_{w_i \in \mathcal{F}} P(w_i|c)^{\mathrm{tf}(w_i,d)} \tag{3.29}$$

avec $\mathrm{tf}(w_i, d)$ la fréquence d'occurrence du terme w_i dans d.

En utilisant l'estimateur de Laplace, qui est une méthodes de lissage parmi d'autres, la probabilité d'avoir un terme w_i conditionnellement à c est :

$$P(w_i|c) = \frac{1 + freq_c(w_i)}{|\mathcal{F}| + \sum_{w \in \mathcal{F}} freq_c(w)} \tag{3.30}$$

avec $|\mathcal{F}|$ la dimension de l'espace d'attributs et $freq_c(w_i)$ le nombre d'occurrence total de w_i dans tous les documents de c.

3.2.5.3 Le classifieur k-NN

Le classifieur k-NN est un modèle simple et performant mais extrêmement coûteux en temps de calcul.
Étant donné une donnée x, la densité autour de x est estimée en déterminant l'ensemble $\mathcal{C}_k$ des k points à proximité de x. x est ensuite affecté à la classe y la plus dense dans cette région (vote majoritaire) soit :

$$y = \mathrm{argmax}_c \, |\{x_i : x_i \in \mathcal{C}_k \text{ et } y_i = c\}| \tag{3.31}$$

où $|A|$ représente le cardinal de l'ensemble A.

La notion de proximité est généralement définie par la distance mais peut aussi être définie par la similarité.

Dans le cas de la catégorisation de texte, notamment dans le cas binaire, une variante des k-NN a été présentée dans [Yan99, Joa02]. La proximité est définie par la similarité basée sur le cosinus. En normalisant toutes les données, la similarité entre deux points x_1 et x_2 est tout simplement définie par $\langle \mathbf{x_1}, \mathbf{x_2} \rangle$. Étant donné x, $\mathcal{C}_k$ est défini par les k points les plus similaires à x. Pour le classement, x est attribué à la classe dont la similarité moyenne avec x est la plus élevée. Dans le cas binaire, pour $y \in \{-1, +1\}$, nous avons l'estimateur empirique de $P(c|x)$ suivant :

$$y = \text{sign}(\frac{\sum_{x_i \in \mathcal{C}_k} y_i \langle \mathbf{x}, \mathbf{x_i} \rangle}{\sum_{x_i \in \mathcal{C}_k} \langle \mathbf{x}, \mathbf{x_i} \rangle}) \tag{3.32}$$

3.2.5.4 **Le classifieur de Rocchio**

Le classifieur de Rocchio[Roc71] a été défini pour le modèle de l'espace vectoriel (VSM). Il est basé sur l'utilisation d'un prototype pour chaque classe. Ainsi, dans le cas binaire le prototype pour les instances positives est la moyenne des instances positives soit (en supposant que toutes les données sont normalisées) :

$$\mathbf{w}_+ = \frac{1}{|\{x_i : x_i \in \mathcal{X} \text{ et } y_i = +1\}|} \sum_{i:y_i=+1} \mathbf{x_i} \tag{3.33}$$

De même pour les instances négatives, le prototype est $\mathbf{w}_-$.

La règle de décision pour classer une donnée x est :

$$y = \text{sign}(\langle \mathbf{w}_+, \mathbf{x} \rangle - \beta \langle \mathbf{w}_-, \mathbf{x} \rangle - b) \tag{3.34}$$

avec β un paramètre permettant d'ajuster l'impact des exemples négatifs et b un réel permettant d'ajuster le seuil de décision.

3.3 **Noyaux pour documents semi-structurés**

Les techniques d'apprentissage statistique sont généralement conçues pour travailler sur des données vectorielles ; chaque mesure est représentée par un ensemble de données numériques de taille fixe. Pendant plusieurs décennies, les recherches en statistique se sont centrées sur des problèmes comme la normalisation des données, le traitement des valeurs manquantes, etc. Depuis une dizaine d'années, sous la pression des applications, nous sommes confrontés à des problème dans lesquels la structure des données porte une information essentielle : textes en langage naturel, documents XML, séquences biologiques, analyse de scènes (images), analyse des réseaux sociaux. Pour gérer ces problèmes, il est nécessaire de trouver un moyen de coder l'information structurelle, ou à défaut de comparer, ou calculer une mesure de similarité, entre deux structures.

A défaut, de nombreux systèmes d'apprentissage numérique des données textuelles utilisent une représentation du texte en sac-de-mot. Ce type de codage, qui a l'avantage de la simplicité, n'utilise que les fréquences d'apparition des mots dans les documents et perd toute information liée à l'ordre des éléments (ordre des mots, structure en paragraphes ou sections, etc).

Dans le cadre du projet InfoM@gic (pôle IMVN Cap Digital), nous avons décidé de travailler sur l'application de ces méthodes à noyaux au traitement de données textuelles structurées, et cette section présente un bref état de l'art dans ce domaine [AV06, AV07]. Nous passons en revue les principaux types de noyaux proposés ces dernières années pour le traitement des séquences et plus généralement des données structurées (arbres, graphes, etc).

3.3.1 Le noyau de convolution

Le noyau de convolution appelé R-noyau [Hau99] permet de définir un cadre général pour les noyaux appliqués aux données structurées telles que les arbres et les graphes.
Les données structurées sont des objets pouvant être décomposés en sous-objets jusqu'à atteindre une unité atomique. L'idée du R-noyau est de calculer la similarité entre deux éléments x et y en effectuant une décomposition de x et de y. Plus formellement, soit x un élément appartenant à un ensemble $\mathcal{X}$ d'éléments structurés de même type, x peut être décomposé en sous éléments $(x_1, \ldots, x_D)$ où x_d peut être un élément structuré ou non appartenant à $\mathcal{X}_d$. Nous définissons la relation binaire :

$$R : \mathcal{X}_1 \times \ldots \times \mathcal{X}_D \to \mathcal{X}$$

qui associe les parties d'un élément x à x et la relation inverse :

$$R^{-1} : \{\overline{x} = (x_1, \ldots, x_D) | R(\overline{x}, x)\}$$

qui retourne pour x l'ensemble de toutes les décompositions possibles. Le noyau de convolution (R-noyau) pour deux éléments x et y de $\mathcal{X}$ est alors :

$$k_R(x, y) = \sum_{\overline{x} \in R^{-1}(x)} \sum_{\overline{y} \in R^{-1}(y)} \prod_{i=1}^{d} k_i(x_i, y_i)$$

avec k_i un noyau calculant la similarité entre les éléments x_i et y_i de même structure i.e. $x_i, y_i \in \mathcal{X}_i$. Il est facile de montrer que si les k_i sont des noyaux valides alors k_R est valide. En effet, si k_i est valide alors sa matrice de Gram est semi-définie positive. Le produit et la somme de matrices semi-définies positives sont des matrices semi-définies positives.

De plus, le R-noyau peut être défini par récurrence lorsque les parties x_i d'un élément x sont de même type de structure ($x_i, x \in \mathcal{X}$). Le critère d'arrêt est défini pour l'élément atomique (par exemple une feuille pour une structure arborescente).
La possibilité de décomposer le calcul de la similarité d'éléments permet de traiter aisément des structures complexes. Toutefois, elle nécessite, en contre-partie, un temps de calcul non négligeable. Ainsi, il est nécessaire de spécialiser ce noyau selon le type de structure afin de réduire la complexité.

3.3.2 Les noyaux pour les séquences de caractères

Les séquences de caractères sont considérées comme faisant partie des données structurées car d'une part une séquence peut être décomposée en sous-partie ainsi les séquences possèdent la propriété des données structurées vue dans le paragraphe précédent et d'autre part les caractères de la séquence sont ordonnés.
Les séquences sont généralement rencontrées dans les domaines liés à la bioinformatique mais aussi dans les documents en langage naturel. En effet, nous pouvons considérer tout un document comme étant une séquence. Ainsi, deux documents peuvent être considérés comme proches

s'ils partagent un nombre important de sous-séquences identiques. Cette approche permet alors de tenir compte des mots composés comme par exemple "économie" et "microéconomie" qui ne peuvent être traités par l'approche en sac de mots. De plus, les mots composés sont très présents dans le domaine de la chimie et les domaines connexes où il n'est pas rare d'avoir des noms de molécules composées.

Bien que cette approche donne de meilleurs résultats que l'approche classique en sac de mots, elle n'en reste pas moins coûteuse en temps de calcul.

3.3.2.1 Le noyau *p-Spectrum*

Le noyau *p-spectrum* (ou n-gram) [LEN02, LSST+02] est le noyau le plus simple pour le traitement de séquences. Il permet d'évaluer le nombre de sous-séquences contiguës de taille p (ou n) que deux documents ont en communs. Plus le nombre de sous-séquences en commun est important et plus la similarité des deux documents sera importante.
Soit l'alphabet Σ, l'espace associé au noyau *p-spectrum* sera de dimension $card(\Sigma)^p$. Le $u^{\text{ème}}$ composant du vecteur $\Phi^p(s)$ associé à la séquence s avec $u \in \Sigma P$ est :

$$\Phi^p_u(s) = card(\{(v_1, v_2)|s = v_1 u v_2\})$$

Avec $v_1 u v_2$ désignant la concaténation des séquences v_1, u et v_2.
Le noyau *p-spectrum* est alors :

$$k_p(s_1, s_2) = \langle \Phi^p(s_1), \Phi^p(s_2) \rangle$$

La complexité de ce noyau est $O(p|s_1||s_2|)$. Toutefois, il est possible de réduire cette complexité à $O(p \times max(|s_1|, |s_2|))$ en utilisant la programmation dynamique et des structures de données appropriées comme les arbres de suffixes [STC04].

Le noyau *p-Spectrum* a été utilisé par [LEN02] pour la classification de séquences de protéines avec l'algorithme SVM (Séparateur à Vaste Marge). Les résultats obtenus sur la base de données SCOP (*Structural Classification of Proteins*) ont montré que la classification par SVM avec le noyau $p - Spectrum$ donne des résultats semblables aux méthodes génératives basées sur les modèles de Markov cachés. Cependant, la méthode SVM avec le noyau Fisher reste la plus performante pour la classification de séquences de protéines.

3.3.2.2 Le noyau *All-SubSequences*

Le noyau *All-SubSequences* [STC04] permet de tenir compte des sous-séquences non contiguës de toutes tailles. Ainsi, la fonction $\Phi^A(s)$ permet de plonger la séquence s dans un espace vectoriel dans lequel le $u^{\text{ème}}$ composant de $\Phi^A(s)$ indique la fréquence d'occurrence de la séquence u dans s. Nous dirons que u est une sous-séquence non contiguë de s, s'il existe un ensemble $I = \{i_1, \ldots, i_{|u|}\}$ tel que $\forall j \in \{2, \ldots, |u|\}, i_{j-1} < i_j$ et $\forall j \in \{1, \ldots, |u|\}, u(j) = s(i_j)$ ($u(j)$ désignant le $j^{\text{ème}}$ élément de u). Nous noterons $s[I] = u$ pour désigner le fait que chaque

élément de $u(j)$ est identique à l'élément $s(i_j)$ avec $i_j \in I$.

$$\Phi_u^A(s) = \sum_{I:s[I]=u} 1$$

D'où :

$$k_A(s,t) = \sum_{(I_1,I_2):s[I_1]=t[I_2]} 1$$

Il est possible de définir ce noyau de manière récursive. En effet, il suffit de remarquer que toute sous-séquence de s contenant le dernier caractère a de s, tel que $s = s'a$, ne peut apparaître dans t qu'entre le premier caractère de t et la dernière occurrence de a dans t. Ainsi, nous avons :

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

L'avantage de ce noyau est qu'il est capable de capturer tous les motifs communs à deux séquences. Le désavantage est que l'espace de projection est de très haute dimension entraînant un temps de calcul important.

3.3.2.3 Le noyau *p-Fixed length SubSequence*

Le noyau *p-Fixed length SubSequence* [STC04] est un compromis entre le noyau *p-Spectrum* et le noyau *All-SubSequences*. Il permet de limiter la recherche de sous-séquences à des sous-séquences non contiguës de taille p. Ainsi, la fonction de projection $\Phi^F(s)$ sera composée des éléments $\Phi_u^A(s)$ tels que $|u| = p$.
De même que précédemment, nous pourrons définir ce noyau par récursion en notant que la récursion sera définie sur la séquence, en retirant à chaque étape le dernier élément de la séquence, mais aussi sur la taille p du motif. En effet, si le dernier caractère du motif a été fixé, le préfixe du motif ne peut être constitué que de $p-1$ éléments.

$$\begin{aligned} k_0(s,t) &= 1 \\ k_p(s,\epsilon) &= 0 \text{ p}our\ p > 0 \\ k_p(s'a,t) &= k_p(s',t) + \sum_{k:t[k]=a} k_{p-1}(s', t[1\ldots k-1]) \end{aligned}$$

3.3.2.4 Le noyau *String Subsequence* (SSK)

L'un des inconvénients des noyaux traitant les sous-séquences non contiguës vus précédemment est qu'ils ne tiennent pas compte de la distance séparant les éléments non contiguë.

En effet, prenons l'exemple de deux séquences "*aaab*" et "*aab*", la séquence "*ab*" est une sous-séquence des deux premières mais elle est plus similaire à la deuxième qu'à la première. Or, les noyaux *All-SubSequences* et *p-Fixed length SubSequence* attribueront la même valeur aux couples ("*aaab*", "*ab*") et ("*aab*", "*ab*").

Le noyau *String Subsequence* [LSST+02] permet de tenir compte de la discontinuité dans le calcul de la similarité en pondérant les séquences en fonction de leur taille. Pour une séquence s, la fonction de projection $\Phi^{SSK}(s)$ sera défini pour tout $u \in \Sigma^n$ par :

$$\Phi_u^{SSK}(s) = \sum_{I:u=s[I]} \lambda^{card(I)}$$

Le noyau SSK, de paramètre n, pour deux séquences s et t est alors :

$$k_{SSK}^n(s,t) = \sum_{u\in\Sigma^n} \sum_{I:u=s[I]} \sum_{J:u=t[J]} \lambda^{card(I)+card(J)}$$

Comme pour les noyaux précédents, en utilisant la programmation dynamique, nous pouvons réduire la complexité à $O(n.|s|.|t|)$.

Des expérimentations ont été menées dans [LSST+02] pour évaluer les noyaux SSK, *p-Spectrum* et le noyau standard sac de mots (BOW - *Bag Of Words*) [Joa02]. La base de données utilisée est la base *Reuters-21578* contenant des documents en langage naturel. L'expérience consistait à effectuer un classement binaire des documents après avoir effectué un apprentissage sur les données prévues à cet effet. Les documents ont été pré-traités en éliminant les mots d'arrêts et les signes de ponctuations. Les résultats ont montré que les *string kernels* sont plus performants que l'approche standard du sac de mots. De plus, le noyau SSK est le plus performant lorsque la valeur de pondération est choisie judicieusement. De même, lorsque la taille p est choisie convenablement, le noyau *p-Spectrum* donne les meilleurs résultats.
Les deux inconvénients pour l'utilisation de ces noyaux sont d'une part le temps de calcul et d'autre part le choix des paramètres qui doivent être fait de manière spécifique à chaque application.

3.3.2.5 Le noyau Séquence marginalisé

Le noyau marginalisé pour les séquences a été introduit par [KT04] afin d'étiqueter des structures complexes tels que les séquences, les arbres et les graphes.
Le problème de l'étiquetage consiste à affecter à une donnée $\mathbf{x} = (x_1, \ldots, x_T)$ un groupe d'étiquettes $\mathbf{y} = (y_1, \ldots, y_T) \in \Sigma_y^T$. Par exemple, en langage naturel $\mathbf{x}$ peut représenter une phrase, et le problème consiste à attribuer à chaque mot de $\mathbf{x}$ une étiquette désignant son groupe grammatical (*Part Of Speech tagging*). La figure 3.3(a) montre un exemple de couple (x, y) où les noeuds noirs représentent les éléments x_i et les noeuds blancs les étiquettes y_i associées à x_i.
Pour résoudre le problème de l'étiquetage, l'approche standard consiste à attribuer à $\mathbf{x}$ la

séquence d'étiquettes $\mathbf{y}$ tel que : $\mathbf{y}$=$argmax_{\mathbf{y}} \sum_i logP(y_i|x_i)$. Il s'agit ici de maximiser la probabilité que la séquence entière soit correcte. Une autre approche consiste à maximiser individuellement la probabilité qu'une étiquette y_i corresponde à l'élément x_i. Soit :

$$y_i = argmax_y P(y_i = y|x_i) = argmax_y \sum_{\mathbf{y}:y_i=y} P(\mathbf{y}|\mathbf{x})$$

Partant de cette dernière approche, [KT04] propose une méthode pour étiqueter une séquence. Cette méthode se base sur l'utilisation d'un perceptron à noyau [STC04].
L'idée étant d'étiqueter les éléments individuellement, une séquence $(\mathbf{x}, \mathbf{y})$ est décomposée en triplés $(\mathbf{x}, t, y_t)$ où y_t est le $t^{ème}$ élément de $\mathbf{y}$. Pour cela, le perceptron à noyau est entraîné avec des séquences étiquetées auxquelles ont été rajoutés des exemples négatifs. Les exemples négatifs sont obtenus pour chaque couple $(\mathbf{x}, \mathbf{y})$ en modifiant les valeurs de $\mathbf{y}$. Ainsi pour ce couple nous obtenons un ensemble de positifs $\{(\mathbf{x}, i, y_i) : 1 \leq i \leq dim(\mathbf{x})\}$ et un ensemble de négatifs $\{(\mathbf{x}, i, z) : 1 \leq i \leq dim(\mathbf{x}) \ \forall z \in \Sigma_y\}$. Après apprentissage, nous affectons à un élément u_t d'une séquence $\mathbf{u}$ l'étiquette y qui maximise le score calculé par le perceptron avec un noyau marginalisé.

Le noyau marginalisé proposé est le suivant :

$$k(\mathbf{x}, \mathbf{x}', \tau, t, y_\tau, y'_t) = \sum_{\mathbf{z}:z_\tau=y_\tau} \sum_{\mathbf{z}':z'_t=y'_t} P(\mathbf{z}|\mathbf{x})P(\mathbf{z}'|\mathbf{x}') \left\langle \Phi(\mathbf{x}, \mathbf{z}; \tau), \Phi(\mathbf{x}', \mathbf{z}'; t) \right\rangle \tag{3.35}$$

y_τ et y'_t étant les étiquettes respectives des éléments x_τ et x'_t ; $\phi_f(\mathbf{x}, \mathbf{y}; t)$ indiquant la fréquence d'occurrence de f dans $(\mathbf{x}, \mathbf{y})$ incluant la $t^{\text{ème}}$ position.
La combinatoire induit par le noyau de l'équation 3.35 peut-être diminuée en effectuant une recherche bidirectionnelle au sein d'une séquence. En effet, la composante $\phi_f(\mathbf{x}, \mathbf{y}; t)$ du vecteur $\Phi(\mathbf{x}, \mathbf{y}; t)$ indique le nombre d'occurrence de f dans $(\mathbf{x}, \mathbf{y})$ incluant la $t^{\text{ème}}$ position de $(\mathbf{x}, \mathbf{y})$ $((x_t, y_t))$. Il est alors possible de décomposer f en f_u et f_d tel que $((x_t, y_t))$ soit le dernier élément de f_u et le premier élément de f_d. Le calcul se limitera alors à évaluer les deux ensembles $F_u = \{(x_i, x_{i+1}, \ldots, x_t) | 1 \leq i \leq t\}$ et $F_d = \{(x_t, x_{t+1}, \ldots, x_k) | t \leq k \leq dim(\mathbf{x})\}$. Les autres composantes de l'espace seront obtenues par combinaison de F_u et de F_d i.e. $F = F_u \times F_d$.
La figure 3.3 illustre ce principe : le couple de séquences (a) peut être obtenu en combinant (b) et (c).

Afin d'effectuer la décomposition d'une séquence en fonction de la position t, nous supposerons que :

$$P(\mathbf{y}|\mathbf{x}) = \prod_t P(y_t|x_t)$$

On posant $\mathbf{x}_u(t) = (x_0, \ldots, x_t)$ et $\mathbf{x}_d(t) = (x_t, \ldots, x_T)$, de même pour $\mathbf{y}_u(t)$ et $\mathbf{y}_d(t)$, nous obtenons :

$$P(\mathbf{y}|\mathbf{x}) = P(\mathbf{y}_u(t)|\mathbf{x}_u(t)).\frac{P(y_t|x_t)}{(P(y_t|x_t))^2}.P(\mathbf{y}_d(t)|\mathbf{x}_d(t))$$

En décomposant, l'équation 3.35 nous obtenons :

$$k(\mathbf{x}, \mathbf{x}', \tau, t, y_\tau, y'_t) = k_u(\mathbf{x}, \mathbf{x}', \tau, t).k_p(\mathbf{x}, \mathbf{x}', \tau, t, y_\tau, y'_t).k_d(\mathbf{x}, \mathbf{x}', \tau, t)$$

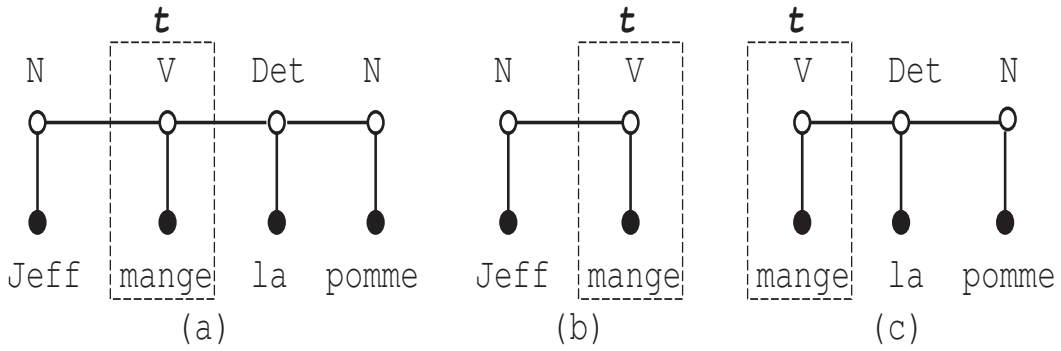


Fig. 3.3 – (a) Exemple de couple de séquences (x, y). (b) sous-séquences de (a) où les x_i avec $i > t$ ont été éliminés. (c) sous-séquences de (a) obtenu en éliminant les x_i de (a) pour $i < t$.

avec :

$$\begin{aligned}
k_u(\mathbf{x}, \mathbf{x}', \tau, t) &= \sum_{\mathbf{y}_u(\tau)} \sum_{\mathbf{y'}_u(t)} P(\mathbf{y}_u(\tau)|\mathbf{x}_u(\tau)) P(\mathbf{y'}_u(t)|\mathbf{x'}_u(t)) \\
&\quad . \left\langle \Phi(\mathbf{x}_u(\tau), \mathbf{y}_u(\tau); \tau), \Phi(\mathbf{x'}_u(t), \mathbf{y'}_u(t); t) \right\rangle \\
k_d(\mathbf{x}, \mathbf{x}', \tau, t) &= \sum_{\mathbf{y}_d(\tau)} \sum_{\mathbf{y'}_d(t)} P(\mathbf{y}_d(\tau)|\mathbf{x}_d(\tau)) P(\mathbf{y'}_d(t)|\mathbf{x'}_d(t)) \\
&\quad . \left\langle \Phi(\mathbf{x}_d(\tau), \mathbf{y}_d(\tau); \tau), \Phi(\mathbf{x'}_d(t), \mathbf{y'}_d(t); t) \right\rangle \\
k_p(\mathbf{x}, \mathbf{x}', \tau, t, y_\tau, y'_t) &= \frac{P(y_\tau|x_\tau) P(y'_t|x'_t) . \langle \Phi(x_\tau, y_\tau; \tau), \Phi(x'_t, y'_t; t) \rangle}{(\sum_{z_\tau} \sum_{z'_t} P(z_\tau|x_\tau) P(z'_t|x'_t) . \langle \Phi(x_\tau, z_\tau; \tau), \Phi(x'_t, z'_t; t) \rangle)^2}
\end{aligned} \tag{3.36}$$

La complexité pour l'évaluation de ces noyaux peut être ramenée à $O(T.T')$ avec T et T' la taille des séquences en utilisant la programmation dynamique. Nous obtenons alors les noyaux suivants :

$$\begin{aligned}
k_u(\mathbf{x}, \mathbf{x}', \tau, t) &= \begin{cases} 0 \text{ si } \tau = 0 \text{ ou } t = 0 \\ c^2 k(x_\tau, x'_t)(k_u(\mathbf{x}, \mathbf{x}', \tau - 1, t - 1) + 1) \end{cases} \\
k_d(\mathbf{x}, \mathbf{x}', \tau, t) &= \begin{cases} 0 \text{ si } \tau > dim(\mathbf{x}) \text{ ou } t > dim(\mathbf{x}') \\ c^2 k(x_\tau, x'_t)(k_d(\mathbf{x}, \mathbf{x}', \tau + 1, t + 1) + 1) \end{cases} \\
k(x_\tau, x'_t) &= \begin{cases} 0 \text{ si } x_\tau \neq x'_t \\ \sum_y P(y|x_\tau) P(y|x'_t) \end{cases} \\
k_p(\mathbf{x}, \mathbf{x}', \tau, t, y_\tau, y'_t) &= \begin{cases} 0 \text{ si } (x_\tau, y_\tau) \neq (x'_t, y'_t) \\ \frac{c^2 P(y_\tau|x_\tau) P(y'_t|x'_t)}{(c^2 k(x_\tau, x'_t)^2)} \end{cases}
\end{aligned}$$

La constante c est utilisée pour pondérer les termes ϕ_f selon la taille de f. Il est aussi possible de permettre des discontinuités dans les sous-séquences comme dans le cas du noyau *String Subsequence*. Il suffit alors, juste, de modifier k_u et k_d (voir [KT04] pour plus de détails).

Ce noyau, combiné au noyau polynomial de degré deux, a été utilisé avec un perceptron pour résoudre un problème de reconnaissance d'entités nommées et un problème d'extraction d'information. La loi uniforme est utilisé pour modéliser $P(y_t|x_t)$. Les expériences ont été menées en utilisant la validation croisée à 3 blocs.
En outre, pour chaque expérience le noyau marginalisé est comparé à un perceptron utilisant le modèle de markov caché [Col02]. L'idée de base de cet algorithme est d'utiliser l'algorithme de Viterbi sur un modèle de markov caché afin d'attribuer la meilleure séquence d'étiquettes à une séquence de terme. Le perceptron est utilisé pour calculer un score à une séquence étiquetée. Ce score sera utilisé par l'algorithme de Viterbi pour trouver la séquence d'étiquettes optimale.

Pour la reconnaissance d'entités nommées, les données utilisées sont un sous-ensemble d'un corpus espagnol fourni par CoNLL2002. Ce corpus est composé de 300 phrases comprenant au total 8541 termes. L'objectif est d'attribuer à chaque terme un des neuf labels désignant le type d'entité nommée (un des neufs labels correspond au type "non-entité nommée"). Les résultats montrent que le noyau marginalisé à un taux de reconnaissance, tant au niveau de la précision que du rappel, supérieur à celui du modèle de markov caché.

La deuxième expérience consistait à extraire des informations concernant l'utilisation de produits. À partir d'une base de 184 phrases (soit 3570 termes) en japonais, l'objectif est de reconnaître le nom du produit, le vendeur, le nombre de produit acheté, les raisons de l'achat etc. Ainsi, il s'agit d'attribuer à chaque terme une des 12 étiquettes correspondants aux informations citées.
Les termes ont été annotés en effectuant une analyse lexicale. En outre, un noyau marginalisé sur les arbres (voir la section sur les arbres) a été utilisé. Pour ce noyau, les données ont été structurées en arbre lexical de dépendance représentant la structure linguistique de la phrase en terme de dépendance entre les mots.
Les expériences montrent que les noyaux marginalisés sont plus performants que le perceptron utilisant le modèle de markov caché. De plus, le noyau marginalisé sur les arbres obtient de meilleurs résultats que le noyau sur les séquences. Ce résultat peut être expliqué par le fait que le noyau sur les arbres tire avantage de l'information structurelle contrairement au noyau sur les séquences.

En outre, d'autres méthodes basées sur les noyaux pour séquences ont été proposées pour l'étiquetage de séquences [CMC+06].

3.3.3 Les noyaux pour les arbres

Les arbres sont des structures de données permettant de représenter efficacement des données organisées de manière hiérarchique. Ainsi, ils sont communément utilisés dans de nombreux domaines.
La majorité des documents structurés et semi-structurés, tels que les documents XML, sont représentés de manière arborescente. Ainsi, il peut être intéressant de tenir compte de cette

structure dans l'évaluation des critères de similarités entre ces différents documents.

3.3.3.1 **le noyau** *Tree kernel*

Le noyau *Tree Kernel* (appelé aussi *parse tree kernel*) [CD02] a été défini pour le calcul de similarité entre les arbres grammaticaux (ou arbres syntaxiques). Un arbre syntaxique est obtenu à partir d'une phrase en la décomposant en groupe grammatical. La figure 3.4 montre l'arbre syntaxique associé à la phrase "Jeff mange la pomme".

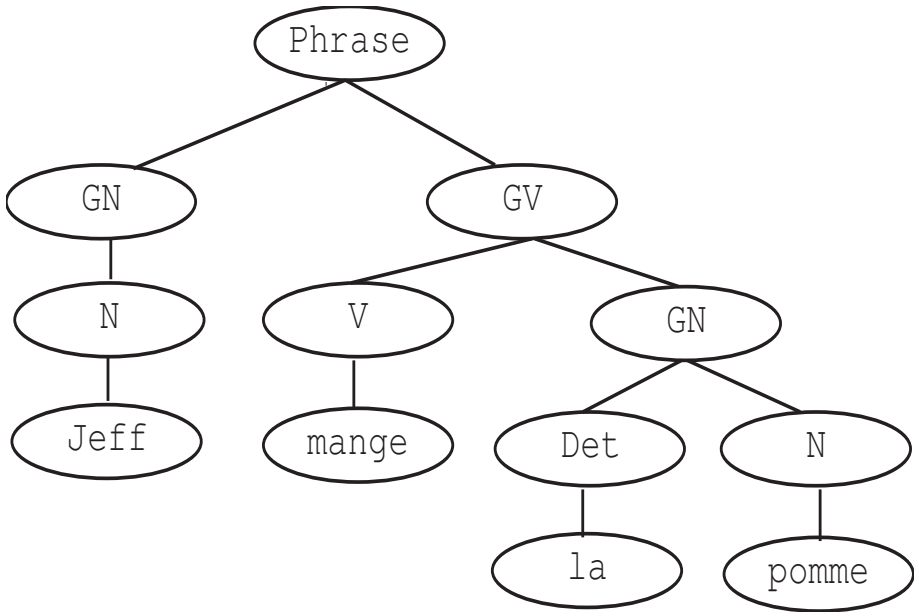


Fig. 3.4 – Arbre syntaxique pour la phrase "Jeff mange la pomme"

Définition 3.3.1 (arbre propre)**.** *Un arbre propre est un arbre ayant une racine et au moins un noeud fils.*

Définition 3.3.2 (Sous-arbre complet)**.** *Un arbre S est dit sous-arbre complet d'un arbre T si et seulement si il existe un noeud n tel que l'arbre induit par n (de racine n) est égal à S. Nous noterons $\tau_T(n)$ l'arbre complet induit par le noeud n de T.*

Définition 3.3.3 (Sous-arbre co-enraciné)**.** *Un arbre S est dit sous-arbre co-enraciné d'un arbre propre T si et seulement si les propriétés suivantes sont vérifiées :*

1. *S est un arbre propre,*
2. *la racine de S ($rac(S)$) est identique à la racine de T ($rac(T)$) (si S et T sont des arbres dont les noeuds sont étiquetés alors les étiquettes de $rac(S)$ et de $rac(T)$ doivent être identiques),*

3. $\forall i, fils_i(rac(S)) = fils_i(rac(T))$,

4. S *peut être obtenu à partir de* T *en supprimant des sous-arbres de* $fils_i(rac(T))$.

La notion de sous-arbre co-enraciné permet de garantir la consistance des règles grammaticales. Par exemple, pour l'arbre T de la figure 3.4, il existe des arbres co-enracinés de T qui produisent : "N V la N", "Jeff mange GN", "GN V GN", ... Toutefois, il n'existe pas d'arbres co-enracinés de T produisant "Jeff mange Det", "GN mange N", ...

Soit Γ l'ensemble de tous les arbres propres possibles, un arbre T peut être plongé, par une fonction Φ^r, dans un espace vectoriel de caractéristiques (*feature-space*). Le $u^{\text{ème}}$ composant de Φ^r, associé à $S_u \in \Gamma$, donne le nombre de noeud n de T tel que S_u est un sous-arbre co-enraciné de $\tau_T(n)$ ($Rfreq_T(S_u)$), soit :

$$\Phi^r_{S_u}(T) = Rfreq_T(S_u) = \sum_{n \in T} I_{S_u}(\tau_T(n)) \tag{3.37}$$

Avec :

$$I_{S_u}(\tau_T(n)) = \begin{cases} 1 & \text{si l'arbre } S_u \text{ est un sous-arbre co-enraciné de } \tau_T(n)\,, \\ 0 & \text{sinon}\,. \end{cases}$$

Le noyau permettant de calculer la similarité entre deux arbres T_1 et T_2 est :

$$\begin{aligned} k_{tree}(T_1, T_2) &= \langle \Phi^r(T_1), \Phi^r(T_2) \rangle \\ &= \sum_{S_u \in \Gamma} \Phi^r_{S_u}(T_1).\Phi^r_{S_u}(T_2) \\ &= \sum_{S_u \in \Gamma} \Big(\sum_{n_1 \in T_1} I_{S_u}(\tau_{T_1}(n_1)) \Big).\Big(\sum_{n_2 \in T_2} I_{S_u}(\tau_{T_2}(n_2)) \Big) \\ &= \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} \sum_{S_u \in \Gamma} I_{S_u}(\tau_{T_1}(n_1)).I_{S_u}(\tau_{T_2}(n_2)) \\ &= \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} k^r_{tree}(\tau_{T_1}(n_1), \tau_{T_2}(n_2)) \end{aligned} \tag{3.38}$$

$k^r_{tree}(T_1, T_2)$ indique le nombre de sous-arbres co-enracinés qu'ont en commun les arbres T_1 et T_2. Cette fonction retourne 0 si 1) les racines sont différentes, ou 2) si le nombre de fils de T_1 et de T_2 ne correspondent pas ou 3) si $\exists i, fils_i(T_1) \neq fils_i(T_2)$. Dans les autres cas, nous pouvons définir $k^r_{tree}(T_1, T_2)$ par récurrence. En effet, cette fonction sera égale au produit des nombres de sous-arbres co-enracinés communs à chacun des fils de T_1 et de T_2. Il est à noter que si $k^r_{tree}(T_1, T_2) \neq 0$ mais que $k^r_{tree}(\tau_{T_1}(fils_i(rac(T_1)), fils_i(rac(T_2)))) = 0$, il existe un unique sous-arbre co-enraciné commun à T_1 et T_2 pour la partie du sous-arbre induit par $fils_i$.
On peut ainsi définir $k^r_{tree}(T_1, T_2)$ dans le cas non nul :

$$k^r_{tree}(T_1, T_2) = \prod_i (k^r_{tree}(\tau_{T_1}(fils_i(rac(T_1))), \tau_{T_2}(fils_i(rac(T_2)))) + 1)$$

La complexité temporelle du noyau $k_{tree}(T_1, T_2)$ est $O(|T_1||T_2|)$ [CD02] avec $|T|$ le nombre de noeuds dans T.
Collins et al. ont utilisé ce noyau pour associer à une phrase, l'arbre syntaxique le plus plausible (*parsing*) [CD02]. La décomposition d'une phrase en arbre syntaxique est un problème difficile. En effet, l'ambiguïté sous-jacente au langage naturel entraîne plusieurs décompositions possibles pour une même phrase. L'objectif proposé par Collins et al. est de sélectionner l'arbre le plus probable par une approche discriminante.
Soit F une fonction permettant de générer un ensemble d'arbres syntaxiques pour une phrase, les données sont représentées par un doublet $(s, F(s))$. Pour l'ensemble d'apprentissage, l'arbre syntaxique correct pour chaque s est connu dans $F(s)$. Un séparateur est alors "appris" en utilisant un perceptron. Pour une phrase s, nous lui associons l'arbre T_{s_i} de $F(s)$ tel que :

$$T_{s_i} = argmax_{T \in F(s)}(w^*.\Phi^r(T))$$

Avec w^* le vecteur optimal associé à :

$$w = \sum_{i,j} \alpha_{i,j}(\Phi^r(T_{s_{i1}}) - \Phi^r(T_{s_{ij}}))$$

Avec s_i une phrase d'apprentissage, $T_{s_{i1}} \in F(s_i)$ l'arbre syntaxique correct de s_i et $T_{s_{ij}} \in F(s_i)$.
Les expériences sur le corpus *Penn treebank ATIS*, qui est un corpus anglais annoté sous forme arborescente, ont montré que l'utilisation de cette méthode améliore de près de 4% les résultats obtenus par une méthode conventionnelle stochastique (*Probabilistic Context Free Grammar*).

3.3.3.2 le noyau *Tree kernel* généralisé

Le noyau *Tree Kernel* a été essentiellement développé pour traiter des arbres spécifiques tels que les arbres syntaxiques. Le *Tree Kernel* se base donc sur les propriétés :

1. les descendants d'un noeud n'ont jamais les mêmes étiquettes que les noeuds ancêtres
2. le noyau utilise la définition de sous-arbre co-enraciné pour calculer la similitude entre deux arbres.

Une généralisation de ce noyau a été proposée dans [KK02] pour traiter des arbres complexes tels que les arbres XML et HTML. Toutefois, nous imposons que l'arbre soit étiqueté et ordonné (tel que c'était le cas pour les arbres syntaxiques).
Le cadre général du noyau *Tree Kernel* reste valide. En effet, pour généraliser le noyau, il suffit de modifier la fonction $I_S(T)$ et de changer le noyau spécifique $k^r_{tree}(T_1, T_2)$. Dans le cas d'un arbre quelconque étiqueté et ordonné T, $I_S(T)$ retournera la fréquence d'occurrence du sous-arbre S dans T.

Définition 3.3.4 (Sous-arbre)**.** *Un arbre S (possédant au moins un noeud) est un sous-arbre de T si et seulement si il existe un noeud n de T tel que $l(n) = l(rac(S))$ ($l(n)$ correspondant à l'étiquette du noeud n) et une liste ordonnée d'indexes $\{j_1, \ldots, j_k\}$ tel que $\forall i, l(fils_i(rac(S))) = l(fils_{j_i}(\tau_T(n)))$ avec $i < j_i$ et $\tau_S(fils_i(rac(S)))$ soit, soit une feuille soit un sous-arbre de $\tau_T(fils_{j_i}(\tau_T(n)))$ partageant la même racine.*

Étant donnée cette définition de sous-arbre, le noyau $k^r_{tree}(T_1, T_2)$ peut être défini comme étant la fonction qui retourne le nombre de sous-arbres commun à T_1 et T_2 avec pour racine $rac(T_1)$, en tenant compte de la fréquence d'occurrence dans chaque arbre. Autrement dit, il s'agit de la somme, pour chaque sous-arbre possible S de racine $rac(T_1)$, des produits des nombres d'occurrences de S dans T_1 et dans T_2.
Les arbres étant ordonnés, le noyau sur T_1 et T_2 peut être calculé en introduisant une récurrence sur le nombre de fils de T_1 $(nf(rac(T_1)))$ et le nombre de fils de T_2. Ainsi, la fonction $S_{T_1,T_2}(i,j)$ est introduite pour calculer $k^r_{tree}(T_{1i}, T_{2j})$ tel que T_{1i} est le sous-arbre de T_1 obtenu en supprimant tous les fils d'index supérieurs à i, ainsi que leurs descendants, de même pour T_{2j}.

$$k^r_{tree}(T_1, T_2) = \begin{cases} 0 \ si \ l(rac(T_1)) \neq l(rac(T_2)) \\ S_{T_1,T_2}(nf(rac(T_1)), nf(rac(T_2))) \end{cases} \tag{3.39}$$

avec

$$\begin{aligned} S_{T_1,T_2}(0,0) &= S_{T_1,T_2}(i,0) = S_{T_1,T_2}(0,j) = 1 \\ S_{T_1,T_2}(i,j) &= S_{T_1,T_2}(i-1,j) + S_{T_1,T_2}(i,j-1) \\ &\quad - S_{T_1,T_2}(i-1,j-1) \\ &\quad + S_{T_1,T_2}(i-1,j-1).k^r_{tree}(\tau_{T_1}(fils_i(rac(T_1))), \tau_{t_2}(fils_j(rac(T_2)))) \end{aligned} \tag{3.40}$$

Le noyau peut être généralisé en introduisant deux concepts : la similarité entre les étiquettes et les sous-arbres non-contigus [KK02].
Soit Σ l'ensemble de toutes les étiquettes et $f : \Sigma \times \Sigma \rightarrow [0,1]$ indiquant un score de "mutation" entre deux étiquettes tel que $f(e,a)$ indique la probabilité d'acceptation de la mutation de l'étiquette a vers e, la fonction de similarité entre deux étiquettes de deux noeuds n_1 et n_2 est :

$$Sim(l(n_1), l(n_2)) = \sum_{a \in \Sigma} f(l(n_1), a).f(l(n_2), a)$$

En introduisant la fonction de similarité dans l'équation 3.39, nous obtenons :

$$k^r_{tree}(T_1, T_2) = Sim(l(n_1), l(n_2)).S_{T_1,T_2}(nf(rac(T_1)), nf(rac(T_2))) \tag{3.41}$$

Le deuxième cadre de généralisation est l'intégration, au niveau du noyau, de la notion d'élasticité des sous-arbres. Ainsi, la définition de sous-arbres se voit élargie en permettant la non contiguïté au niveau des noeuds d'un chemin. En effet, il n'est plus nécessaire qu'un chemin d'un sous-arbre apparaisse de manière contiguë dans un arbre. Cependant, la contrainte d'arbre ordonné reste valable.

Définition 3.3.5 (Sous-arbre non contigu)**.** *Un arbre S (possédant au moins un noeud) est un sous-arbre de T si et seulement si il existe un noeud n de T tel que $l(n) = l(rac(S))$ et une liste ordonnée d'indexes $\{j_1, \ldots, j_k\}$ tel que $\forall i, \tau_S(fils_i(rac(S)))$ soit un sous arbre de $\tau_T(fils_{j_i}(\tau_T(n)))$.*

Afin de prendre en compte la définition de sous-arbre non-contigu, il est nécessaire de généraliser la formule de $k^r_{tree}(T_1, T_2)$ (équations 3.39 et 3.41). En effet, cette formule retourne une valeur nulle (ou faible selon la similarité) si les étiquettes des racines sont différentes. En d'autre terme, tout sous-arbre commun à T_1 et T_2 doit être enraciné aux noeuds racines de T_1 et T_2 impliquant ainsi que les arbres possèdent la même racine. Or, dans le cas des sous-arbres non-contigus, un sous-arbre commun à T_1 et T_2 peut être construit par combinaison (ordonnée) à partir de sous-arbres enracinés à n'importe quelles noeuds descendants de T_1 et T_2.

Soit k^r_{old} le noyau définit par l'équation 3.39, le noyau spécifique pour les arbres élastiques utilisés par le noyau *Tree Kernel* (équation 3.38) est :

$$k^r_{tree}(T_1, T_2) = \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} k^r_{old}(\tau_{T_1}(n_1), \tau_{T_2}(n_2))$$

Cette formule peut être calculée efficacement de manière récursive :

$$\begin{aligned} k^r_{tree}(T_1, T_2) &= k^r_{old}(T_1, T_2) + \sum_{n_i = fils_i(T_1)} k^r_{tree}(\tau_{T_1}(n_i), T_2) \\ &\quad + \sum_{n_j = fils_j(T_2)} k^r_{tree}(T_1, \tau_{T_2}(n_j)) \\ &\quad - \sum_{n_i = fils_i(T_1)} \sum_{n_j = fils_j(T_2)} k^r_{tree}(\tau_{T_1}(n_i), \tau_{T_2}(n_j)) \end{aligned}$$

De même que dans le cas du noyau *parse tree kernel* vu dans la section précédente, la complexité du noyau *tree kernel* est $O(|T_1|.|T_2|)$ dans les différents cas de généralisation.

[KK02] ont utilisé les noyaux généralisés élastiques et non élastiques, sans tenir compte des mutations d'étiquettes, pour la classification et l'extraction d'information à partir de documents HTML. Les performances ont été évaluées en utilisant la méthode de validation croisée *leave-one-out*. L'apprentissage a été effectué en utilisant l'algorithme du perceptron "kernelisé".

Pour la classification de documents, une base de données comprenant 30 documents HTML en japonais et 30 documents HTML en anglais a été construite en extrayant les documents aléatoirement sur les sites américains et japonais d'IBM. La classification devant être structurelle, seules les balises HTML ont été préservées, éliminant ainsi les attributs et les données. De plus, les noyaux sur arbres ont été combinés avec le noyau polynomial. Les résultats ont montré que le noyau non élastique était de 12% plus performant que le noyau élastique en atteignant près de 80% de bon classement. Ces résultats ont été obtenus avec un noyau polynomial d'ordre 4 (resp. 3).

L'extraction d'information consiste à apprendre et à reconnaître une information précise dans des documents HTML. Il s'agit ici du marquage des noeuds pertinents. Le problème du marquage consiste à apprendre à partir d'arbres correctement marqués puis à marquer les noeuds pertinents des arbres non traités. Ce problème peut être ramené à un problème de classification en effectuant une transformation du marquage. En effet, pour un noeud marqué, nous

insérons entre le noeud concerné et le noeud père, un noeud portant une étiquette appropriée pour signaler le marquage. Puis, un ensemble d'arbres négatifs est générés à partir des arbres corrects en retirant le marquage et en les plaçant sur des noeuds non initialement marqués. Un apprentissage peut ensuite être effectué sur ces données. Pour le marquage sur un arbre, les auteurs effectuent pour chacun de ses noeuds un marquage puis le classent.
Pour l'expérimentation, une base a été créée à partir de 54 pages HTML extraites d'un catalogue de vente d'ordinateurs portables d'IBM Japon. L'objectif de l'expérience était de retrouver l'image de l'ordinateur à vendre. Pour cela, les noeuds contenants les images pertinentes ont été marqués et les données textuelles présentes dans les pages ont été éliminées. Les résultats ont montré que le noyau élastique a permis d'extraire l'information avec une précision de 99.3% et un rappel de 68.6% contre une précision de 11.9% et un rappel de 79.6% pour le noyau non élastique. Ces résultats ont été obtenus sans la combinaison avec le noyau polynomial. En effet, ce dernier n'a pas permis d'améliorer les résultats.

3.3.3.3 le noyau *Tree kernel* marginalisé

Le noyau marginalisé pour les arbres a été introduit par [KT04] pour répondre aux problèmes d'étiquetages (voir la section sur le noyau marginalisé sur les séquences).
L'objectif de ce noyau est de permettre de calculer la similarité entre deux arbres étiquetés. Un arbre étiqueté étant simplement un arbre où chaque noeud représente un élément observable (un terme) et à chaque noeud est associé une étiquette. L'avantage de résoudre un problème d'étiquetage en utilisant un modèle de donnée arborescent, plutôt que séquentiel, est qu'il est possible d'exploiter l'information structurelle pour améliorer la discrimination.

Le noyau marginalisé sur les arbres est obtenu en intégrant le noyau *Tree kernel* généralisé dans le cadre théorique du noyau marginalisé définit par la série d'équation 3.36.
Ainsi, $k_d(T_1, T_2, \tau, t)$ est le noyau ne prenant en compte que les sous-arbres ayant la même racine $rac(T_{1_\tau})$ (T_{1_τ} indique ici le sous-arbre de T_1 induit par le noeud d'index τ). De même, $k_u(T_1, T_2, \tau, t)$ ne prend en compte que les sous-arbres ayant une feuille correspondant au noeud d'index τ de T_1 ($rac(T_{1_\tau})$).

Le calcul de k_d se base sur les équations 3.39 et 3.40. L'équation 3.40 calcul la somme des contributions des sous-arbres communs à T_1 et T_2 en explorant à chaque niveau les fils de droite à gauche. En modifiant cette équation, nous obtenons :

$$\begin{aligned}
S_F(T_1, T_2, \tau, t, 0, 0) &= S_F(T_1, T_2, \tau, t, i, 0) = S_F(T_1, T_2, \tau, t, 0, j) = 1 \\
S_F(T_1, T_2, \tau, t, i, j) &= S_F(T_1, T_2, \tau, t, i-1, j) + S_F(T_1, T_2, \tau, t, i, j-1) \\
&\quad - S_F(T_1, T_2, \tau, t, i-1, j-1) \\
&\quad + S_F(T_1, T_2, \tau, t, i-1, j-1).k_d(T_1, T_2, ch(T_1, \tau, i), ch(T_2, t, j))
\end{aligned}$$

Avec $ch(T_1, \tau, i)$ l'index, dans T_1, du $i^{\text{ème}}$ fils de la racine de T_{1_τ}.
Le noyau k_d devient alors :

$$k_d(T_1, T_2, \tau, t) = c^2 k(T_{1_\tau}, T_{2_t}).(1 + S_F(T_1, T_2, \tau, t, nf(rac(T_{1_\tau})), nf(rac(T_{2_t}))))$$

Pour k_u, le calcul s'effectue de la feuille vers la racine. Ainsi, la fonction $pa(T_1, \tau)$, qui retournera l'index du père du $\tau^{\text{ème}}$ noeud dans T_1, est utilisée. En outre, il faut aussi tenir compte des frères gauches et des frères droits du $\tau^{\text{ème}}$ noeud de T_1. Pour la contribution des frères gauches, la fonction S_F pourra être utilisée. Quant aux frères droits, il faudra les explorer de la gauche vers la droite. La fonction $n = chID(T_1, \tau)$ est utilisée pour indiquer que le noeud d'index τ est le $n^{\text{ème}}$ fils de son père. Une fonction correspondant à S_F est, donc, définie :

$$\begin{aligned}
S_B(T_1, T_2, \tau, t, i, j) &= 1 \text{ si } i \geq nf(rac(T_{1_\tau})) \text{ ou si } j \geq nf(rac(T_{2_t})) \\
S_B(T_1, T_2, \tau, t, i, j) &= S_B(T_1, T_2, \tau, t, i+1, j) + S_B(T_1, T_2, \tau, t, i, j+1) \\
&\quad - S_B(T_1, T_2, \tau, t, i+1, j+1) \\
&\quad + S_B(T_1, T_2, \tau, t, i+1, j+1).k_d(T_1, T_2, ch(T_1, \tau, i), ch(T_2, t, j))
\end{aligned}$$

L'expression de k_u est :

$$\begin{aligned}
k_u(T_1, T_2, \tau, t) &= c^2 k(T_{1_\tau}, T_{2_t}).(1 + k_u(T_1, T_2, \tau, t)) \\
&\quad . S_F(pa(T_1, \tau), pa(T_2, t), \tau, t, chID(T_1, \tau) - 1, chID(T_2, t) - 1) \\
&\quad . S_B(pa(T_1, \tau), pa(T_2, t), \tau, t, chID(T_1, \tau) + 1, chID(T_2, t) + 1)
\end{aligned}$$

Les expériences menées sur ce noyau sont décrites dans la section sur le noyau marginalisé pour les séquences.

3.3.4 Les noyaux pour les graphes

Le graphe est une structure de donnée très utilisée dans le domaine informatique pour modéliser des informations structurées complexes. En effet, les séquences et les arbres vus précédemment sont, en réalité, des graphes acycliques orientés. De plus, dans le cas de la séquence, le graphe est de degré maximum 1.
Nous définirons un graphe étiqueté G par le triplé (V, E, σ) où V est l'ensemble des noeuds de G, σ l'ensemble des étiquettes tel que σ_i représente l'étiquette du noeud i (il est aussi possible d'étiqueter les arcs : nous noterons $\sigma_{(i,j)}$ l'étiquette de l'arc reliant le noeud i à j) et E, une matrice d'adjacence tel que $E_{ij} = 1$ si et seulement si il existe un arc reliant le noeud i au noeud j (afin de simplifier l'écriture nous utiliserons la même notation i, j pour désigner les indexes dans E que pour designer les noeuds de V). L'une des propriétés de la matrice d'adjacence est que $[E^n]_{ij}$ indique le nombre de chemin de longueur n reliant le noeud i au noeud j.

La conception d'un noyau nécessite une définition de la similarité entre deux graphes. Pour cela, il existe deux approches standards [Gär03, GLS06]. La première consiste à déterminer si les deux graphes sont isomorphes (ils ne se distinguent que par l'ordre des noeuds) ou à déterminer le nombre de sous-graphes isomorphes communs. Cependant, il est connu que ce problème est fortement combinatoire.
La deuxième approche consiste à projeter le graphe G dans un espace vectoriel où chaque dimension est indexée par un graphe H tel que la valeur de la projection de G sur cet axe représente la fréquence d'occurrence de H, en tant que sous-graphe, dans G. Il est alors possible de concevoir un noyau qui identifie certaines propriétés dans les sous-graphes H. En particulier, il est possible de concevoir un noyau qui effectue le produit scalaire dans l'espace vectoriel en se limitant aux sous-graphes qui sont des chemins hamiltoniens (H est un chemin hamiltonien de G si et seulement si H est un sous-graphe de G et si H est de même ordre que G i.e. H contient tous les noeuds de G exactement une fois). De même que pour la première approche, le problème du chemin hamiltonien est NP-difficile.

Afin de réduire la complexité dans l'évaluation de la similarité d'autres approches ont été explorées. L'approche la plus répandue consiste à calculer la similarité en se basant sur les chemins parcourus [GDR03, GLS06, KI02, KTI03]. L'un des problèmes principaux de cette approche est qu'il existe une infinité de chemins possibles dès lors qu'il existe un cycle dans le graphe. Nous avons alors le noyau :

$$k(G,G') = \lim_{N\to\infty} \sum_{i=1}^{N} \sum_{p\in P_i(G)} \sum_{p'\in P_i(G')} \lambda_i . k_p(p,p') \tag{3.42}$$

Avec $P_l(G)$ l'ensemble des chemins de G de longueur l, λ_l un réel pondérant les chemins de longueur l (on fixera $\lambda_l = \lambda^l$) et k_p un noyau défini sur les chemins. Il existe plusieurs façons de définir k_p selon qu'on veuille tenir compte des étiquettes sur les noeuds, sur les arcs ou encore permettre des discontinuités (*gap*).

Dans [GDR03, GLS06], k_p est défini sur des chemins contiguës en tenant compte des étiquettes sur les noeuds et sur les arcs. Ainsi, le noyau sur les arcs revient à énumérer le nombre de chemins communs aux deux graphes.
De plus, l'équation 3.42 est réécrite plus élégamment en utilisant la propriété de la matrice d'adjacence. Pour cela, un nouveau graphe $G_\times : (V_\times, E_\times, \sigma_\times)$ est introduit en effectuant le produit direct des graphes $G : (V, E, \sigma)$ et $G' : (V', E', \sigma')$:

$$\begin{aligned} V_\times &= \{(v,v') \in V \times V' | \sigma_v = \sigma_{v'}\} \\ \sigma_{\times_{k=(v,v')}} &= \sigma_v \end{aligned}$$

Pour (i,j) correspondant à $((u,u'),(v,v')) \in V_\times^2$

$$\begin{aligned} [E_\times]_{i,j} &= \begin{cases} 1 \text{ si } [E]_{u,v} = [E']_{u,v'} = 1 \text{ et } \sigma_{(u,v)} = \sigma'_{(u',v')} \\ 0 \text{ sinon} \end{cases} \\ \sigma_{\times_{(i,j)}} &= \sigma_{(u,v)} \end{aligned}$$

L'équation 3.42 devient :

$$k(G, G') = \lim_{N \to \infty} \sum_{i=1}^{N} \sum_{u,v}^{|V_\times|} \lambda^i . [E_\times^i]_{u,v}$$

L'expression $E_\times^i$ peut être simplifiée si la matrice $E_\times$ est diagonalisable. Dans le cadre d'un graphe non-orienté, la matrice $E_\times$ étant une matrice réelle et symétrique, elle peut être diagonalisée.
Ainsi, si $E_\times$ peut être exprimé sous la forme $T^{-1}.D.T$ alors $E\times^i = T^{-1}.D^i.T$. Nous pouvons alors réécrire le noyau sous la forme :

$$k(G, G') = \sum_{u,v}^{|V_\times|} (T^{-1}.(\lim_{N \to \infty} \sum_{i=1}^{N} \lambda^i . D^i).T)_{u,v}$$

Pour calculer la limite, [GDR03] propose d'utiliser une décomposition en série exponentielle ou en série géométrique.
La décomposition en série exponentielle se base sur le principe que :

$$e^{\beta . E} = \lim_{N \to \infty} \sum_{i=0}^{N} \frac{(\beta E)^i}{i!}$$

Ainsi, en fixant $\lambda^i = \frac{\beta^i}{i!}$, nous obtenons :

$$k(G, G') = \sum_{u,v}^{|V_\times|} (T^{-1}.e^{\beta . D}.T)_{u,v}$$

De même, la décomposition en série géométrique se base sur, pour $\gamma < 1$:

$$\lim_{N \to \infty} \sum_{i=0}^{N} \gamma^i = \frac{1}{1 - \gamma}$$

En fixant $\gamma = \lambda.D$ et en veillant à ce que $\lambda.D < \mathbf{I}$, nous avons :

$$k(G, G') = \sum_{u,v}^{|V_\times|} (T^{-1}.(\mathbf{I} - \lambda.D)^{-1}.T)_{u,v}$$

Dans [KI02], Kashima et al. ont décomposé la similarité de deux graphes par une somme de similarité entre noeuds :

$$k(G, G') = \frac{1}{|V|.|V'|} \sum_{v_i \in V, v_j \in V'} k_n(v_i, v_j)$$

Avec V et V' l'ensemble des noeuds de G et respectivement de G'.
La similarité entre deux noeuds sera d'autant plus importante qu'il existera des chemins longs

communs aux deux graphes issus de ces noeuds. Pour assurer la terminaison du calcul, une probabilité $1-\lambda$ est fixée pour terminer le chemin et une probabilité λ pour continuer vers un successeur du noeud courant. Ainsi, plus le chemin sera long et plus la probabilité de terminer le chemin deviendra importante. Nous obtenons, ainsi, le noyau suivant :

$$k_n(u,u') = I(u,u').((1-\lambda) + \lambda. \sum_{(v,v') \in A_G(u) \times A_{G'}(u')} \frac{I_A((u,v),(u',v'))}{|A_G(u)|.|A_{G'}(u')|}.k_n(v,v'))$$

Avec $A_G(u) = \{v \in V | E_{uv} = 1\}$, $I(u,u') = 1$ si les étiquettes des noeuds u de G et u' de G' sont identiques ou 0 sinon et de même pour $I_A((u,v),(u',v'))$ qui retourne 1 si l'étiquette de l'arc reliant u et v de G est identique à l'étiquette de l'arc reliant u' et v' de G'.

Dans [KTI03], un noyau marginalisé sur tous les chemins possibles est proposé en ne considérant que des graphes orientés. Ainsi, le noyau est défini par :

$$k(G,G') = \lim_{L \to \infty} \sum_{l=1}^{L} \sum_{\mathbf{h}} \sum_{\mathbf{h}'} k_z(G,G',\mathbf{h},\mathbf{h}').P(\mathbf{h}|G).P(\mathbf{h}'|G')$$

La probabilité a posteriori d'avoir un chemin $\mathbf{h}$ de G de longueur l ($P(\mathbf{h}|G)$) est définie en fonction de la probabilité de débuter un chemin par un noeud $h_1 (P_s(h_1))$, la probabilité de terminer ce chemin par un noeud h_l ($Pq(h_l)$) et les probabilités d'effectuer une transition d'un noeud h_i vers un noeud h_{i+1} ($P_t(h_{i+1}|h_i)$). D'où :

$$P(\mathbf{h}|G) = P_s(h_1). \prod_{i=2}^{l=|\mathbf{h}|} P_t(h_i|h_{i-1}).P_q(h_l)$$

Le noyau k_z effectue la comparaison des deux chemins $\mathbf{h}$ et $\mathbf{h}'$ des graphes respectifs G et G'. Pour cela, le noyau calcule le produit des similarités entre les étiquettes des noeuds et des arcs du chemins :

$$k_z(G,G',\mathbf{h},\mathbf{h}') = \begin{cases} 0 \text{ si } |\mathbf{h}| \neq |\mathbf{h}'| \\ k_e(\sigma_{h_1},\sigma'_{h'_1}) \prod_{i=2}^{l} k_e(\sigma_{(h_{i-1},h_i)},\sigma'_{(h'_{i-1},h'_i)}).k_e(\sigma_{h_l},\sigma'_{h'_l}) \end{cases}$$

On rappelle que σ_{h_i} indique l'étiquette du noeud h_i de G et que $\sigma_{(h_i,h_{i+1})}$ indique l'étiquette de l'arc reliant le noeud h_i à h_{i+1}. Étant données deux étiquettes e et e', nous pouvons définir k_e comme étant un noyau retournant 1 si $e = e'$ ou 0 sinon. Toutefois, nous pouvons définir un noyau plus complexe si les étiquettes sont des réelles avec une certaine métrique. Nous pourrions alors définir un noyau gaussien qui tolérerait certaines différences entre les étiquettes.

Outre les graphes que nous venons de voir, Suzuki et al. ont introduit dans [SHSM03, SSM03] la notion de graphe acyclique orienté hiérarchique (*HDAG*). Les *HDAG* sont des graphes dont certains noeuds contiennent des graphes acycliques orientés. Cette structure a été proposée pour permettre la représentation de documents textuels ainsi que d'informations connexes. En effet,

un document textuel peut subir de multiple pré-traitement et des informations grammaticales et sémantiques peuvent lui être ajoutées. Ces informations combinées entre elles forment des structures hiérarchiques complexes.
En outre, un noyau a été proposé pour calculer la similarité entre les *HDAG*. Ce noyau a été évalué sur un problème de classification multi-classes avec l'algorithme SVM et la méthode "un contre tous" (un classifieur SVM par classe). Une base de données de 3000 questions divisées en 148 classes a été utilisée pour l'expérimentation. Les questions ont été pré-traitées à l'aide d'un *parser*. En outre, les entités nommées ont été étiquetées et les informations sémantiques ajoutées.
Les résultats ont montrés que le noyau *HDAG* était plus performant que le noyau *SubString Kernel* et le noyau "sac de mots".

Il est à noter que les noyaux pour graphes font l'objet d'une recherche très active [Bor07].

3.4 Conclusion

Dans la première partie de ce chapitre, nous avons très succinctement présenté un cadre permettant l'application de méthodes d'apprentissage numérique pour le traitement de données textuelles. Ainsi, nous avons énuméré les points clés dans la chaîne de pré-traitement qui sont principalement responsables des bonnes ou mauvaises performances d'un système. Parmi ces éléments clés, nous avons cité essentiellement la représentation des documents et la mesure de similarité s'appliquant à cette représentation. Dans le cadre de notre travail, nous nous focalisons sur ces deux points. Nous cherchons à définir des espaces sémantiques permettant une représentation fine du texte. De plus, nous présentons des mesures de similarité en définissant des noyaux appropriés.

En outre, les travaux, présentés dans [Joa02, Joa98], sur les SVM avec le noyau linéaire (sac-de-mots) pour la catégorisation de textes ont fourni des résultats prometteurs pour l'apprentissage numérique. Ces travaux ont donc ouvert la voie à une recherche riche et dynamique dans le domaine de la catégorisation de textes. Nos travaux s'inscrivent naturellement dans cette continuité. Par conséquent, nous utilisons les SVM comme algorithme de classement pour évaluer nos noyaux. Le noyau sac-de-mots nous sert de méthode de référence pour nous positionner.

Dans la seconde partie de ce chapitre, nous avons présenté une collection de méthodes assez variées, adaptées aux différents cas rencontrés lors du traitement des données structurées. La diversité de ces méthodes rend pour l'instant délicate toute évaluation comparative de leurs performances respectives. Le caractère très récent de ces travaux fait qu'il n'existe pour l'instant que très peu d'étude expérimentale comparative des comportements de ces différents algorithmes sur des données issues du monde réel et plus particulièrement du langage naturel. Insistons sur le fait que la souplesse des méthodes à noyaux facilite la construction de méthodes ad-hoc adaptées à la structure du problème à traiter.

CHAPITRE 4

Un nouveau noyau sémantique pour documents semi-structurés

4.1 Introduction

Dans ce chapitre, nous proposons un noyau utilisant une ontologie de concepts pour exploiter la sémantique des textes dans le traitement des documents [AVB07]. Ce noyau est défini pour être utilisé sur des documents semi-structurés en sections. Pour illustrer son application, nous nous focaliserons sur des documents biomédicaux semi-structurés composés de diverses observations médicales tels que les dossiers patients. L'ontologie biomédicale de l'UMLS (*Unified Medical Language System*) est utilisée, dans le cadre de ce travail, pour définir le noyau capable d'extraire les attributs sémantiques de tels documents. Nous montrons comment modéliser un document sous forme d'arbre sémantique en utilisant l'environnement UMLS. Ces arbres serviront, ensuite, de données d'entrée au noyau sémantique. Le noyau est défini en utilisant le cadre défini par le noyau de convolution [Hau99]. En outre, il incorpore une mesure de similarité sémantique basée sur l'UMLS.

4.2 L'environnement UMLS

L'*Unified Medical Language System* [1] est un environnement de connaissances linguistiques pour le domaine biomédical. Il contient aussi bien des bases de données linguistiques que des outils logiciels spécialement conçus pour faciliter son utilisation dans le cadre du traitement automatique des langues. L'UMLS a été spécialement développé pour servir de cadre général dans lequel viendraient s'imbriquer les différentes sources de connaissances linguistiques, telles que

[1] http ://www.nlm.nih.gov/research/umls

les ontologies, pour le domaine biomédical. Ainsi, l'UMLS peut être perçu comme un système d'unification des connaissances biomédicales. Il est essentiellement composé de trois grandes parties :

Le *Specialist Lexicon* est un lexique (*lexicon*) anglais incluant les termes biomédicaux. Il est fourni avec des outils de TAL et notamment des logiciels d'étiquetage morpho-syntaxique.

Le *Metathesaurus* est un thésaurus à savoir une base de vocabulaire contenant des informations sur les concepts biomédicaux telles que les noms des concepts et les relations entre ces derniers. Ce thésaurus peut incorporer des thésaurus provenant de diverses sources externes et de différentes langues. Chaque source définit une ontologie. Parmi ces sources, on peut citer MeSH et SNOMED-CT. Le *Metathesaurus* est fourni avec un programme d'installation appelé "MetamorphoSys" qui permet à l'utilisateur de sélectionner les sources de vocabulaire. Dans le cadre de notre travail, nous utilisons le *Metathesaurus* de manière typique. Ainsi, nous avons installé toutes les sources gratuites et SNOMED-CT. L'installation permet de créer la base de connaissance répartie en plusieurs fichiers textes. Parmi ces fichiers, nous citons les plus importants qui sont : *MRREL*, *MRSTY*, *MRXNS* et *MRXNW*. Ces fichiers définissent, respectivement, 1) les relations entre concepts, 2) la correspondance entre les concepts et les classes sémantiques, 3) la correspondance entre les groupes de mots normalisés et les concepts, et 4) la correspondance entre les mots normalisés et les concepts.

Le Réseau Sémantique définit 135 classes sémantiques avec 54 relations différentes les reliant. Les classes sémantiques sont utilisées pour regrouper les concepts du *Metathesaurus* en groupe de signification commune avec un niveau de granularité assez élevé.

4.3 Représentation Arborescente des Documents

Dans ce chapitre, nous nous intéressons aux documents médicaux semi-structurés. Nous définissons, ici, les données semi-structurées comme étant des données ne possédant pas un modèle relationnel strict comme les documents XML. En outre, nous restreignons notre travail aux documents ayant les propriétés suivantes :

1. un document est composé d'une ou plusieurs parties,
2. toutes les parties sont des données textuelles (les éléments non textuels seront simplement ignorés).

Par exemple, nous pouvons considérer les dossiers médicaux où chaque dossier patient est composé de plusieurs parties dont chacune est une observation médicale provenant d'un service spécifique comme, par exemple, une observation clinique ou une observation issue de la radiologie. Avec de tels documents, le calcul de la similarité entre deux documents, pour une tâche

de catégorisation ou de classification, nécessite de gérer de manière convenable les différentes parties en fonction de la source. En effet, il peut être inapproprié de comparer, par exemple, une observation radiologique avec une observation psychologique.

Pour chaque partie d'un document, nous effectuons les pré-traitements suivants :

1. Nettoyage du document : Tous les éléments qui peuvent introduire du bruit sont éliminés. Ainsi, tous les nombres, aussi bien sous un format numérique que sous un format littéral et les données non-textuelles sont retirés. Il est à noter que nous utilisons le terme "bruit" dans un sens général. En effet, les nombres et les symboles apportent une information sans quoi, il est peu probable qu'ils aient été introduits dans les documents. Cependant, le fait de ne pouvoir les traiter correctement peut introduire une information non pertinente voire même erronée. Cette information risquerait de noyer le sens général du document.

2. Segmentation du texte : La segmentation est utilisée pour découper le texte en unités de sens. La méthode la plus simple est de découper le texte en mots séparés par des ponctuations ou des espaces. Toutefois, avec une telle segmentation, nous pouvons détériorer ou même complètement perdre le sens correct d'un groupe de mots, sens induit par la présence des mots du groupe. Ceci est particulièrement vrai dans le domaine biomédical où les mots sont souvent combinés entre eux pour former des expressions avec des sens nouveaux. Pour gérer ce problème, une méthode consiste à segmenter le texte en unités lexicales (éléments lexicaux). Une unité lexicale peut être un unique mot ou un groupe de mots. Contrairement aux mots, les unités lexicales sont difficiles à identifier. En effet, des connaissances a priori sur le domaine sont nécessaires pour le découpage. Il s'agit pour cette tâche d'avoir un lexique (*lexicon*) propre au domaine. En utilisant le lexique, une façon naïve de segmenter le texte consiste à utiliser un algorithme simple de correspondance entre les mots du texte et le lexique, en favorisant en premier lieu les groupes contenant le plus de mot. L'algorithme 4.3.1 illustre le découpage naïf. Néanmoins, cette méthode ne tient pas compte de la catégorie morpho-syntaxique des mots. Ainsi, il peut regrouper des mots n'appartenant pas à la même catégorie grammaticale. Certaines unités lexicales peuvent ainsi être incohérentes et mener à une perte d'information. Il est donc préférable d'utiliser un étiqueteur morpho-syntaxique et d'utiliser un algorithme de correspondance prenant en compte la catégorie grammaticale des mots. Pour cela, nous avons utilisé le logiciel *dTagger* [DBL06] fourni avec le lexique UMLS (le *SPECIALIST lexicon*). Ce logiciel est capable d'étiqueter et découper le texte en unités lexicales.

3. Élimination des mots vides : A partir de l'ensemble des unités lexicales, nous éliminons les unités lexicales qui ne sont composées que de mots vides. Nous rappelons que par mots vides, nous désignons les mots qui n'apportent aucune information comme les

Algorithme 4.3.1 Algorithme de Segmentation Naïf

1: Pour un document d, découper d en mots séparés par des ponctuations ou des espaces
2: $G \leftarrow$ le premier mot de d
3: $i \leftarrow 1$
4: **tant que** tous les mots de d n'ont pas été traités **faire**
5: $\quad i \leftarrow i + 1$
6: $\quad w_i \leftarrow$ le $i^{\text{ème}}$ mot de d
7: $\quad$ Normaliser $G \cup w_i$ selon le lexique pour une recherche
8: $\quad$ **si** $G \cup w_i$ appartient au lexique **alors**
9: $\quad\quad$ ajouter w_i à la fin de G
10: $\quad$ **sinon**
11: $\quad\quad$ ajouter G à la liste des unités lexicales de d
12: $\quad\quad G \leftarrow w_i$
13: $\quad$ **fin si**
14: **fin tant que**
15: **si** G n'est pas vide **alors**
16: $\quad$ ajouter G à la liste des unités lexicales de d
17: **fin si**

déterminants et les verbes auxiliaires. Pour reconnaître, ces mots, nous utilisons une liste de mots vides disponibles sur internet [2].

4. Normalisation des mots : Les mots peuvent apparaître sous différentes formes fléchies fournissant, ainsi, des informations grammaticales. Ces informations induisent un espace à grande dimension dans lequel le processus d'apprentissage est rendu complexe. Pour réduire le nombre de dimension, chaque unité lexicale est normalisée, à savoir que les mots composant l'unité lexicale sont réduits à leur racine. Certains mots peuvent avoir différentes racines, par exemple la normalisation du mot "montre" donne les racines "montre" (en tant que nom) et "montrer". Par conséquent, chaque unité lexicale sera associée à un ensemble de formes normalisées. Dans l'UMLS, le *SPECIALIST lexicon* est fourni avec le programme java *Lexical Variant Generation (lvg)* qui peut être utilisé pour normaliser les groupes de mots. La normalisation est faite en 6 étapes illustrées par la figure 4.1.

5. L'annotation sémantique : L'information morphologique n'est pas suffisante pour calculer efficacement la similarité entre deux mots. En effet, il est généralement préférable de comparer la signification des mots plutôt que de comparer les mots proprement dits. Pour cela, nous devons déterminer le sens de chaque unité lexicale normalisée. Dans le *Metathesaurus* de l'UMLS, les mots sont définis avec leurs concepts en sachant qu'un

[2]Des liste de mots vides pour diverses langues sont disponible sur internet à l'adresse : http ://snowball.tartarus.org

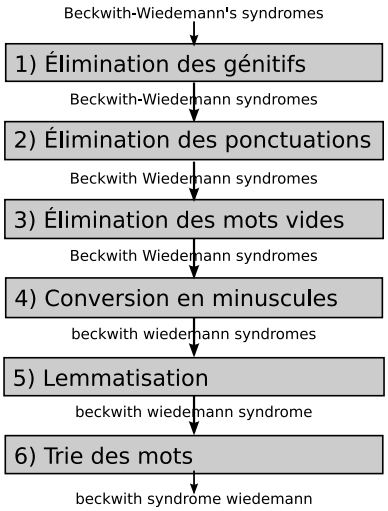


Fig. 4.1 – Le processus de normalisation d'une unité lexicale.

mot peut avoir un ou plusieurs concepts. Un concept peut être perçu comme une classe de mots regroupant les termes ayant un sens approximativement commun. De plus, les concepts sont reliés entre eux par diverses relations sémantiques. L'une des relations usuellement utilisée est la relation "est-un" qui décrit une relation "père-fils" d'abstraction et de spécificité entre deux concepts. Ainsi, les concepts et leurs relations peuvent être utilisés pour calculer une valeur de similarité sémantique. Dans ce but, un ensemble de concepts sera associé à chaque unité lexicale normalisée. L'algorithme d'annotation sémantique, que nous avons développé, est décrit par l'algorithme 4.3.2.

Finalement, après ces étapes de pré-traitement, chaque document peut être représenté sous une forme arborescente comme illustrée par la figure 4.2.

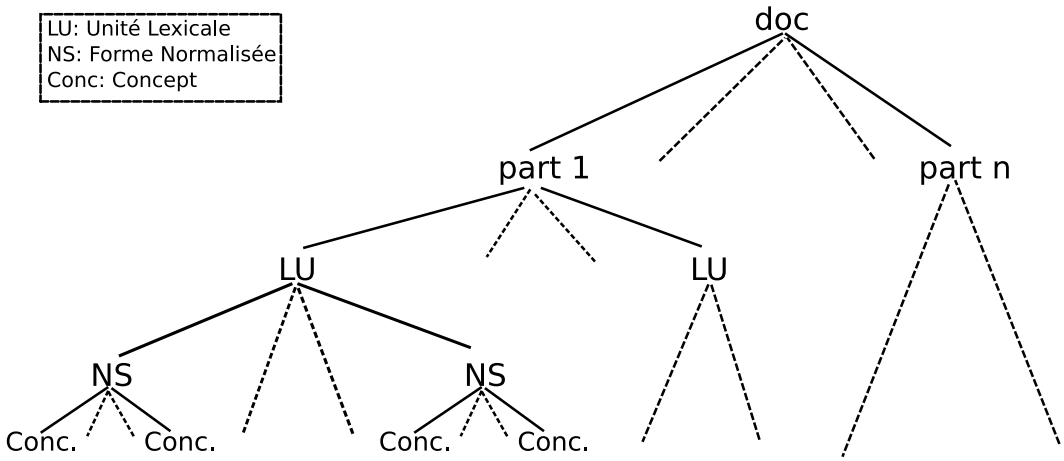


Fig. 4.2 – La forme arborescente d'un document après le pré-traitement.

Algorithme 4.3.2 Algorithme d'annotation sémantique

1: **pour tout** document d **et** partie $p \in d.parts$
   et unité lexicale $lu \in p.LU$ **faire**
2: **pour tout** groupe de mots normalisés$ns \in lu.NS$ **faire**
3: $ns.concepts \leftarrow$ trouver les concepts de ns à partir du fichier MRXNS de l'UMLS
4: **si** $ns.concepts$ est vide **alors**
5: $ns.concepts \leftarrow$ trouver les concepts de ns à partir du fichier MRXNW de l'UMLS
6: **fin si**
7: **fin pour**
8: **si** $\forall ns \in lu.NS$, $ns.concepts$ est vide
   et ns est groupe de mots constitué de plusieurs mots **alors**
9: retirer lu de $p.LU$
10: $p.LU \leftarrow p.LU \cup$ {découper lu en mots unitaires séparés par des caractères blancs}
11: **fin si**
12: $lu.NS = \emptyset \Rightarrow$ retirer lu de $p.LU$
13: $p.LU = \emptyset \Rightarrow$ retirer p de $d.parts$
14: $d.parts = \emptyset \Rightarrow$ retirer d
15: **fin pour**

4.4 Le Noyau Sémantique

4.4.1 Le cadre général

Nous avons vu dans la section 2.3.6 que les noyaux nous permettent de définir des produits scalaires dans un espace de Hilbert sans avoir à définir explicitement la fonction de transformation de l'espace d'entrée vers l'espace d'attributs. En outre, aucune contrainte n'est imposée sur le type des données d'entrée. On peut ainsi définir une fonction de similarité pour des documents semi-structurés. Il suffira de montrer que cette fonction définit un produit scalaire dans l'espace de Hilbert en montrant qu'elle est semi-définie positive. Pour cela, le théorème 2.3.14 se révèle être un outil puissant.

Lorsque les données d'entrée ne sont pas des vecteurs mais des données complexes telles que des documents semi-structurés, définir un noyau peut s'avérer être une tâche difficile. Pour faire face à ce problème, un noyau général a été proposé dans [Hau99]. Ce noyau appelé noyau de convolution a été étudié au chapitre précédent, à la section 3.3.1. L'idée est de décomposer la structure d'entrée en primitives et d'utiliser des noyaux différents sur ces primitives selon le type de donnée. Plus formellement, nous rappelons qu'étant donné $\mathcal{X}_1, \ldots, \mathcal{X}_n$, n espaces différents tels que $\forall x \in \mathcal{X}$ l'espace d'entrée, x peut être décomposé en $\mathbf{x} = x_1, \ldots, x_n$ avec $x_i \in \mathcal{X}_i$. Soit R une relation telle que $R(\mathbf{x}, x)$ est vrai si et seulement si $\mathbf{x}$ sont des composants de x, nous pouvons définir une relation inverse $R^{-1}(x) = \{\mathbf{x} : R(\mathbf{x}, x)\}$. Étant donné x et $y \in \mathcal{X}$, le noyau convolution est :

$$k(x,y) = \sum_{\mathbf{x}\in R^{-1}(x)} \sum_{\mathbf{y}\in R^{-1}(y)} k_{composite}(\mathbf{x},\mathbf{y}) \tag{4.1}$$

Bien que le noyau composite soit défini dans [Hau99] comme étant :

$$k_{composite}(\mathbf{x},\mathbf{y}) = \prod_{i=1}^{n} k_i(x_i,y_i) \tag{4.2}$$

avec k_i le noyau défini pour le type de donnée i sur $\mathcal{X}_i^2$, nous préférons donner une définition plus générale :

$$k_{composite}(\mathbf{x},\mathbf{y}) = F(K_1(x_1,y_1),\ldots,k_n(x_n,y_n)) \tag{4.3}$$

avec F une fonction de mélange.

Il est aisé de montrer que le noyau de convolution est un noyau valide si le noyau composite est un noyau valide. En effet, la somme de noyaux valides est un noyau valide [STC04].

4.4.2 Le noyau basé sur l'UMLS

Étant donné la structure arborescente présentée dans la figure 4.2, nous définissons $\mathcal{D}$ comme étant l'espace des arbres ayant le noeud *doc* comme racine et $\mathcal{P}_1,\ldots,\mathcal{P}_n$ comme étant les sous-espaces de $\mathcal{D}$ où $\mathcal{P}_i$ est l'espace des arbres ayant pour racine le noeud $part_i$. Le noyau basé sur l'UMLS est alors défini par la formule 4.1 où $\mathcal{X} = \mathcal{D}$ et $\mathcal{X}_i = \mathcal{P}_i$. L'expression du noyau composite (formule 4.3) reste valide. Toutefois, nous traitons les différentes parties d'un document d'une manière identique. Nous pouvons, ainsi, simplifier l'équation 4.3 en fixant $k_i(x_i,y_i) = k_p(x_i,y_i)$. En outre, nous définissons une fonction de mélange polynomial pondérée :

$$F(a_1,\ldots,a_n) = \left(\frac{\sum_{i=1}^{n} w_i a_i}{\sum_{i=1}^{n} w_i}\right)^l \tag{4.4}$$

avec $a_i, w_i \in \mathbb{R}^+$ et $l \in \mathbb{N}$. La fonction composite est alors :

$$k_{composite}(\mathbf{x},\mathbf{y}) = \left(\frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i.k_p(x_i,y_i)\right)^l \tag{4.5}$$

En fixant $l = 1$ et en ayant un poids uniforme, $k_{composite}$ devient l'espérance de la similarité entre les parties du même espace et pour $l > 1$, $k_{composite}$ utilise un espace d'attributs plus riche, c'est à dire un espace d'attributs de dimension supérieur. Nous pouvons montrer que si k_p est un noyau valide alors $k_{composite}$ est aussi un noyau valide pour $l \in \mathbb{N}$ et $w_i \in \mathbb{R}^+$ [STC04].

En utilisant le cadre général du noyau de convolution, nous définissons les noyaux suivants :

$$
\begin{aligned}
k_p(x_i, y_i) &= \sum_{u_1 \in \Gamma(x_i)} \sum_{u_2 \in \Gamma(y_i)} \sigma(u_1)\sigma(u_2) k_l(u_1, u_2) \\
k_l(u_1, u_2) &= \sum_{n_1 \in \Gamma(u_1)} \sum_{n_2 \in \Gamma(u_2)} k_n(n_1, n_2) \\
k_n(n_1, n_2) &= \begin{cases} \delta_{n_1,n_2} \\ \quad \text{si } \Gamma(n_i) = \emptyset \text{ ou } |\Gamma(n_i)| > \tau,\ i = 1, 2 \\ \sum_{c_1 \in \Gamma(n_1)} \sum_{c_2 \in \Gamma(n_2)} k_c(c_1, c_2) \\ \quad \text{sinon} \end{cases}
\end{aligned} \tag{4.6}
$$

où $\sigma(u)$ est la fréquence de l'unité lexicale u dans le document, δ_{n_1,n_2} est la fonction delta de Kronecker, $\Gamma(t)$ est l'ensemble des enfants du noeud t, τ est un entier naturel définissant un seuil et k_c est un noyau, défini à la section 4.4.3, calculant la similarité entre deux concepts. k_p calcule la similarité entre deux parties de documents en comparant chaque paire d'unités lexicales. De la même manière, k_l compare chaque paire de groupes de mots normalisés et k_n compare chaque paire de concepts. Cependant, pour k_n, si un groupe de mots normalisés d'une unité lexicale contient beaucoup de concepts (au delà d'un seuil τ fixé) alors le groupe de mots est considéré trop ambigu et utiliser la similarité entre concepts ne fera qu'ajouter du bruit. En outre, la similarité entre concepts ne peut être calculée si un groupe de mots ne possède pas de concepts. Dans ces deux cas, k_n utilisera une fonction d'identité pour comparer deux groupes de mots.

Il est aisé de montrer que si k_n est un noyau valide alors k_l et k_p sont des noyaux valides.

Proposition 4.4.1. *Soit k_c un noyau valide, k_n est, alors, un noyau valide.*

Démonstration. k_n peut être réécrit sous la forme suivante :

$$
\begin{aligned}
k_n(n_1, n_2) &= k_{nc}(n_1, n_2) + k_{nid}(n_1, n_2) \\
k_{nc}(n_1, n_2) &= \sum_{c_1 \in \Gamma(n_1)} \sum_{c_2 \in \Gamma(n_2)} k_c(c_1, c_2) \\
k_{nid}(n_1, n_2) &= (1 - g(n_1)g(n_2)) k_{id}(n_1, n_2) \\
k_{id}(n_1, n_2) &= 1 \text{ si } n_1 = n_2 \text{ sinon } 0 \\
g(n) &= 1 \text{ si } 0 < |\Gamma(n)| \leq \tau \text{ sinon } 0
\end{aligned}
$$

k_{nc} est un noyau de convolution, par conséquent, si k_c est un noyau valide, alors k_{nc} est aussi un noyau valide. Pour k_{nid}, la matrice de Gram associé, $\bar{K}_{nid}$, est, pour tout sous-ensemble fini, une matrice diagonale où chaque cellule de la diagonale est égal à 0 ou 1. Les valeurs propres de $\bar{K}_{nid}$ étant positives, la matrice est semi-définie positive. Et, selon le théorème 2.3.14, k_{nid} est un noyau valide. □

De plus, tous les noyaux k_p, k_l et k_n sont normalisés pour accorder le même poids à chaque noeud. Ainsi, les noeuds de tailles différentes (en terme de nombre d'enfants) peuvent être

comparés sur une même échelle. La normalisation est effectuée selon la formule suivante :

$$\hat{k}(x,y) = \frac{k(x,y)}{\sqrt{k(x,x)}\sqrt{k(y,y)}} \tag{4.7}$$

4.4.3 Le noyau de concepts

Le noyau de concepts, k_c est utilisé pour calculer la similarité entre deux concepts. Il utilise, pour cela, une taxonomie de relations "est-un" (relation père -fils) entre les concepts. Une taxonomie peut être représentée par un graphe orienté acyclique dans lequel chaque fils peut avoir un ou plusieurs parents. A partir du *Metathesaurus* de l'UMLS, nous pouvons construire une telle taxonomie :

1. en fusionnant toutes les ontologies installées sans tenir compte du champs "source" de ces dernières,
2. en éliminant de l'ontologie "unifiée" toutes les relations qui ne sont pas des relations "est-un",
3. en éliminant les relations résiduelles qui introduisent des cycles. Il est à noter que ces relations proviennent essentiellement de la fusion naïve des ontologies.

Étant donné une taxonomie de concepts, le noyau k_c se repose sur deux mesures de similarité sémantique entre les concepts. Ces mesures de similarité sont :

– **La similarité conceptuelle :** Cette similarité est basée sur la distance conceptuelle. L'idée est que plus deux concepts sont similaires et plus ils seront proches dans la taxonomie. La distance métrique entre deux concepts c_1 et c_2 peut être exprimée comme étant le nombre minimum de concepts (p) séparant c_1 de c_2. Autrement dit, cette distance est donnée par le nombre de concepts contenus dans le plus court chemin séparant c_1 de c_2 dans la taxonomie. La similarité est, alors, donnée par l'inverse de la distance. Leacock et Chodorow [LC98, PPPC06] ont proposé de normaliser cette similarité par la longueur maximum d'un chemin dans la taxonomie. Cette longueur est égale à deux fois la profondeur de la taxonomie. Un logarithme est ensuite utilisé sur le résultat. Nous avons utilisé la mesure de Leacock et Chodorow, après normalisation, comme une distance conceptuelle :

$$sim_{lch}(c_1,c_2) = -\frac{\log\left(\frac{p}{2.depth}\right)}{\log\left(2.depth\right)} \tag{4.8}$$

Cette mesure de similarité est insuffisante pour exprimer correctement le lien entre deux concepts. Premièrement, elle donne un poids égal à chaque concept de la taxonomie alors que les concepts spécifiques fournissent bien plus d'information que les concepts généraux. Deuxièmement, elle ne prend pas en compte le degré d'importance d'un lien entre un concept père et un concept fils. En effet, un concept père, qui est au passage un concept général, peut avoir plusieurs concepts fils spécialisés. Toutefois, le père peut être

plus associé à certains fils qu'à d'autres.

– **La similarité entropique :** Nous utilisons la mesure de Lin [Lin98], qui, étant donné deux concepts c_1 et c_2, donne une valeur de similarité plus élevée lorsque d'une part, l'ancêtre commun le plus proche (le moins général) de c_1 et c_2 ($\text{nca}(c_1, c_2)$) est le plus spécifique, à savoir situé le plus bas possible dans la taxonomie et d'autre part, c_1 et c_2 à proximité de l'ancêtre $\text{nca}(c_1, c_2)$. La mesure de Lin utilise la quantité d'information contenue (*Information Content*, IC) [Res95] comme une mesure de spécificité. En effet, les concepts spécifiques ont une quantité d'information (IC) plus importantes que les concepts généraux.

$$sim_{lin} = \frac{2.IC(nca(c_1, c_2))}{IC(c_1) + IC(c_2)} \tag{4.9}$$

$$IC(c) = -\log(\frac{freq(c)}{freq(root)}) \tag{4.10}$$

$$freq(c) = freq(c, \mathcal{C}) + \sum_{s \in G(c)} freq(s)$$

où $G(c)$ est l'ensemble des concepts qui ont c pour parent et $freq(c, \mathcal{C})$ est le nombre d'unités lexicales, dans un corpus de référence $\mathcal{C}$, pouvant être associées au concept c. Dans le cadre de nos travaux, nous utilisons le corpus d'apprentissage avec le corpus de test comme corpus de référence. Cette utilisation rend notre méthode d'apprentissage, semi-supervisée. En effet, dans le cas de l'apprentissage semi-supervisée, les documents non étiquetés, ici les documents de test, sont utilisés d'une façon ou d'une autre pour influencer l'apprentissage supervisé.

Étant donné les mesures de similarité définies ci-dessus, le noyau de concept peut être défini ainsi :

$$k_c(c_1, c_2) = sim_{lch}(c_1, c_2) \times sim_{lin}(c_1, c_2) \tag{4.11}$$

Theorème 4.4.2. *k_c est un noyau valide.*

Démonstration. Pour tout sous-ensemble fini de l'espace des concepts, la matrice de Gram $\bar{K}_c$ associée à k_c est une matrice symétrique avec des valeurs positives puisque les mesures de similarité sont des fonctions symétriques positives. $\bar{K}_c$ peut être diagonalisée et toutes ses valeurs propres sont positives. Par conséquent, $\bar{k}_c$ est semi-definie positive. Nous pouvons en conclure que d'après le théorème 2.3.14, k_c est un noyau valide. □

4.5 Évaluation expérimentale

Afin d'évaluer les performances du noyau sémantique basé sur l'UMLS, nous avons comparé ce noyau au noyau linéaire sac-de-mots [Joa98] et au classifieur multinomial naïf de Bayes [Joa97]. Ces méthodes ont été utilisées pour résoudre un problème de catégorisation de texte. Les noyaux ont été utilisés avec un classifieur SVM [Vap95].

4.5.1 Le corpus

Pour nos expériences, nous avons utilisé le corpus médical ICD-9-CM [PBM+07] mis à disposition par le centre médical informatique (CMC) de l'hôpital pour enfants de Cincinnati. Ce corpus a été fourni dans le cadre du challenge 2007 sur le TAL appliqué au domaine médical organisé par le CMC [3].

```
<doc id="97636670" type="RADIOLOGY_REPORT">
    <codes>
        <code origin="CMC_MAJORITY" type="ICD-9-CM">786.2</code>
        <code origin="COMPANY3" type="ICD-9-CM">786.2</code>
        <code origin="COMPANY1" type="ICD-9-CM">204.0</code>
        <code origin="COMPANY1" type="ICD-9-CM">786.2</code>
        <code origin="COMPANY1" type="ICD-9-CM">V42.81</code>
        <code origin="COMPANY2" type="ICD-9-CM">204.00</code>
        <code origin="COMPANY2" type="ICD-9-CM">786.2</code>
    </codes>
    <texts>
        <text origin="CCHMC_RADIOLOGY" type="CLINICAL_HISTORY">
            Eleven year old with ALL, bone marrow transplant on Jan. 2,
            now with three day history of cough.
        </text>
        <text origin="CCHMC_RADIOLOGY" type="IMPRESSION">
            1. No focal pneumonia. Likely chronic changes at the left
            lung base.  2. Mild anterior wedging of the thoracic
            vertebral bodies.
        </text>
    </texts>
</doc>
```

Fig. 4.3 – Un exemple de document issu du corpus ICD-9-CM.

Le corpus est composé d'une base d'apprentissage et d'une base de test préparées pour le problème de catégorisation de texte. La base d'apprentissage et la base de test contiennent chacune 1000 documents semi-structurés. Chaque document est structuré en deux sections : l'une étant une observation clinique et l'autre une observation radiologique d'un patient. La figure 4.3 donne un exemple de document issu de ce corpus. Dans la base d'apprentissage, chaque document a été étiqueté, par trois différents organismes, avec un ou plusieurs codes ICD-9-CM. Le ou les codes pertinents sont obtenus pour un document par un vote majoritaire entre les trois organismes. Ainsi, la base d'apprentissage est étiquetée avec 45 différents codes ICD-9-CM. Il existe 94 combinaisons de code distinctes utilisées pour le multi-étiquetage de ces documents.

Le code ICD-9-CM est un code hiérarchique catégorisant les différents types de maladie. Un exemple de code est donnée dans la figure 4.4. Ces codes sont utilisés pour justifier les

[3] www.computationalmedicine.org/challenge

procédures médicales pour un patient. En outre, ils sont utilisés par les compagnies d'assurance pour le remboursement des frais.

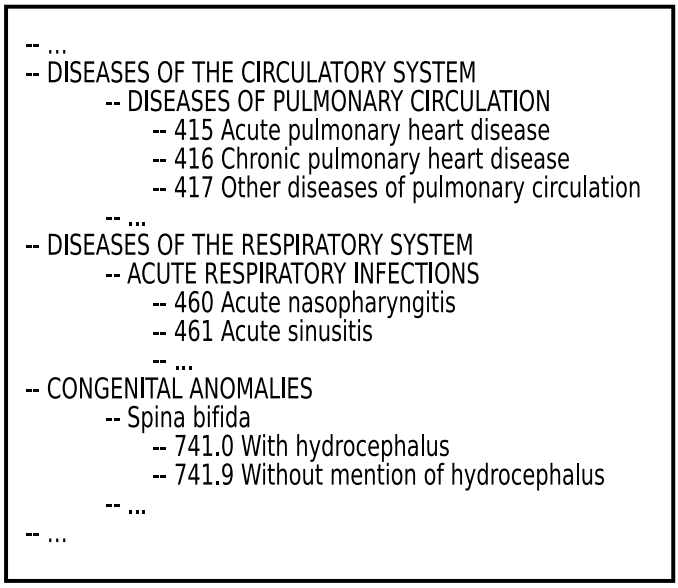


Fig. 4.4 – Un extrait de codes ICD-9-CM.

L'objectif de la tâche de catégorisation est d'attribuer à chaque document un ou plusieurs codes ICD-9-CM indiquant le type de maladie. Pour cela, la relation entre les observations et les codes peut être "apprise" à partir du corpus d'apprentissage.

4.5.2 La préparation des expériences

L'ensemble des données non-étiquetées, à savoir les données de test, n'a été utilisé, dans le noyau sémantique basé sur l'UMLS, que pour la mesure de similarité de Lin.

Toutes les expériences ont été effectuées sur l'ensemble d'apprentissage avec une validation croisée à 10 blocs (*10-folds cross-validation*) répétée 10 fois. Les résultats sont exprimés sous la forme d'une moyenne avec l'écart type sur 100 essais.

Pour le noyau sémantique, les poids w_i ont été fixés à un dans l'équation 4.5. En effet, nous souhaitions donner un poids égal à toutes les parties d'un document faute d'autre connaissance a priori.

Pour le noyau linéaire et le classifieur naïf de Bayes, les parties d'un document ont été fusionnées. La représentation sac-de-mots a été utilisée. Les mots vides ont été éliminés et la dimension de l'espace a été réduite en réduisant les mots fléchis à une forme commune avec l'algorithme de *stemming* de Porter[4]. De plus, pour le noyau linéaire, chaque mot a été pondéré en utilisant la pondération TF-IDF. Les vecteurs, représentant les documents, ont été normalisés pour donner un poids égal à chaque document.

[4] www.snowball.tartarus.org

Ce problème de classification étant un problème de multi-catégories, nous avons utilisé, pour les classifieurs SVM avec les noyaux sémantiques et linéaires, une stratégie un-contre-tous. Pour chaque classe, un classifieur a été entraîné avec les documents de cette classe comme étant des instances positives et le reste comme étant des instances négatives. Ainsi, 45 classifieurs par essai ont été entraînés pour la catégorisation de texte sur ce corpus.

Pour affecter un ou plusieurs codes à un document, nous avons utilisé la stratégie *SCut* [Yan01] définie dans la section 3.2.3.3. L'ensemble d'apprentissage a été découpé en un ensemble d'apprentissage proprement dit et un ensemble de validation. Le nouvel ensemble d'apprentissage a alors été utilisé pour entraîner le classifieur. Le classifieur a ensuite été utilisé pour attribuer un score à chaque document de l'ensemble de validation. 45 scores par document ont été attribués à l'ensemble de validation. Pour chaque classe c, un seuil t_c est défini et un document d est assigné à une catégorie c si et seulement si le score de d pour la classe c est supérieur ou égal au seuil t_c. Chaque seuil est fixé à partir de l'ensemble de validation en optimisant le score micro-moyenné F1 défini dans la section 3.2.4. Pour nos expériences, nous avons utilisé 80% des données d'apprentissage pour l'apprentissage du modèle et les 20% restants ont été utilisés pour constituer l'ensemble de validation.

4.5.3 Les résultats expérimentaux

La figure 4.5 montre la variation de la performance F1 du noyau sémantique en fonction du degré du polynôme l de la fonction de mélange avec $\tau = 10$. Le meilleur score, micro-F1=84%, est obtenu pour $l = 2$ puis la performance décroît lorsque l croît. Néanmoins, le score F1 se stabilise pour $l > 30$. Ces résultats montrent que, pour ce corpus, le fait de mélanger la similarité entre des parties différentes ne fournit aucune information. En effet, une valeur de similarité élevée pour un type de partie n'induit pas forcément une valeur de similarité élevée pour l'autre partie. Ainsi, les différents types de parties ne devraient pas être directement reliés pour le calcul de la similarité entre documents. Une simple moyenne pondérée permet d'obtenir d'excellents résultats pour ces données. Il s'avère qu'un polynôme de degré 2 est un bon compromis.

La figure 4.6 montre que la variation du seuil de concepts τ n'influence pas vraiment les performances du noyau sémantique. Notre hypothèse principale concernant les concepts étaient qu'un groupe de mots normalisés associé à beaucoup de concepts était probablement trop ambigu pour apporter une information claire en accord avec le contexte. Bien que cette hypothèse semble pertinente, nous avons utilisé une taxonomie spécialisée qui correspond bien au contexte du corpus. Par conséquent :

1) un groupe de mots normalisés est, généralement, associé à moins d'une quinzaine de concepts,

2) tous les concepts, pour un même groupe, ont approximativement la même signification.

Toutefois, une valeur de 10 pour τ améliore légèrement les résultats.

Les performances du noyau sémantique (avec $\tau = 10$ et $l = 2$), du noyau linéaire et du classifieur naïf de Bayes sont indiquées dans les tableaux 4.1 et 4.2. Comme nous pouvons le

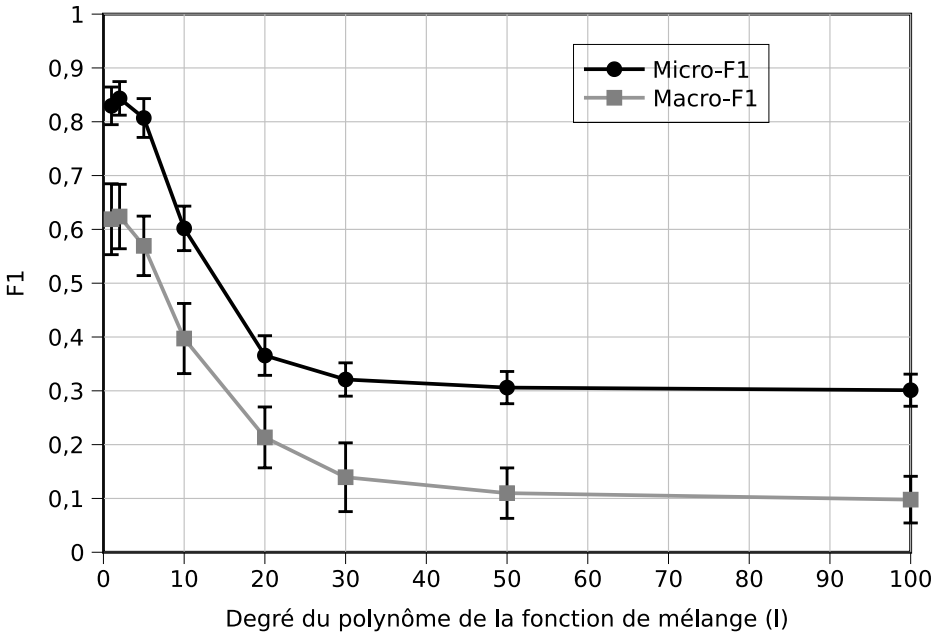


Fig. 4.5 – Variation du score F1 en fonction du degré l du polynôme de la fonction de mélange (eq. 4.4) sur le corpus ICD-9-CM (avec $\tau = 10$).

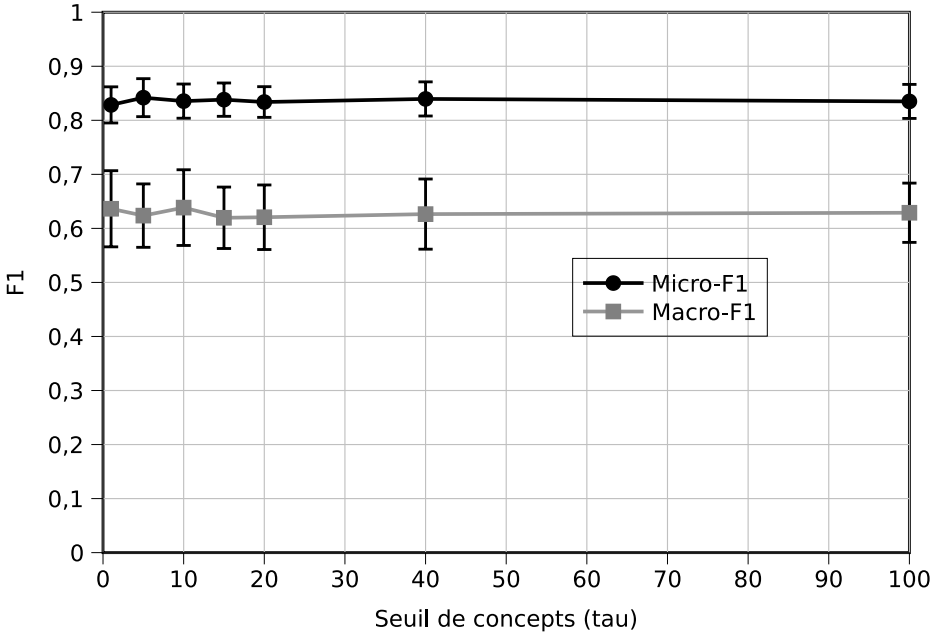


Fig. 4.6 – Variation du score F1 en fonction du seuil de concepts τ (eq. 4.6) sur le corpus ICD-9-CM (avec $l = 2$).

constater, le noyau sémantique est clairement plus performant de 7% que le noyau linéaire et de 26% que le classifieur naïf de Bayes pour le score F1 micro-moyenné. Les bonnes performances du noyau linéaire montrent que les mots utilisés dans les documents d'une même classe ont une même forme morphologique. Ainsi, après le *stemming*, les documents d'une même classe partagent, généralement, les mêmes termes.

Classifieur	**F1**	**Précision**	**Rappel**
Noyau UML	$83\% \pm 3\%$	$82\% \pm 3\%$	$84\% \pm 3\%$
Noyau Linéaire	$76\% \pm 3\%$	$77\% \pm 3\%$	$75\% \pm 4\%$
Bayes Naïf	$57\% \pm 4\%$	$59\% \pm 5\%$	$55\% \pm 4\%$

Tab. 4.1 – Scores micro-moyennés pour le corpus ICD-9-CM.

Néanmoins, d'après le tableau 4.2, bien que la macro-moyenne du score F1 du noyau sémantique soit supérieur de 2% à celle du noyau linéaire, ils ont approximativement les mêmes performances. Pour la macro-moyenne, les scores sont calculés pour chaque classe puis moyennés. Un même poids est attribué à chaque classe. Ainsi, les petites classes, à savoir les classes avec très peu de documents, vont influencer le score global de la même manière que les grandes classes. Le système de classification ICD-9-CM étant un système de code hiérarchique, certaines classes appartiennent à la même branche hiérarchique. Par conséquent, ces classes sont sémantiquement similaires. Le noyau sémantique va avoir tendance à affecter aux documents d'une petite classe, une classe plus large qui sera sémantiquement proche. Nous rappelons que nous avons utilisé la stratégie un-contre-tous. Le macro-rappel pour une petite classe sera ainsi faible et la macro-précision pour une grande classe sera, aussi, faible. Ceci aura pour effet d'affaiblir le score F1. Le noyau linéaire réagit différemment. En effet, il est influencé par les mots, après *stemming*, communs aux documents. Ainsi, si une petite classe possède des mots spécifiques qui n'apparaissent pas dans la grande classe, la performance du noyau linéaire ne sera pas affectée par ce problème.

Classifieur	**F1**	**Précision**	**Rappel**
Noyau UML	$62\% \pm 4\%$	$63\% \pm 5\%$	$63 \pm 4\%$
Noyau Linéaire	$60\% \pm 6\%$	$62\% \pm 6\%$	$61\% \pm 6\%$
Bayes Naïf	$36\% \pm 5\%$	$40\% \pm 7\%$	$38\% \pm 6\%$

Tab. 4.2 – Scores macro-moyennés pour le corpus ICD-9-CM.

L'impact du nombre de documents d'apprentissage utilisés sur les performances du classifieur est indiqué dans les figures 4.7 et 4.8. L'expérience montre que la performance, en terme de micro et de macro F1, du classifieur naïf de Bayes s'améliore linéairement en fonction du nombre de données d'apprentissage. Ce résultat était quelque peu prévisible étant donné que ce classifieur est une méthode probabiliste basée sur la fréquence d'occurrences des termes. Ainsi, il requiert une quantité importante de données d'apprentissage pour estimer la distribution des mots dans le corpus. Le noyau sémantique n'a besoin que de 40% de données d'apprentissage pour atteindre sa valeur optimale pour le micro-F1 et 70% pour atteindre sa valeur macro-F1 optimale. La différence entre les deux pourcentages peut être expliquée par le fait que le corpus contient des catégories avec très peu de documents. Une fraction importante de documents d'apprentissage est alors nécessaire pour capturer l'information discriminante. La micro-F1 donne un poids égal à tous les documents. Par conséquent, cette valeur ne sera pas affectée par les petites catégories. Cependant, la macro-F1 étant une moyenne sur toutes les catégories, les petites classes vont avoir une influence importante sur cette valeur. La variation du score F1 du noyau linéaire est proche de celle du noyau sémantique mais avec un aspect un peu plus linéaire. La micro-F1 optimal pour le noyau linéaire est atteint par le noyau sémantique avec uniquement 20% des données d'apprentissage. Ce résultat montre que l'utilisation du noyau sémantique améliore de manière significative les performances de catégorisation.

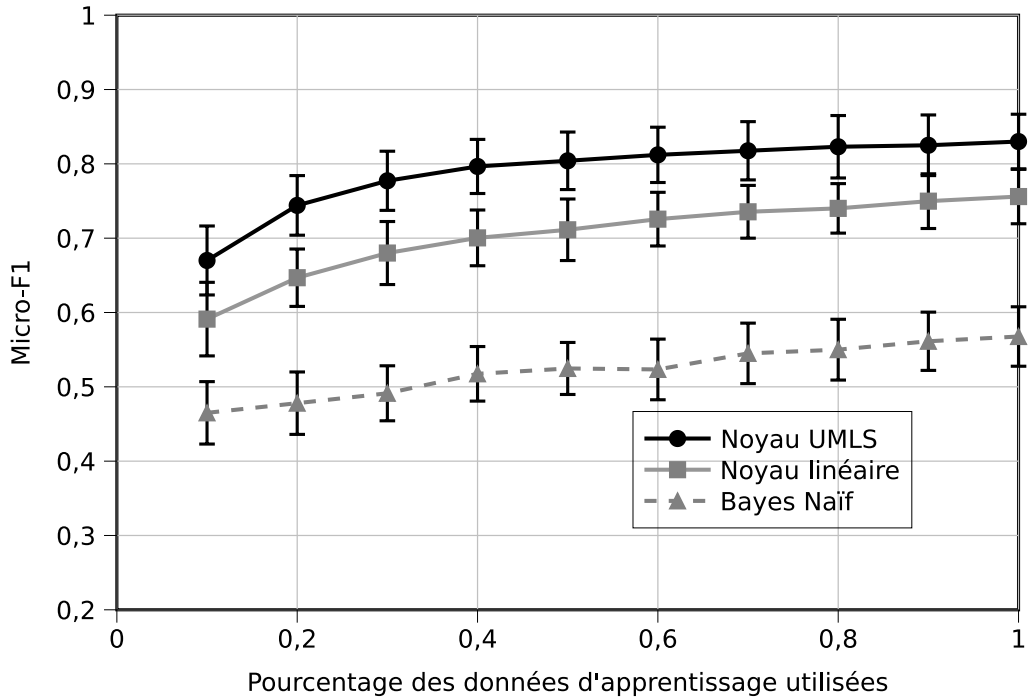


Fig. 4.7 – Variation de la micro-F1 en fonction du pourcentage de données d'apprentissage utilisées.

Un autre point intéressant est montré dans la figure 4.8. Pour une faible quantité de données

d'apprentissage, la macro-F1 du noyau linéaire est plus élevée que celle du noyau sémantique. Comme nous l'avons souligné plus haut, ceci est principalement dû aux petites catégories. Les sens sémantiques de ces classes, extraites par le noyau sémantique, sont trop ambigus et corrélés avec les sens des catégories plus grandes. Le noyau linéaire est basé sur la fréquence des termes. En trouvant les mots discriminants pour les petites classes, la macro-F1 du noyau linéaire est moins affectée par ce problème.

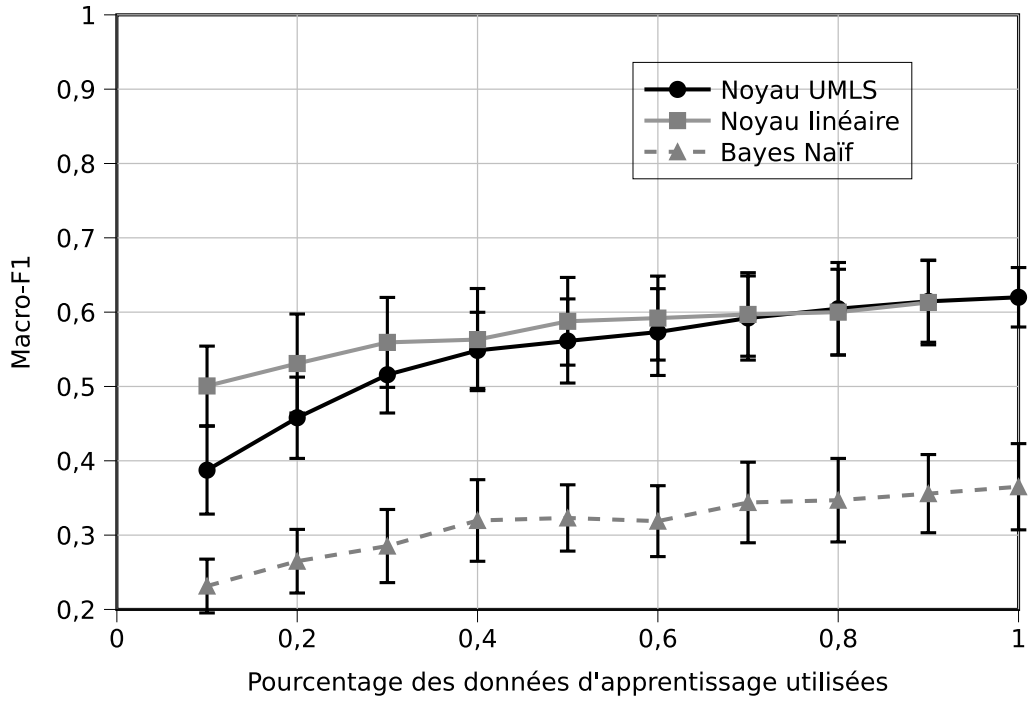


Fig. 4.8 – Variation de la macro-F1 en fonction du pourcentage de données d'apprentissage utilisées.

Comme dernière évaluation, nous avons entraînés un classifieur SVM avec un noyau sémantique sur la totalité de l'ensemble d'apprentissage. Nous avons toutefois réservé 20% de ces données pour l'ensemble de validation. Le classifieur a, alors, été utilisé pour étiqueter les documents non-étiquetés de l'ensemble de test. Les documents de test nouvellement étiquetés ont été ensuite envoyés au CMC (*Computational Medicine Center*) de l'hôpital pédiatrique de Cincinnati pour le challenge international de catégorisation de documents médicaux 2007. Notre algorithme a obtenu une valeur de micro-F1 de 85%. Le noyau sémantique a ainsi été classé dans les 10 meilleurs algorithmes sur les 44 algorithmes utilisés pour le challenge. Le tableau 4.3 donne le classement des participants [5] en fonction du score micro-F1. Ce tableau est extrait de la page web http ://www.computationalmedicine.org/challenge/res.php.

[5] À l'heure actuelle les méthodes utilisées par les différents participants n'ont pas été rendues publiques.

Position	Participant	Micro-F1	Macro-F1
1	Szeged	0.8908	0.7691
2	University at Albany	0.8855	0.7291
3	University of Turku	0.8769	0.7034
4	PENN	0.8760	0.7210
5	LMCO-IS & S	0.8719	0.7760
6	GMJ_JL	0.8711	0.7334
7	SULTRG	0.8676	0.7322
8	MANCS	0.8594	0.6676
9	otters	0.8509	0.6816
10	**Aseervatham**	**0.8498**	**0.6756**
11	LHC/NLM	0.8469	0.6916
12	ohsu_dmice	0.8457	0.6542
13	Stockholm	0.8392	0.6684
14	Fabrizio Sebastiani	0.8375	0.6870
15	UOE	0.8360	0.6802
16	Davide	0.8284	0.6784
17	SIM	0.8277	0.6959
18	Yasui Biostatistics Team	0.8274	0.6825
19	SUNY-Buffalo	0.8258	0.6276
20	I2R	0.8180	0.6763
21	LLX	0.8147	0.7343
22	BME-TMIT	0.7992	0.6406
23	Watson	0.7977	0.5488
24	ErasmusMC	0.7969	0.6369
25	NJU-NLP Group	0.7950	0.5846
26	bozyurt	0.7916	0.5643
27	Dharmendra	0.7900	0.5758
28	strappa	0.7822	0.5662
29	UMN	0.7741	0.5540
30	cl.naist.jp	0.7657	0.5571
31	MITRE	0.7455	0.5097
32	BAL	0.7341	0.5936
33	Delbecque	0.7246	0.5682
34	ravim	0.7145	0.5522
35	MIRACLE	0.7067	0.5109
36	Magne Rekdal	0.6869	0.5238
37	cu_nlp	0.6865	0.5052
38	CNRC	0.6823	0.5089
39	ARAMAKI	0.6786	0.4459
40	SINAI	0.6719	0.5248
41	Hirukote	0.6556	0.4770
42	MERLIN	0.5768	0.3345
43	ILMA	0.3905	0.3351
44	CLaC	0.1541	0.1918

Tab. 4.3 – Résultats et classement des participants au challenge international 2007 de catégorisation de documents médicaux organisé par le CMC de l'hôpital pédiatrique de Cincinnati.

4.6 Conclusion

Dans ce chapitre, nous avons proposé un noyau sémantique qui utilise l'environnement de l'UMLS pour extraire des sens sémantiques à partir de documents textes. Le noyau a été conçu pour traiter des documents médicaux semi-structurés. Nous avons décrit une méthode de prétraitement pour représenter ces documents par des arbres de concepts. Le noyau sémantique utilise une taxonomie de concepts, construite à partir du *Metathesaurus* de l'UMLS, pour calculer la similarité entre deux arbres de concepts.

Les performances du noyau sémantique ont été évaluées expérimentalement sur une tâche de catégorisation de documents. Le corpus médical ICD-9-CM de l'hôpital pédiatrique de Cincinnati a été utilisé pour l'évaluation. Les résultats ont montré que le noyau sémantique est significativement plus performant que le noyau linéaire et le classifieur multinomial naïf de Bayes. De plus, ce noyau a été évalué lors du challenge international 2007 de catégorisation de documents médicaux (*Classifying Clinical Free Text Using Natural Language Processing*) organisé par le CMC. À cette occasion, notre noyau sémantique a été classé dans les 10 meilleurs algorithmes parmi 44 méthodes de catégorisation.

Bien que le noyau sémantique ait été principalement conçu pour le domaine biomédical, il peut facilement être adapté pour d'autres domaines. Il suffit, pour cela, d'utiliser une taxonomie de concepts adaptée au domaine. Dans un futur proche, nous souhaitons utiliser ce noyau sur un corpus plus général tel que les corpus Reuters en utilisant la taxonomie WordNet. En outre, il existe deux points d'amélioration possibles. Premièrement, un processus de désambiguïsation devrait être introduit lors de la normalisation des unités lexicales et de l'annotation sémantique de ces unités lexicales. Deuxièmement, un ordre ou des liens devraient être établis entre les unités lexicales d'un document pour mieux modéliser le document.

CHAPITRE 5

Des nouveaux noyaux basés sur l'information sémantique latente

5.1 Introduction

Dans ce chapitre, nous nous intéressons aux concepts latents extraits automatiquement à partir des documents textuels. L'objectif est essentiellement d'envisager les concepts latents comme alternatives aux concepts linguistiques. Ainsi, il serait possible de mettre en place des systèmes de traitement de documents textuels entièrement autonomes. Ces systèmes pourraient, ainsi, apprendre, s'adapter et, voire même, évoluer selon la nature des textes.
Dans le cadre de cette thèse, nous nous limitons à l'Analyse Sémantique Latente (LSA) comme méthode statistique d'extraction de concepts latents.

Ce chapitre est organisé en trois parties. La première partie présente l'information latente et expose quelques méthodes statistiques d'extraction de cette information. Dans la seconde partie, nous proposons un modèle d'espace vectoriel construit à partir de concepts linguistiques. Nous montrons, dans cette partie, que l'information latente extraite à partir de cet espace permet d'améliorer les performances de catégorisation. Dans la dernière partie, nous proposons l'utilisation d'espaces sémantiques locaux orthogonaux entre eux. Chaque espace sémantique local est obtenu en appliquant une LSA locale à une catégorie de documents. Nous montrons, expérimentalement, que cette méthode est performante lorsque les dimensions de chaque espace local sont faibles.

5.2 Information Latente

Bien que le modèle de l'espace vectoriel, VSM (section 3.2.2.1), serve de référence pour la représentation de textes en langage naturel, il souffre de deux lacunes : il ne tient pas compte

de l'ordre d'apparition des mots dans un texte et de la signification sémantique des mots. L'ordre d'apparition des mots ainsi que la structure peuvent être prises en compte en définissant une structure adaptée pouvant intégrer ces informations. Il suffit ensuite de définir un noyau pour données structurées pour obtenir un espace de description dans lequel la structure sera représentée sous forme vectorielle. Typiquement, pour tenir compte de l'ordre d'apparition des mots, il suffira d'utiliser un noyau pour séquences (section 3.3.2) en remplaçant, au besoin, la notion de séquence de caractères par la notion de séquence de mots.

La seconde lacune du VSM est la non-prise en compte de la notion de sémantique d'un terme, ou plus généralement, d'un document. L'exploitation de la sémantique permet de tenir compte des relations entre termes qui échappent à la VSM. Il est, ainsi, possible d'améliorer les performances d'un système en utilisant un espace sémantique plutôt qu'un espace induit par la morphologie des termes tel que le VSM.

Dans cette partie, nous nous intéressons aux espaces sémantiques et plus précisément aux espaces obtenus par extraction de concepts latents. Les concepts latents sont extraits par des méthodes statistiques telles que l'Analyse Sémantique Latente (LSA) [DDL+90]. Nous décrivons, dans cette section, principalement la LSA et les méthodes dérivées.

5.2.1 De l'espace des termes à un espace sémantique

Nous avons vu que l'un des problèmes majeurs du VSM est qu'il se limite à l'information morphologique des mots et à la fréquence d'occurrence de ces mots, il semble évident que les performances des systèmes se basant sur ce modèle seront fortement liées à la présence de termes, ou de combinaison de termes, spécifiques. Cependant, certains mots peuvent être sémantiquement proches (synonymes) et d'autres, par contre, peuvent avoir plusieurs sens selon le contexte (polysémie). La présence de ces termes aura une influence non négligeable sur les performances. Ainsi, il est intéressant de se baser, non plus sur la forme du mot, mais sur son sens, à savoir la sémantique du mot. Pour cela, nous utiliserons la notion de concept que nous définirons comme étant une classe de mots partageant le même sens sémantique. En définissant un espace de concepts, donc un espace sémantique, les documents ne seront non plus représentés par une information morphologique mais par un sens défini par un vecteur de concepts. Ainsi, les documents partageant des concepts proches seront projetés dans la même région de l'espace. Il sera, alors, possible de regrouper les documents en classe de concepts. Ceci permettra de gérer la synonymie et la polysémie. Nous souhaitons, toutefois, signaler que la polysémie sera gérée de manière indirecte par la présence des termes du contexte. Ces termes auront pour effet d'orienter le document vers des régions spécifiques de l'espace sémantique. Une manière plus intéressante de prendre en charge la polysémie consisterait à utiliser une étape de désambiguïsation qui permettrait de sélectionner le ou les concepts les plus significatifs d'un terme selon son contexte.

En outre, il est à noter que le VSM est un espace de termes. Ainsi, les vecteurs, représentant les documents, résident dans un espace propre à une langue. Il sera donc, impossible, de représenter un document émanant d'une autre langue si le dictionnaire qui a été utilisé pour

définir le VSM n'a pas été défini pour prendre en charge cette langue. L'espace des concepts, ou l'espace sémantique, est quant à lui indépendant de la langue. Les documents, quelqu'ils soient, peuvent être représentés dans cet espace à condition qu'il soit possible d'associer aux termes des documents des concepts de l'espace sémantique. Il s'agira, par exemple, d'avoir un thésaurus propre à une langue avec des concepts unifiés.

Le passage de l'espace des termes à l'espace sémantique est un point délicat. Il existe deux approches permettant de définir l'espace sémantique :

Approche agnostique : Dans cette approche, nous partons de la supposition qu'il est possible d'extraire automatiquement des connaissances sans l'utilisation de sources externes. Le système est ainsi totalement indépendant. L'espace sémantique est défini, dans cette approche, par des concepts extraits par des méthodes statistiques. Nous appellerons ces concepts des concepts statistiques. Parmi ces méthodes, nous pouvons citer deux grandes catégories : 1) les techniques de *clustering* et 2) les méthodes basées sur l'Analyse Sémantique Latente (LSA). Les méthodes de *clustering* consistent à regrouper les mots en classe selon un critère d'appartenance. Par exemple, dans [BM98a], la divergence de Kullback-Leibler est utilisée pour regrouper les mots et dans [BPd$^+$92], l'information mutuelle est utilisée. Les méthodes basées sur la LSA utilisent, quant à elles, l'information de co-occurrence des termes pour extraire des relations entre ces termes. Ces relations, généralement linéaires, sont appelées "concepts". Les relations sont extraites par une décomposition de la matrice de termes par document.

Approche a priori : Cette approche se base sur l'utilisation de connaissances a priori spécifiques à un domaine. Ces connaissances proviennent de sources externes, choisies en fonction du corpus, et sont intégrées directement au processus de traitement. Les sources externes sont usuellement constituées d'un thésaurus et d'une ontologie. Le thésaurus intègre les termes spécifiques au domaine du corpus et les regroupe en catégories hiérarchiques, appelées "concepts". Nous utiliserons l'expression "concept linguistique" pour désigner ces concept et faire la différence avec les concepts statistiques mentionnés plus haut. L'ontologie définit les différentes relations, non exhaustives, entre ces concepts linguistiques. Les concepts linguistiques avec leurs relations sont utilisés pour calculer des similarités entre les documents. Généralement, ils sont utilisés au sein d'une fonction de similarité [PPPC06, Lin98, JC97, Res95] ou dans des noyaux. Les noyaux ont été un sujet d'intérêt considérable, principalement, dû au fait qu'ils peuvent être définis dans des espaces vectoriels complexes implicites. Dans [SdB00], le thésaurus WordNet [Fel98] a été utilisé comme source d'information externe. Une valeur de similarité sémantique est calculée en utilisant l'inverse de la longueur du chemin dans la taxonomie puis lissée avec un noyau gaussien. Ce noyau a été amélioré dans [BCM06] en remplaçant la longueur du chemin par une similarité de densité conceptuelle et en éliminant le noyau gaussien.

Chacune de ces approches a, bien évidemment, des points positifs et des points négatifs. L'avantage de l'approche agnostique est qu'elle est entièrement automatique et indépendante du domaine. Elle réduit l'intervention humaine au minimum, en réduisant, par la même, les coûts. Toutefois, si les concepts extraites ont une interprétation statistique, ils n'ont, en revanche, aucun sens linguistique. L'interprétation et l'explication sont, par conséquent, rendues complexes. En outre, d'une part, cette approche ne permet pas de mettre à jour des relations riches entre les concepts telles que celles que nous pourrions trouver dans les ontologies. D'autre part, les méthodes statistiques d'extraction de concepts étant basées sur les fréquences de co-occurrences, et donc sur l'espace des termes, l'espace sémantique obtenu hérite de certains problèmes de l'espace des termes. En effet, par exemple, il ne sera plus possible de représenter, dans l'espace sémantique, des documents de langues différentes comme cité plus haut.

L'approche a priori semble plus justifiée au niveau linguistique. Étant donné l'utilisation de connaissances externes adaptées spécifiquement au problème, les résultats sont souvent plus précis. Cependant, les systèmes développés sont dépendants du domaine et donc, rarement, généralisables. De plus, cette approche nécessite une intervention humaine importante en amont pour la définition des thésaurus et des ontologies.

Dans la suite de cette partie, nous nous intéressons uniquement à l'approche agnostique et plus précisément à l'extraction d'information latente à partir de méthodes basées sur la LSA. Nous justifions ce choix par le fait que nous souhaitons nous intéresser à des systèmes autonomes capables d'acquérir des connaissances par apprentissage. En outre, l'avantage majeur des méthodes basées sur la LSA est que ces méthodes sont non-supervisées, à savoir qu'elles n'utilisent pas l'information concernant les étiquettes des documents.

5.2.2 Le modèle d'espace vectoriel généralisé (GVSM)

Le modèle d'espace vectoriel généralisé (GVSM) a été proposé dans [WZW85]afin d'introduire la notion de similarité sémantique dans le modèle d'espace vectoriel standard (VSM). Ainsi, deux termes t_1 et t_2 sont dits sémantiquement liés si ces termes apparaissent fréquemment ensemble dans les documents. L'importance sémantique du lien sera déterminée par la somme des fréquences de co-occurrences des termes. Ainsi, l'importance du lien sémantique pour t_1 et t_2 dans un corpus de n documents ($\{d_1, \ldots, d_n\}$) sera donnée par :

$$\mathbf{t_1}.\mathbf{t_2} = \sum_{i=1}^{n} \mathrm{tf}(t_1, d_i).\mathrm{tf}(t_2, d_i) \tag{5.1}$$

avec $\mathbf{t_1} = (\mathrm{tf}(t_1, d_1), \ldots, \mathrm{tf}(t_1, d_n))$ et $\mathrm{tf}(t_1, d_1)$ la fréquence d'occurrence du terme t_1 dans le document d_1.

En faisant l'analogie avec la covariance, le lien sémantique représente la corrélation entre les fréquences d'occurrences de t_1 et de t_2. Nous pouvons noter qu'il n'est pas nécessaire que ces termes apparaissent ensemble dans deux documents pour que ces derniers soient jugés similaires. En effet, la similarité sera, plutôt, déterminée par la co-distribution des termes au sein

du corpus.

En utilisant, l'équation 5.1, nous pouvons définir le noyau GVSM [STC04] pour deux documents d_1 et d_2 comme étant :

$$k_{GVSM}(d_1, d_2) = \sum_{i,j} \text{tf}(t_i, d_1).\text{tf}(t_j, d_2)\mathbf{t_i}.\mathbf{t_j} \tag{5.2}$$

La notation matricielle est :

$$k_{GVSM}(d_1, d_2) = \phi(d_1)DD^T\phi(d_2)^T \tag{5.3}$$

avec $\phi(d_1) = (\text{tf}(t_1, d_1), \ldots, \text{tf}(t_k, d_1))$ et D la matrice de fréquences des termes par document.

Nous pouvons remarquer que le GVSM ne traite que partiellement la synonymie. En effet, il y a deux cas dans lesquelles deux termes synonymes peuvent apparaître. Le premier cas est lorsque l'auteur d'un document souhaite éviter les répétitions. Il emploiera ainsi des synonymes. Les synonymes auront dans ce cas une fréquence de co-occurrence élevée. Le deuxième cas est lorsqu'il existe des termes sémantiquement proches dans le sens linguistique dont certains seront utilisés dans certains documents et les autres dans d'autres documents. Le GVSM peut, ainsi, détecter le premier cas grâce à la co-occurrence. Par contre, il ne pourra pas traiter le deuxième cas puisque la fréquence de co-occurrence sera, cette fois, très faible. Pour gérer, le deuxième cas, ainsi que la polysémie, il faudra tenir compte des fréquences de co-occurrences du second ordre ou supérieur comme nous le verrons plus bas pour la LSA.

5.2.3 L'Analyse Sémantique Latente (LSA)

L'analyse sémantique latente (LSA) [DDL+90] consiste à découvrir des structures sémantiques latentes dans un corpus de documents. Pour cela, les informations de co-occurrences des termes sont utilisées pour découvrir des relations. Ainsi, si deux termes apparaissent fréquemment ensembles (nombre de co-occurrences important) alors ils ont plus de chance d'exprimer le même concept (synonymes). De même, soit trois termes t_1, t_2 et t_3, si les couples (t_1, t_2) et (t_1, t_3) ont un nombre de co-occurrences important mais que le couple (t_2, t_3) a un nombre de co-occurrences faible alors le terme t_1 exprime plusieurs concepts selon le contexte (polysémie).

La LSA utilise le même principe que le GVSM mais avec une technique d'extraction plus perfectionnée. La LSA définit un espace de concept dans lequel les termes ayant une co-occurrence importante sont projetés, par une combinaison linéaire, sur un même axe représentant un concept. Ainsi, la matrice de co-occurrences sera représentée par une matrice diagonale dont les valeurs représentent "l'importance" du concept. La LSA utilise une décomposition de la matrice des termes par document, D, en valeurs singulières (SVD) :

$$D = U\Sigma V^T \tag{5.4}$$

Les matrices U et V sont des matrices orthonormées. U est la matrice de passage de l'espace vectoriel (VSM) à l'espace des concepts (U relie les termes aux concepts). De même, V est la matrice de passage de l'espace des termes (dont le repère est défini par l'ensemble des documents du corpus) à l'espace des concepts (V relie les documents aux concepts). Les vecteurs colonnes de U (resp. V) étant les vecteurs propres de DD^T (resp. D^TD).
La fonction ϕ_{LSA} projetant un document dans l'espace des concepts est :

$$\phi_{LSA}(d) = \phi(d)U \tag{5.5}$$

Le noyau LSA [CSTL02] est alors défini par :

$$k_{LSA}(d_1, d_2) = \langle \phi_{LSA}(d_1), \phi_{LSA}(d_2) \rangle = \phi(d_1)UU^T\phi(d_2)^T \tag{5.6}$$

Il est possible de tenir compte de la matrice diagonale Σ exprimant l'importance de chaque concept dans le corpus. La fonction de projection devient alors :

$$\phi_{\Sigma LSA}(d) = \phi(d)U\Sigma^{\frac{1}{2}} \tag{5.7}$$

Le noyau LSA pondéré est alors :

$$k_{\Sigma LSA}(d_1, d_2) = \phi(d_1)U\Sigma U^T\phi(d_2)^T \tag{5.8}$$

En outre, la méthode SVD utilise une décomposition en minimisant la somme des carrés de la différence entre le point d'origine et le point projeté. Il est alors ainsi possible de réduire le nombre de concepts en éliminant les vecteurs propres associés aux valeurs propres les plus faibles. Il suffira ainsi de remplacer, dans les équations précédentes, les matrices Σ, U et V par, resp., Σ^k, U^k et V^k correspondant aux k valeurs singulières les plus importantes et aux k vecteurs singuliers associés. Ainsi, seules les relations (concepts) les plus importantes définiront l'espace sémantique. La valeur de k, à savoir la dimension de l'espace sémantique, doit être fixée de manière empirique.

La LSA doit, essentiellement, son succès à deux propriétés principales :

1) la LSA est une technique de réduction de dimension, lorsqu'on choisit k inférieur au nombre de documents utilisés pour la SVD. Cette réduction permet de diminuer le bruit résidant dans les données d'origine,

2) La décomposition de la matrice de termes par document permet de mettre en lumière les relations de co-occurrence entre les termes. En effet, la SVD permet de tenir compte, non seulement, des co-occurrences du premier ordre, comme la GVSM, mais aussi les co-occurrences d'ordre supérieur. Dans le premier cas, deux termes apparaissant souvent ensemble se situeront à proximité l'un de l'autre dans l'espace sémantique. Dans le second cas, deux termes t_1 et t_2 n'ont pas besoin d'apparaître ensemble dans un document. t_1

peut apparaître avec un ensemble de mots S_1, appelé contexte, et t_2 avec un ensemble S_2. Si S_1 et S_2 partagent des termes similaires alors t_1 et t_2 possèdent une relation de co-occurrence de second ordre. Par conséquent, t_1 et t_2 seront très proches l'un de l'autre dans l'espace sémantique en fonction du nombre et de la fréquence d'occurrence des termes partagés dans le contexte. Cette propriété permet de gérer la synonymie entre les termes car les termes synonymes ont des contextes similaires et peuvent être capturés par une analyse de co-occurrence d'ordre supérieur. De même pour la polysémie, si les couples (t_1, t_2) et (t_1, t_3) ont une fréquence de co-occurrence élevée mais que le couple (t_2, t_3) a une fréquence faible alors la SVD n'introduira pas t_1, t_2 et t_3 dans la même relation. Les deux couples seront, ainsi, traités différemment. En effet, il est vrai que les termes polysémiques possèdent des contextes très différents.

Ces propriétés font, qu'en général, les documents sont mieux traités dans l'espace sémantique latent que dans le VSM. En effet, il a été montré dans [CSTL02] que le noyau LSA obtient les mêmes performances, avec un espace de faible dimension, que les noyaux dans le VSM étudié dans [Joa02, Joa98].

Toutefois, le problème principal des méthodes LSA est que les concepts extraits à partir de la transformation SVD sont des concepts artificiels, à savoir statistiques. Il est, ainsi, très difficile de les interpréter.

5.2.4 L'Analyse Sémantique Latente Locale

La LSA est une méthode globale non supervisée. Par conséquent, elle ne tient pas compte de la classe des documents (non-supervisée) et traite tous les documents de la même manière (globale). Ainsi, les classes ne contenant que très peu de documents risquent d'être éliminées. En effet, les documents, appartenant à des petites classes, vont utiliser des termes spécifiques qui auront une fréquence d'occurrence faible. Ces documents seront alors essentiellement représentés, dans l'espace sémantique, par des vecteurs propres associés à des valeurs propres faibles. Ces vecteurs propres risquent d'être éliminés lors de la phase de réduction de dimension (sélection du nombre de dimension de l'espace sémantique) [WPW95]. Ceci aura un impact négatif sur les performances du système et plus spécifiquement sur les macro-performances.

Dans [Hul94], la LSA locale a été introduite pour tenir compte des catégories peu fréquentes. L'idée est de remplacer la LSA globale par plusieurs LSA locales qui se focaliseront sur des régions critiques. Les régions critiques sont définies comme étant les zones où se trouvent les documents appartenant à des catégories peu fréquentes. En effectuant une LSA locale à chaque région critique, il sera alors possible de capturer les concepts importants des catégories de cette région et, ainsi, de construire un espace sémantique suffisamment riche pour regrouper, dans l'espace, les documents de même catégorie et d'avoir, par la même, un espace discriminant. Toutefois, dans [Hul94], chaque région critique est définie par des documents de la même

catégorie. Ainsi, bien que la LSA locale ait permis d'extraire les concepts fondamentaux de cette catégorie, elle a été incapable d'extraire des concepts discriminants. La frontière entre les documents de cette catégorie et les autres n'a pas été prise en compte dans la définition de l'espace sémantique. Les performances du systèmes ont alors été fortement biaisées par le manque d'information discrimintante de l'espace sémantique.

Ce problème a ensuite été géré dans [Hul95, SHP95, WPW95] en élargissant les zones critiques. En effet, dans chaque zone, des documents non pertinents sont ajoutés à l'ensemble des documents pertinents pour mettre l'accent sur la frontière, à savoir les termes discriminants. Les expériences ont montré que la LSA locale, avec la nouvelle définition de la zone critique, obtient des performances nettement supérieur à la LSA globale. Les principales stratégies utilisées pour définir les régions critiques sont les suivantes :

- **Sélection de documents pertinents :** il s'agit de la technique de base qui consiste à ne sélectionner, pour la région, que les documents pertinents. Par exemple, pour une catégorie, seuls les documents de cette catégorie seront utilisés pour définir la région [Hul94].

- **Sélection par requête de termes prédictifs :** cette stratégie a été définie dans [WPW95]. Une requête, à savoir un document artificiel, est créée pour une catégorie c en sélectionnant n termes ayant les scores de discrimination les plus élevés. Le score de discrimination peut être calculé selon plusieurs critères. Dans [WPW95], le score de pertinence est utilisé. Ce score, pour un terme t, est donné par $RS(t,c) = \log \frac{P(t|c)+d}{P(t|\overline{c})+d}$, avec d une constante. La région critique est alors définie, pour c, comme étant composée des k documents les plus similaires à la requête. La similarité est simplement calculée par un produit scalaire entre les documents dans le VSM.

- **Sélection par expansion de requête :** cette stratégie est généralement utilisée lorsqu'une requête initiale est déjà définie. La requête est étendue en ajoutant des termes et en tenant compte des documents pertinents et non pertinents. Pour étendre la requête la méthode d'expansion de Rocchio [Roc71, BSA94] est utilisée. Dans cette méthode, les X termes les plus fréquents dans les documents pertinents seront ajoutés à la requête initiale. Toutefois les termes seront ajoutés après pondération selon la formule suivante :

$$Q^{new} = \alpha.Q^{old} + \beta\frac{1}{|D_P|}\sum_{d\in D_P} d - \gamma\frac{1}{|D_N|}\sum_{d\in D_N} d \tag{5.9}$$

  avec Q^{old} la requête initiale, D_P l'ensemble des documents pertinents, D_N l'ensemble des documents non-pertinents et α, β et γ des constantes à fixer. La requête et les documents sont exprimés dans le VSM. En outre, seuls les X termes les plus fréquents sont pris en compte dans d pour $d \in D_P \cup D_N$.
  Une fois la requête étendue, les k documents les plus similaires à la requête étendue sont utilisés pour former la région critique. De même que précédemment, la similarité entre un document et la requête est déterminée par le produit scalaire dans le VSM. Cette

stratégie a été utilisée dans [SHP95].

– **Sélection par SVM :** pour cette approche, un classifieur SVM est entraîné sur les documents d'apprentissage, pour une classe c. Puis le classifieur est utilisé sur les documents pour leur attribuer un score de confiance. Les k documents ayant les scores les plus élevés sont utilisés pour former la région critique pour c. Cette stratégie a été utilisée dans [LCZ+04].

Dans [LCZ+04], des expériences ont montré, notamment sur le corpus Reuters, que les performances avec la LSA locale sont toujours supérieures à la LSA globale. La stratégie de sélection par requête de termes prédictifs a obtenu de meilleures performances que les autres stratégies. Le critère de sélection des termes pour cette stratégie était l'information mutuelle et le χ^2. L'information mutuelle a été légèrement plus performante que le χ^2.

En outre, une LSA locale pondérée (LRW-LSI) a été présentée dans [LCZ+04]. Dans les versions précédentes, une région critique est définie en lui affectant ou non un document. Ceci peut se traduire par une pondération binaire : les documents affectés à la zone critique reçoivent une pondération de 1 et les autres une pondération de 0. Dans [LCZ+04], une stratégie de pondération plus douce est adoptée. En effet, un document d_i sera pondéré en fonction de son score de pertinence pour la catégorie c définissant la zone critique. Le score de pertinence de d_i, $rs(d_i, c)$, est calculé par un classifieur tel que le SVM pour c. La pondération w_i pour d_i est alors définie par une sigmoïde selon la formule :

$$w_i = f(d_i, c) = \frac{1}{1 + e^{-a(rs(d_i,c)+b)}} \tag{5.10}$$

avec a et b des paramètres de la sigmoïde contrôlant le lissage de la courbe.
Les n documents ayant les scores de pertinence les plus élevés sont alors utilisés pour définir la région critique de c.
L'algorithme 5.2.1 illustre la phase d'apprentissage d'un classifieur pour une catégorie c dans un espace obtenu par la LSA locale pondérée. L'algorithme 5.2.2 illustre, quant à lui, la phase de classement d'un document dans la catégorie c.

Algorithme 5.2.1 Algorithme d'apprentissage LRW-LSI pour la classe c

1: Attribuer un score de pertinence avec le classifieur initial de c pour chaque document d'apprentissage
2: Pondérer chaque document en fonction du score de pertinence avec l'équation 5.10
3: Les n documents ayant les scores de pertinence les plus élevés sont utilisés pour définir la région critique de c
4: Effectuer la LSA locale avec les documents de la zone critique
5: Projeter tous les documents d'apprentissage dans l'espace sémantique local
6: Utiliser cet espace sémantique pour l'apprentissage d'un classifieur pour la catégorie c

Algorithme 5.2.2 Algorithme de classification LRW-LSI pour un document d et pour la classe c

1: Attribuer un score de pertinence pour d avec le classifieur initial de c
2: Pondérer d avec l'équation 5.10
3: Projeter d dans l'espace sémantique local de c
4: Utiliser le classifieur "appris" dans cet espace sémantique pour attribuer ou non la classe c à d

Dans [LCZ+04], le classifieur SVM a été utilisé en tant que classifieur inital et pour le classement dans les espaces sémantiques locaux. Les résultats sur le corpus Reuters montrent très clairement que la LSA locale pondérée (LRW-LSI) permet d'obtenir les mêmes performances, voire légèrement supérieures, qu'aux SVM dans le VSM classique avec une réduction de dimension importante. En effet, le VSM avait une dimension supérieur à 5000 alors que les espaces sémantiques locaux avaient une dimension inférieur à 200.

Bien que la LSA locale donne de très bon résultats, l'inconvénient majeur est le temps de calcul, qui est suffisamment important, pour effectuer la décomposition SVD. Pour diminuer ce temps, il est possible de réduire le nombre de régions critiques. Dans [WPW95], des "méta-classes" ont été définies manuellement pour regrouper des classes voisines. Une région critique a été, ensuite, définie pour chaque méta-classe. Ainsi, le nombre de LSA locales a pu être diminué. Toutefois, il s'agissait, ici, d'un compromis. Par conséquent, bien que les performances aient été satisfaisantes (meilleures que celles de la LSA globale), elles ont été légèrement plus faibles que la LSA locale sur chaque catégorie. Cette approche entraîne, elle aussi, une autre problématique qui est de déterminer le nombre de méta-classes nécessaires. Cette problématique n'est pas nouvelle puisqu'elle se pose régulièrement dans le domaine de l'apprentissage non-supervisé et plus particulièrement dans les méthodes de classification (*clustering*).

5.2.5 Le modèle d'espace vectoriel de domaine (*Domain VSM*)

Le modèle d'espace vectoriel de domaine (*Domain VSM*)[GS05] est une spécialisation du modèle d'espace vectoriel généralisé (GVSM). Le GVSM utilise l'information de co-occurrence provenant de la matrice des termes par document,D, pour représenter un document. En effet, dans cet espace, un document est représenté par une combinaison linéaire des documents de la matrice de D. Ainsi, en reprenant l'équation 5.3, la matrice de passage du VSM vers le GVSM est déterminée par D. La fonction de passage d'un document d vers le GVSM est alors donnée par :

$$\phi_{GVSM}(d) = \phi(d).D \tag{5.11}$$

avec $\phi(d) = (\mathrm{tf}(t_1, d), \ldots, \mathrm{tf}(t_k, d))$ la représentation de d dans le VSM et D la matrice des k termes du VSM par document.

Ce qui revient, sous forme vectorielle à :

$$\phi_{GVSM}(d) = \sum_i \langle \phi(d), \phi(d_i) \rangle . \phi(d_i) \tag{5.12}$$

avec d_i les documents utilisés pour composer d. Le GVSM est ainsi défini par les documents de D.

Dans le *Domain VSM*, la matrice de termes par documents, D, est remplacée par une matrice de termes par domaine, D_D. La matrice D_D servira ainsi de matrice de passage du VSM vers un espace de domaine : le *Domain VSM*. Cette matrice définit le degré de relation entre les termes du VSM et les domaines sémantiques du *Domain VSM*. Le tableau 5.1 donne un exemple de matrice de domaine pour un espace de domaine composé de deux dimensions : médecine et informatique. Les termes sont ainsi associés à chaque domaine selon un degré de relation. Dans cet espace, tous les documents seront représentés par une combinaison linéaire des domaines sémantiques informatique et médecine.

	Médecine	**Informatique**
HIV	1	0
SIDA	1	0
Virus	0.5	0.5
Ordinateur	0	1

Tab. 5.1 – Un exemple de matrice de domaine.

La représentation vectorielle $\phi_{DVSM}(d)$ d'un document d dans l'espace de domaines sémantiques est donnée par :

$$\phi_{DVSM}(d) = \phi(d) I^{IDF} D_D \tag{5.13}$$

avec I^{IDF} une matrice diagonale dont chaque cellule $I^{IDF}_{i,i}$ correspond à la pondération IDF pour le terme t_i. Cette pondération permet de donner une plus grande importance aux termes rares d'un corpus qu'aux termes fréquents dans le corpus.

Une fois la définition du *Domain VSM* formalisée, le problème réside dans la définition de la matrice de termes par domaines sémantiques. La question est de savoir comment définir ces domaines. L'approche agnostique et l'approche a priori peuvent ainsi être utilisées pour répondre à cette question. Dans [GS05], l'approche agnostique a été adoptée. Les domaines sémantiques sont obtenus par la LSA. Ainsi, les domaines sémantiques sont définis comme étant des concepts latents. La matrice des termes par domaines est définie par :

$$D_D = I^N U^k \sqrt{\Sigma^k} \tag{5.14}$$

avec $D = U \Sigma V^T$, D la matrice des termes par documents, U^k les k vecteurs singuliers de D associés aux k valeurs singulières les plus élevées (σ^k) et I^N une matrice diagonale avec

$I^N_{i,i} = \frac{1}{\sqrt{\langle \mathbf{w_i}, \mathbf{w_i} \rangle}}$ où $\mathbf{w_i}$ est le vecteur donné par la $i^{\text{ème}}$ ligne de $(U^k\sqrt{\Sigma^k})$.
La normalisation fixée par I^N permet d'attribuer un même poids pour chaque terme dans ses relations avec les différents domaines. Ainsi, les termes fréquents et les termes peu fréquents seront sur une même échelle.

Le noyau pour l'espace de domaine sémantique est défini par :

$$k_{DVSM}(d1, d2) = \frac{\langle \phi_{DVSM}(d_1), \phi_{DVSM}(d_2) \rangle}{||\phi_{DVSM}(d_1)||.||\phi_{DVSM}(d_2)||} \tag{5.15}$$

Les expériences de catégorisation de textes, dans [GS05], sur le corpus Reuters-21578 et 20-Newsgroups ont montré que ce noyau est plus performant que le noyau linéaire du VSM (noyau sac-de-mots). En outre, il a été montré que le noyau du *Domain VSM* est très efficace pour l'apprentissage à partir d'une faible quantité de données.

5.2.6 La LSA probabiliste

L'un des inconvénients principaux de la LSA est que les concepts obtenus par la décomposition SVD sont difficilement interprétables. Pour répondre à ce problème, la LSA probabiliste (PLSA) a été présentée dans [Hof99a, Hof99b, Hof01]. La PLSA est une méthode générative qui se base sur l'introduction de variables cachées ou encore appelées variables latentes. Les variables sont introduites selon un modèle statistique appelé "modèle d'aspect" [HP98, HPJ99]. Le modèle d'aspect permet de modéliser des co-occurrences entre les données en introduisant des variables latentes. Dans le cas des documents textes, la PLSA permet de modéliser la co-occurrence d'un terme $t \in \mathcal{W}$ et d'un document $d \in \mathcal{D}$ avec des variables latentes $z \in \mathcal{Z}$. Les figures 5.1 et 5.2 illustrent le modèle d'aspect. Dans la figure 5.1, une utilisation asymétrique est illustrée. A savoir que la co-occurrence de d et t est modélisée en affectant une variable latente à d puis en générant t à partir de z. Plus formellement, on a :

$$\begin{aligned} P(d,t) &= P(d)P(t|d) \\ &= P(d)\sum_{z\in\mathcal{Z}} P(t,z|d) \\ &= P(d)\sum_{z\in\mathcal{Z}} \frac{P(t,d|z)P(z)}{P(d)} \end{aligned} \tag{5.16}$$

En faisant, l'hypothèse d'indépendance conditionnelle entre t et d sachant z (à savoir $P(t,d|z) = P(t|z)P(d|z)$), on a :

$$\begin{aligned} P(d,t) &= P(d)\sum_{z\in\mathcal{Z}} \frac{P(t|z)P(d|z)P(z)}{P(d)} \\ &= P(d)\sum_{z\in\mathcal{Z}} P(z|d)P(t|z) \end{aligned} \tag{5.17}$$

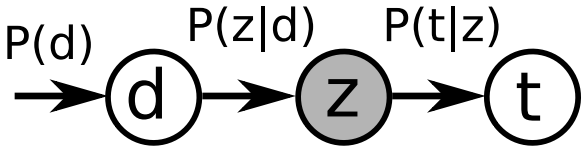


Fig. 5.1 – Modèle d'aspect de la PLSA avec une paramétrisation asymétrique.

De même, l'utilisation asymétrique illustrée par la figure 5.2 peut être formalisée par la formule suivante :

$$\begin{aligned} P(d,t) &= \sum_{z\in\mathcal{Z}} P(d,t|z)P(z) \\ &= \sum_{z\in\mathcal{Z}} P(z)P(d|z)P(t|z) \end{aligned} \tag{5.18}$$

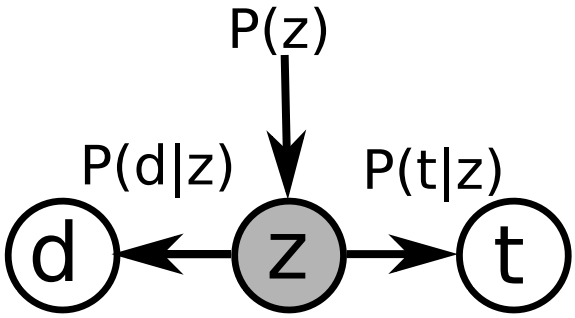


Fig. 5.2 – Modèle d'aspect de la PLSA avec une paramétrisation symétrique.

Les probabilités liées à la variable latente peuvent ensuite être déterminées en utilisant l'algorithme EM [MK97]. Pour cela, il est nécessaire de maximiser la fonction suivante :

$$\begin{aligned} \mathcal{L}(\theta) &= \log P(\mathcal{D}|\theta) \\ &= \sum_i \sum_j \mathrm{tf}(t_j,d_i) \log P(d_i,t_j) \end{aligned} \tag{5.19}$$

avec $\mathrm{tf}(t_j,d_i)$ la fréquence d'occurrence du terme t_j dans d_i et θ l'ensemble des paramètres du modèle à savoir $\theta = \{P(z), P(t|z), P(d|z)\}$
La maximisation de cette fonction par l'algorithme EM est définie en deux étapes itératives. La première est l'estimation de $P(z|d,t)$ qui est donnée par :

$$P(z|d,t) = \frac{P(z)P(d|z)P(t|z)}{\sum_{z'\mathcal{Z}} P(z')P(d|z')P(t|z')} \tag{5.20}$$

La seconde étape est la maximisation qui permet de mettre à jour θ avec l'estimation de

$P(z|d,t)$. Nous obtenons alors :

$$\begin{aligned} P(t|z) &\leftarrow \frac{\sum_d \mathrm{tf}(t,d)P(z|d,t)}{\sum_{d,t} \mathrm{tf}(t,d)P(z|d,t)} \\ P(d|z) &\leftarrow \frac{\sum_t \mathrm{tf}(t,d)P(z|d,t)}{\sum_{d,t} \mathrm{tf}(t,d)P(z|d,t)} \\ P(z) &\leftarrow \frac{\sum_d \sum_t \mathrm{tf}(t,d)P(z|d,t)}{\sum_{d,t} \mathrm{tf}(t,d)} \end{aligned}$$

Les deux étapes sont répétées jusqu'à convergence. En outre, pour éviter les optimums locaux et améliorer la généralisation, l'utilisation d'une variante du recuit simulé est proposée dans [Hof99a, Hof99b, Hof01].

Le noyau pour la PLSA est défini par :

$$k(d_1, d_2) = \sum_k P(d_1|z_k)P(d_2|z_k)P(z_k) \tag{5.21}$$

Il est possible de faire l'analogie avec la LSA. La matrice de termes par documents, D, est décomposée en $D = U\Sigma V^T$. Σ correspondrait à une matrice diagonale avec $\Sigma_{i,i}$ qui serait associée à $P(z_i)$. De même, nous avons $U_{i,j}$ qui correspondrait à $P(t_i|z_j)$ et $V_{i,j}$ qui correspondrait à $P(d_i|z_j)$. Plus généralement, il a été montré que la PLSA est une instance de l'ACP multinomiale [GG05, Bun02].

L'avantage majeur de la PLSA réside dans sa théorie statistique induite par le modèle d'aspect alors que la LSA est basée sur la SVD qui est difficilement interprétable. Toutefois, il n'a pas été clairement établi que la PLSA obtienne de meilleures performances que la LSA.

5.2.7 Discussion

Dans cette partie du chapitre, nous avons présenté des méthodes pour extraire et exploiter des concepts sous-jacents aux données textuelles. Nous nous sommes essentiellement focalisés sur l'Analyse Sémantique Latente. Nous avons vu que la LSA et les méthodes dérivées réussissent à améliorer les performances des systèmes de traitement. Cette amélioration peut être expliquée par deux facteurs. Le premier facteur est l'extraction des relations liant les termes avec de fortes co-occurrences. Ces relations permettent de définir un espace sémantique plus adapté aux traitements de documents notamment en tenant compte de la synonymie. Le second facteur est la réduction de dimension obtenue en limitant la définition de l'espace sémantique aux relations les plus significatives. Ainsi, le bruit contenu dans les données d'origine se trouve éliminé dans l'espace sémantique étant donné qu'il est représenté par des relations associés aux termes de faible co-occurrence.

L'inconvénient de la LSA réside essentiellement dans la difficulté d'interprétation des concepts statistiques induits par les relations linéaires entre les termes extraites par une décomposition SVD. L'Analyse Sémantique Latente Probabiliste (PLSA), quant à elle, se base sur un modèle

statistique bien établi, le modèle d'aspect. L'interprétation est rendue plus aisée. En effet, dans ce modèle, les concepts latents peuvent être perçus comme des *clusters* ou comme une source générative.

5.3 **Un modèle d'espace vectoriel de concepts pour noyaux sémantiques**

Les mesures de similarité sont des éléments clés dans les algorithmes de traitement automatique des langues. Elles sont utilisés pour orienter le processus d'extraction de connaissance. Ainsi, elles sont les principales responsables des performances d'un algorithme. Si une mesure de similarité pertinente améliorera les performances, une mauvaise mesure risque de mener à des résultats incohérents. La définition d'une bonne mesure n'est pas un processus aisé. En effet, la mesure doit donner une bonne indication sur le degré de similarité entre deux documents. La notion de sémantique n'est pas clairement définie. Bien que nous essayons d'imiter la perception humaine, l'information sémantique peut prendre différente forme selon l'approche adoptée. Il existe deux grandes approches : l'une basée sur l'information statistique telle que la fréquence de co-occurrence et l'autre basée sur des sources de connaissances externes telles que les ontologies.

Dans cette partie du chapitre, nous présentons un modèle d'espace vectoriel de concepts (CVSM) pour la représentation des documents textuels. Cette représentation, induite par des connaissances a priori, est présentée, ici, comme une alternative au modèle classique d'espace vectoriel (VSM). Nous avons vu à la section 3.2.2.1 que le VSM est basé sur la fréquence d'occurrence des termes. Pour le CVSM, une taxonomie de concepts linguistiques est utilisée comme source de connaissances pour définir l'espace vectoriel. De plus, nous proposons deux noyaux basés sur les concepts. Le premier noyau, le noyau CVSM linéaire, est défini dans le CVSM où chaque document est représenté par un vecteur de concept intégrant l'information sur la taxonomie des concepts. Le second noyau, le noyau CVSM latent, mélange l'approche agnostique basée sur l'information statistique et l'approche a priori utilisant des connaissances externes propres au domaine. Basé sur le noyau LSA [CSTL02], le noyau CVSM latent utilise une décomposition en valeurs singulières (SVD) pour découvrir des structures latentes entre les concepts linguistiques du CVSM.

L'utilisation des noyaux CVSM est illustrée par une tâche de catégorisation de texte dans le domaine biomédical. L'*Unified Medical Language System* (UMLS) est utilisé en tant que source a priori de connaissances biomédicales pour l'extraction de concepts à partir de documents textuels. Les performances de ces noyaux sont évaluées sur cette tâche en utilisant un classifieur SVM. Le corpus Ohsumed qui est connu pour être un corpus difficile, est utilisé pour l'évaluation expérimentale.

5.3.1 Noyau linéaire du modèle d'espace vectoriel de concepts

5.3.1.1 Pré-traitement des documents

Les documents textuels sont formatés pour une utilisation humaine. Ainsi, ils contiennent une riche variété de symboles et de protocoles tels que les règles typographiques. Ces informations sont ajoutées pour rendre la lecture et la compréhension plus aisées. Toutefois, le traitement automatique de ces données est rendu compliqué. En effet, si les éléments d'un document ne sont pas correctement gérés, ils peuvent être une source de bruits et d'ambiguïtés qui entraînera inexorablement à une baisse des performances du système. Une façon d'éviter ce problème est de pré-traiter le document pour avoir une représentation adaptée au système de traitement.

Dans le cadre de ce travail, nous utiliserons le pré-traitement présenté à la section 4.3. Bien que ce pré-traitement ait été donné pour le domaine médical, il peut être utilisé pour n'importe quel domaine en utilisant, simplement, la source de connaissances appropriées. En outre, la forme arborescente obtenue après le pré-traitement ne sera pas exploitée ici.

5.3.1.2 Le modèle d'espace vectoriel de concepts

Le modèle d'espace vectoriel (VSM), présenté à la section 3.2.2.1, est basé sur la forme morphologique des termes. Il est, ainsi, hautement dépendant de la langue du texte et de la fréquence d'occurrences des termes. En effet, le VSM utilise simplement la fréquence des termes pour capturer l'information d'un document. Il est, ainsi, limité par le fait qu'il ne peut correctement gérer les termes synonymes et les termes polysémiques. De plus, les liens entre les termes sémantiquement proches ne peuvent être modélisés. Toutefois, l'espace à haute dimension induit par le VSM permet aux systèmes, qui l'utilisent, d'obtenir des performances qui sont parmi les meilleures [Joa98].
Dans cette section, nous présentons le modèle d'espace vectoriel de concepts (CVSM), basé sur le VSM. Ce modèle devrait permettre de gérer les problèmes, rencontrés par le VSM, énumérés ci-dessus. Le CVSM est un espace vectoriel dans lequel chaque axe représente un concept défini dans un dictionnaire de concepts. Le dictionnaire est constitué des concepts définis dans un thésaurus et des mots racines, obtenus par *stemming*, lorsque ces mots ne peuvent être associés à des concepts. Nous prenons, alors, l'hypothèse que ces mots expriment un concept à part entière. Dans le CVSM, les documents sont pré-traités selon la méthode décrite plus haut. La figure 5.3 montre un exemple de document pré-traité.

Une fois qu'un document d a été pré-traité, chaque unité lexicale l du document d est associée à un vecteur de concepts local $\phi(l)$. Le $i^{\text{ème}}$ composant $\phi_i(l)$ de $\phi(l)$ associé au concept c_i est alors donné par :

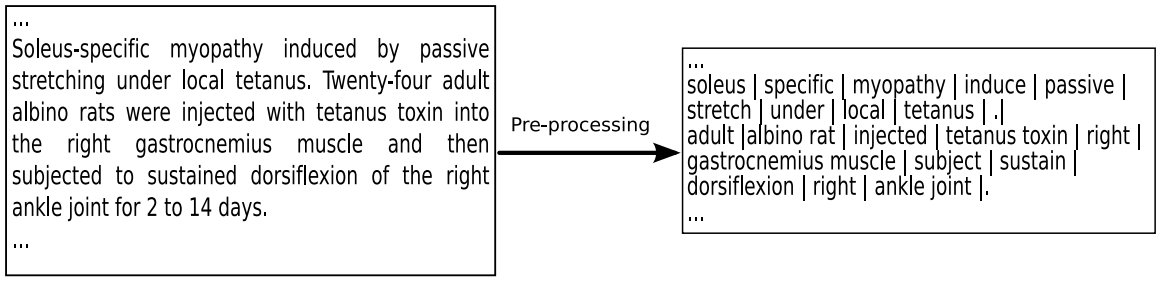


Fig. 5.3 – Un exemple de document pré-traité.

$$\phi_i(l) = \sum_{t \in \mathcal{N}(l)} \frac{\phi_i^n(t)}{||\phi^n(t)||} \tag{5.22}$$

où $\mathcal{N}(t)$ est l'ensemble des termes normalisés d'une unité lexicale l et $\phi_i^n(t)$ est le $i^{\text{ème}}$ composant, associé au concept c_i, du vecteur de concept $\phi^n(t)$ pour le terme normalisé t. $\phi_i^n(t)$ est défini par :

$$\phi_i^n(t) = \sum_{c \in \mathcal{C}(t)} \frac{\sum_{p \in \mathcal{P}(c)} \sigma_p(c, c_i)}{\sqrt{\sum_j (\sum_{p \in \mathcal{P}(c)} \sigma_p(c, c_j))^2}} \tag{5.23}$$

où

$$\sigma_p(c_i, c_j) = \begin{cases} (d_p(c_i, c_j) + 1)^{-\alpha} & \text{si } c_i, c_j \in p\,, \\ 0 & \text{sinon}\,, \end{cases} \tag{5.24}$$

$\mathcal{C}(t)$ est l'ensemble des concepts associés à t, $\mathcal{P}(c)$ est l'ensemble des chemins allant du concept c au concept racine dans la taxonomie, c'est à dire les chemins allant du concept spécifique c au concept le plus général, $d_p(c_i, c_j)$ est la distance entre le concept c_i et le concept c_j sur le chemin p et $\alpha \in [0, 1]$ est une valeur exprimant la puissance de décroissance. α est utilisé pour décroître l'influence des concepts généraux sur la représentation d'un terme. En effet, étant donné un terme t, un concept c associé à t, et p un chemin allant de c au concept le plus général (le concept racine), les concepts proches, en terme de distance, de c fourniront le sens principal de t comparés aux concepts éloignés de c. Les concepts généraux n'expriment pas clairement le sens pertinent de t et peuvent même mener à des ambiguïtés voire du bruit. Par conséquent, il peut être intéressant de diminuer le pouvoir expressif des concepts selon leurs distances par rapport au concept spécifique c. En fixant α à un, nous pouvons considérablement réduire l'expressivité des concepts généraux et en fixant α à zéro, nous pouvons donner le même pouvoir d'expression à tous les concepts en ne faisant aucune différence entre les concepts généraux et spécifiques. α doit être fixé de manière empirique en fonction du corpus.

Étant donné un ensemble de vecteurs de concepts locaux $\{\phi(l_1), \ldots, \phi(l_n)\}$ pour un document d, le vecteur de concept global $\mathbf{\Phi}(d)$ pour d est donné par la formule suivante :

$$\mathbf{\Phi}(d) = \sum_{i=1}^{n} \frac{1}{||\phi(l_i)||}.\phi(l_i) \tag{5.25}$$

La normalisation dans l'équation 5.25 est utilisée pour représenter des documents de longueurs différentes avec une même échelle. De plus, les concepts qui apparaissent dans beaucoup de documents du corpus, ne fourniront pas d'information utile pour la discrimination de documents alors que les concepts rares dans un corpus peuvent être significatifs. Par conséquent, nous proposons d'utiliser un vecteur pondéré $\mathbf{\Phi}^{IDF}(d)$ en utilisant la pondération IDF (*Inverse Document Frequency*). Ce vecteur est défini tel que :

$$\Phi_i^{IDF}(d) = -\log(\frac{card(\mathcal{D}(c_i))}{N}).\Phi_i(d) \tag{5.26}$$

où N est le nombre de documents dans le corpus et $\mathcal{D}(c_i)$ est l'ensemble des documents du corpus contenant le concept c_i.

La figure 5.4 illustre la modélisation d'un document textuel dans le CVSM.

5.3.1.3 Le noyau linéaire

Nous définissons le noyau linéaire dans le modèle d'espace vectoriel de concept comme ceci :

$$k_{CVSM}(d_1, d_2) = \frac{\langle \mathbf{\Phi}^{IDF}(d_1), \mathbf{\Phi}^{IDF}(d_2) \rangle}{||\mathbf{\Phi}^{IDF}(d_1)||.||\mathbf{\Phi}^{IDF}(d_2)||} \tag{5.27}$$

5.3.2 Le noyau CVSM latent

Nous avons vu, à la section 5.2.3, que l'analyse sémantique latente permettait de mettre à jour des relations entre des termes. Ces relations sont extraites par une décomposition linéaire en valeurs singulières de la matrice des fréquences des termes par document. Elles sont par conséquent des relations de co-occurrences des termes dans les documents. Les relations extraites sont appelés "concepts latents". La LSA permet, ainsi, non seulement de diminuer la dimension de l'espace en ne conservant que les relations les plus importantes mais aussi de gérer la polysémie.

Dans un même esprit, nous proposons, dans cette partie, l'utilisation de la LSA dans le CVSM pour capturer les structures latentes entre les concepts. Les concepts LSA peuvent être perçus comme des concepts abstraits de haut niveau. Il peut être intéressant d'analyser expérimentalement l'effet du mélange entre concepts linguistiques et concepts statistiques.
Nous définissons la matrice M de concepts par documents dans le CVSM comme ceci :

$$M = \begin{bmatrix} \frac{\Phi_1^{IDF}(d_1)}{||\mathbf{\Phi}^{IDF}(d_1)||} & \cdots & \frac{\Phi_1^{IDF}(d_n)}{||\mathbf{\Phi}^{IDF}(d_n)||} \\ \vdots & \ddots & \vdots \\ \frac{\Phi_m^{IDF}(d_1)}{||\mathbf{\Phi}^{IDF}(d_1)||} & \cdots & \frac{\Phi_m^{IDF}(d_n)}{||\mathbf{\Phi}^{IDF}(d_n)||} \end{bmatrix} \tag{5.28}$$

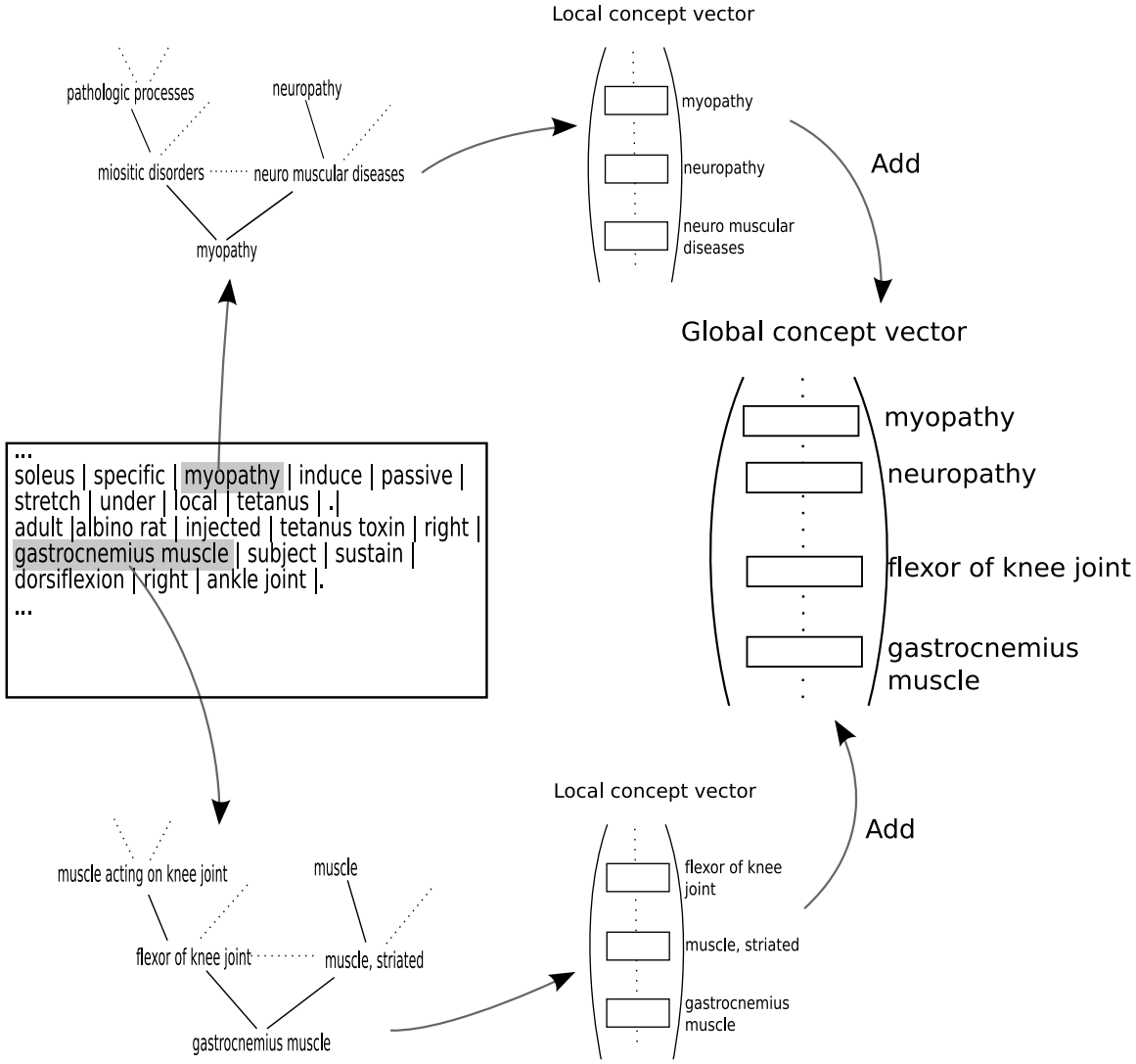


Fig. 5.4 – La représentation d'un document texte selon le modèle d'espace vectoriel de concepts.

avec m la dimension du CVSM (le nombre total de concepts) et n le nombre de documents. La LSA étant une méthode non-supervisée qui n'utilise donc pas l'information sur les étiquettes des documents, nous utiliserons les documents étiquetés (documents d'apprentissage) et les documents non-étiquetés (documents de test) pour M.
La SVD de M donne :

$$M = U\Sigma V^T \tag{5.29}$$

Nous dénotons par U_k, les k vecteurs singuliers associés aux k valeurs singulières les plus élevées et Σ_k la matrice diagonale contenant les valeurs singulières.
La projection du vecteur $\mathbf{\Phi}^{IDF}(d)$ du CVSM à l'espace sémantique latente est donnée par :

$$\mathbf{\Phi}^{lsa}(d) = \frac{\mathbf{\Phi}^{IDF}(d)}{||\mathbf{\Phi}^{IDF}(d)||}.U_k.\Sigma_k^{\frac{1}{2}} \tag{5.30}$$

Le noyau CVSM latent est, alors,, défini par :

$$k_{LCVSM}(d_1, d_2) = \frac{\langle \mathbf{\Phi}^{lsa}(d_1), \mathbf{\Phi}^{lsa}(d_2) \rangle}{||\mathbf{\Phi}^{lsa}(d_1)||.||\mathbf{\Phi}^{lsa}(d_2)||} \tag{5.31}$$

5.3.3 Évaluation expérimentale

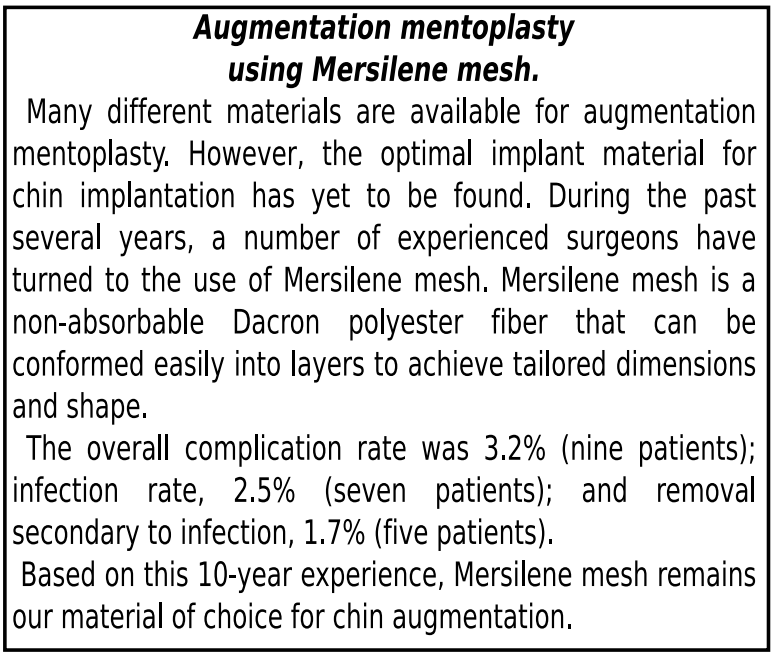
Nous avons mené plusieurs expériences sur un corpus médical pour évaluer la représentation CVSM. Nous avons utilisé le modèle d'espace vectoriel standard, VSM, comme base de comparaison. Dans cette section, nous présentons le corpus médical utilisé, le mode opératoire pour les expériences et un résumé des résultats expérimentaux.

5.3.3.1 Le corpus Ohsumed

Les expériences ont été menées sur le corpus Ohsumed qui est une collection de documents médicaux bien connue dans la littérature. Ce corpus est un sous-ensemble de la base bibliographique médicale "MEDLINE". La collection contient 348.566 références provenant de 270 journaux médicaux couvrant les années de 1987 à 1991. Nous avons suivi la préparation du corpus décrite dans [Joa02, Joa98]. Ainsi, à partir des références de l'année 1991, nous avons gardé uniquement les 20.000 références possédant un résumé. Les 10.000 premiers résumés sont utilisés pour l'apprentissage et le reste pour le test. Chaque document appartient à une ou plusieurs des 23 catégories de MeSH (*Medical Subject Headings*)[1] correspondant aux maladies cardio-vasculaire.

[1]les catégories MeSH sont les principales rubriques médicales. Elles définissent un système de classification pour le domaine médical.

L'ensemble d'apprentissage contient 6.286 documents distincts affectés à une ou plusieurs catégories (soit 10.000 documents "mono-catégorie"). La base de test contient 7.643 documents distincts associés à une ou plusieurs catégories. La figure 5.5 montre un exemple de document issu de ce corpus.



Augmentation mentoplasty using Mersilene mesh.

Many different materials are available for augmentation mentoplasty. However, the optimal implant material for chin implantation has yet to be found. During the past several years, a number of experienced surgeons have turned to the use of Mersilene mesh. Mersilene mesh is a non-absorbable Dacron polyester fiber that can be conformed easily into layers to achieve tailored dimensions and shape.

The overall complication rate was 3.2% (nine patients); infection rate, 2.5% (seven patients); and removal secondary to infection, 1.7% (five patients).

Based on this 10-year experience, Mersilene mesh remains our material of choice for chin augmentation.

Fig. 5.5 – Un exemple de document issu du corpus Ohsumed.

La catégorisation consiste à assigner, à chaque document de test, une ou plusieurs des 23 catégories de maladie cardio-vasculaire.

La tâche de catégorisation sur ce corpus est connue pour être difficile. En effet, les systèmes de catégorisation qui fonctionnent relativement bien sur les corpus tels que le Reuters-21578 et le 20-NewsGroups voient leurs performances diminuées. Par exemple, dans [Joa98], le classifieur linéaire SVM sur la VSM atteint une performance de 65.9% alors qu'il atteint une performance de 86% sur le corpus Reuters 21578. Les difficultés de catégorisation sont dues au fait que les données sont bruitées avec des termes médicaux très spécifiques et que les catégories ont un haut degré de corrélation.

5.3.3.2 La préparation des expériences

Dans toutes nos expériences, les documents ont été pré-traités selon la méthode définie à la section 5.3.1.1 pour la représentation selon le modèle d'espace vectoriel de concepts (CVSM). Pour le modèle d'espace vectoriel, VSM, nous avons utilisé le *stemming* pour réduire les mots fléchis à leurs bases communes. Nous avons, en outre, éliminés les mots vides. Pour le *stemming*, nous avons utilisé l'algorithme de Porter [Por80].

Pour la gestion du problème à multi-catégories. nous avons utilisé la stratégie "un-contre-tous". Cette stratégie a mené à la décomposition du problème principal en 23 sous-problèmes de catégorisation binaire. La librairie libSVM [CL01] a été utilisée pour l'apprentissage des classifieurs SVM.

Afin d'évaluer le CVSM, nous avons utilisé le VSM comme base de comparaison. Nous avons utilisé un noyau linéaire, dans le VSM, avec une pondération TF-IDF et une normalisation des vecteurs selon le mode opératoire defini dans [Joa98]. Ce noyau est nommé "noyau Sac-de-mots"(*Bag Of Words - BOW - kernel*). De plus, nous avons aussi utilisé un noyau LSA dans le VSM en utilisant l'équation 5.31. Ce noyau LSA est utilisé comme base de comparaison pour le noyau CVSM latent. Nous utiliserons la dénomination "*BOW SVD*" pour désigner le noyau LSA dans le VSM.

La mesure utilisée pour évaluer les performances des classifieurs est, principalement, la mesure F1 [Seb02] détaillée dans la section 3.2.4.

5.3.3.3 Évaluation de la puissance de décroissance α

Dans la première expérience, nous cherchons, empiriquement, la valeur optimale de la puissance de décroissance α pour le corpus Ohsumed. Nous rappelons que α contrôle la manière dont les concepts généraux sont pris en compte dans l'équation 5.24. Une valeur de zéro donnera un poids égal aux concepts généraux et spécifiques. Pour cette expérience, nous utilisons uniquement 10% des données d'apprentissage et 10% des données de test pour évaluer la performance. De plus, nous utilisons un échantillonnage stratifié pour conserver les proportions. La figure 5.6 montre les scores micro-F1 pour différentes valeurs de $\alpha \in [0, 1]$. Le meilleur score est obtenu pour $\alpha = 0.1$ avec une valeur micro-F1 de 53.9%. En outre, une meilleure performance est obtenue pour $\alpha = 0$ que pour $\alpha = 1$. Ce point signifie que les concepts généraux jouent un rôle important dans la tâche de catégorisation pour le corpus Ohsumed. C'est, effectivement, le cas lorsque plusieurs mots, avec un sens general commun, sont utilisés dans différents documents d'une même catégorie.

5.3.3.4 Évaluation du nombre de concepts latents

Pour les noyaux basés sur la LSA, la dimension de l'espace sémantique doit être fixée empiriquement. Par conséquent, nous avons mené un ensemble d'expériences sur le corpus entier en faisant varier le nombre de concepts latents, à savoir le nombre de vecteurs singuliers associés aux valeurs singulières les plus élevées. La figure 5.7 montre la variation du score micro-F1. Les noyaux CVSM latent et BOW SVD atteignent leurs performances quasi-optimales pour approximativement 2000 concepts latents. En fait, il y a une croissance rapide des performances pour un nombre de concepts allant de 0 à 1000. Ceci montre que les 1000 premiers vecteurs singuliers fournissent l'information principale pour décrire les documents. Puis, la croissance des performances diminuent pour un nombre de concepts de 1000 à 2000, indiquant ainsi la

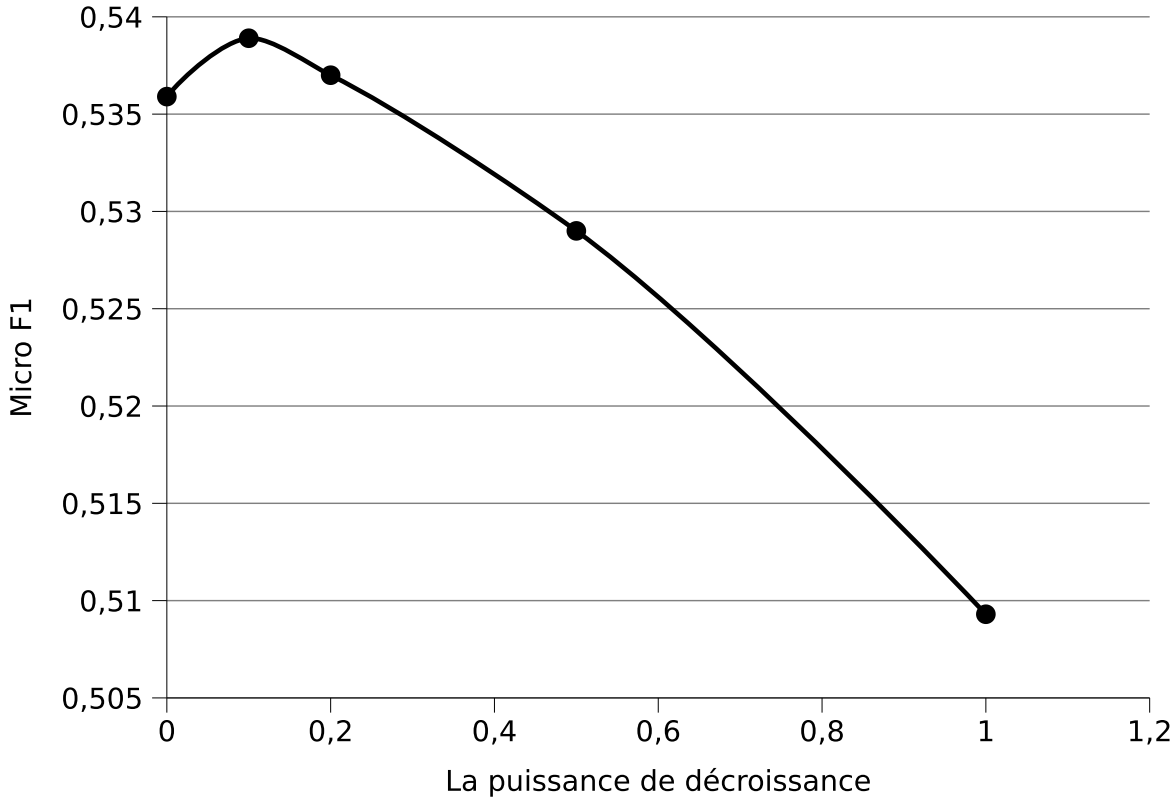


Fig. 5.6 – La variation de la micro-F1 en fonction du pouvoir de décroissant α. Les résultats sont obtenus en utilisant 10% des données d'apprentissage et 10% des données de test.

présence d'une faible quantité d'information.
Les performances du noyau CVSM latent sont meilleures de près de 2% par rapport au noyau BOW SVD. De plus, les différences sont plus prononcées pour un nombre de dimensions faibles. Ceci signifie que le noyau CVSM latent est capable de capturer et d'exprimer l'information principale dans un espace de faible dimension, c'est à dire que l'information principale est résumée par un faible nombre de vecteurs singuliers.
Pour le reste des expériences, nous avons fixé le nombre de concepts latents à 3000.

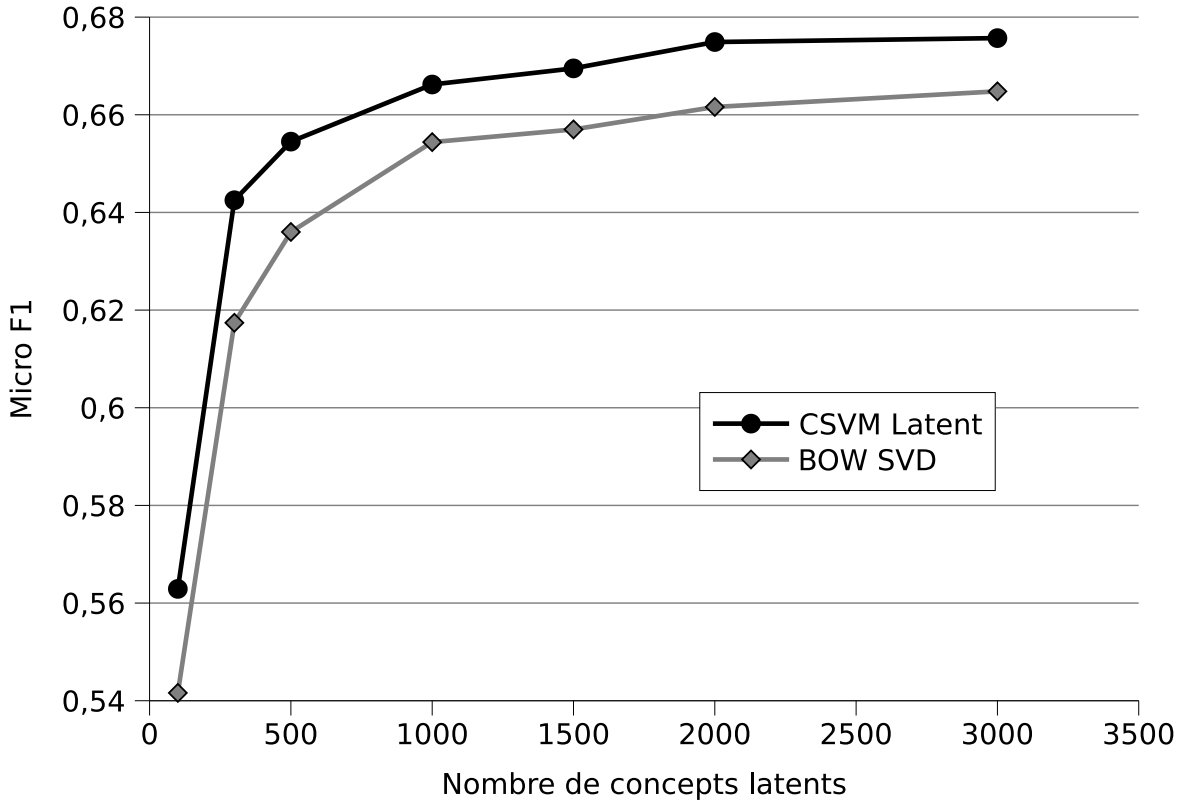


Fig. 5.7 – La variation de la micro-F1 en fonction du nombre de concepts latents.

5.3.3.5 **Évaluation des noyaux**

Les performances des noyaux sur le corpus Ohsumed sont reportées dans les tableaux 5.2 et 5.3. Les expériences ont été réalisées sur l'ensemble des données sans sélection de termes. Les noyaux dans le CVSM obtiennent de meilleurs résultats, jusqu'à 2% de plus, que les noyaux dans le VSM. Comme attendu, les noyaux LSA ont de meilleures performances que les noyaux linéaires. Ceci est dû au fait que les concepts abstraits de haut niveau de l'espace sémantique ré-

sument l'information principale et réduisent le bruit. En outre, le noyau CVSM linéaire est plus performant que le noyau BOW SVD. Nous en déduisons que les concepts basés sur l'ontologie (les concepts linguistiques) sont plus adaptés pour exprimer le sens des documents médicaux que les concepts abstraits (les concepts statistiques) obtenus par la décomposition linéaire de la LSA.

Kernel	**F1**	**Précision**	**Rappel**
BOW Linéaire	65.83%	75.65%	58.26%
BOW SVD	66.48%	76.21%	58.95%
CVSM Linéaire	67.08%	77.77%	58.98%
CVSM Latent	67.57%	76.01%	60.82%

Tab. 5.2 – Les scores micro-moyennés pour le corpus Ohsumed.

Kernel	**F1**	**Précision**	**Rappel**
BOW Linéaire	60.32%	76.01%	51.7%
BOW SVD	60.62%	74.87%	52.69%
CVSM Linéaire	61.31%	78.07%	52.39%
CVSM Latent	62.71%	75.48%	55.29%

Tab. 5.3 – Les scores macro-moyennés pour le corpus Ohsumed.

5.3.3.6 Réduction du nombre d'attributs

Dans cette expérience, nous avons testé l'influence de la réduction d'attributs (réduction de dimension) sur les performances des noyaux. Nous avons utilisé le gain d'information [YP97, Seb02], décrit à la section 3.2.2.2, pour la sélection des termes dans l'espace de description. Le gain d'information fournit une indication sur le degré de dépendance entre une catégorie et un terme. En outre, nous avons utilisé la fonction max pour fonction globale sur toutes les catégories. Cette fonction donne le gain d'information maximum d'un terme en fonction des catégories (voir section 3.2.2.2 pour plus de détail.).

Pour nos expériences, nous avons utilisé différents pourcentages d'attributs en ne conservant que les attributs ayant les valeurs de gains d'informations les plus élevées. Les performances des classifieurs sont indiquées dans la figure 5.8. Les classifieurs dans la VSM, à savoir ceux utilisant les noyaux linéaires et BOW SVD, ont besoin de quasiment tous les attributs pour être efficace. Ceci est quelque peu étonnant étant donné que généralement le gain d'information réussit à réduire, sans perte, le nombre d'attributs. Ce résultat peut être expliqué par le fait que le corpus Ohsumed contient un nombre important de termes spécialisés avec des fréquences d'occurrences faibles. Les classifieurs dans la CVSM atteignent leurs meilleurs performances avec seulement 30% des attributs ayant les gains d'informations les plus élevés. Le

reste des attributs introduise du bruit et diminue les performances. Un autre point intéressant est que les noyaux basés sur la LSA sont plus performants que les noyaux linéaires. En effet, l'utilisation des concepts latents améliorent les résultats. Toutefois, le noyau linéaire CVSM est plus performant que le BOW SVD.

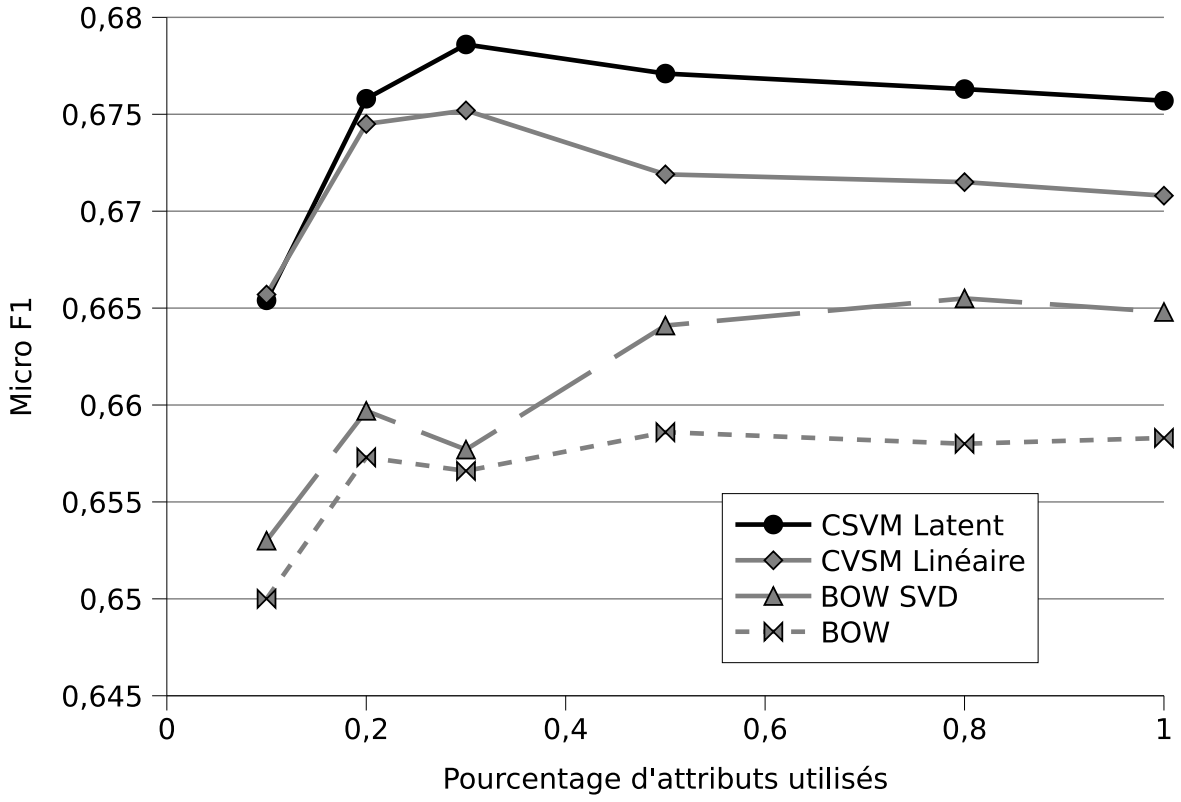


Fig. 5.8 – La variation de la micro-F1 en fonction du pourcentage d'attributs utilisés.

5.3.3.7 **Influence de la quantité de données d'apprentissage**

La figure 5.9 montre l'impact de la quantité de données utilisées pour l'apprentissage sur les performances des noyaux. Comme précédemment, les noyaux LSA sont plus performants que les noyaux linéaires. Néanmoins, il existe deux points intéressants dans ces résultats. Premièrement, le noyau BOW SVD est plus performant que le noyau CVSM linéaire lorsque 30% ou moins des données d'apprentissage sont utilisées. Au delà des 30% le noyau CVSM linéaire est plus performant. Ceci signifie que les concepts statistiques ont un pouvoir discriminant supérieur aux concepts du CVSM pour des petits échantillons de données d'apprentissage. Toutefois, quand la base d'apprentissage est suffisamment importante, les concepts CVSM deviennent plus expressifs. Deuxièmement, tous les noyaux ne réussissent pas à capturer l'information principale avec une faible quantité de données. En effet, les perfomances ne cessent de s'améliorer en

fonction de la quantité de données d'apprentissage. Par conséquent, nous pouvons en déduire que chaque document d'apprentissage fournit une nouvelle information qui améliore la performance de catégorisation. Ceci est principalement dû au fait que ce corpus contient des termes spécifiques avec une faible fréquence d'occurrence.

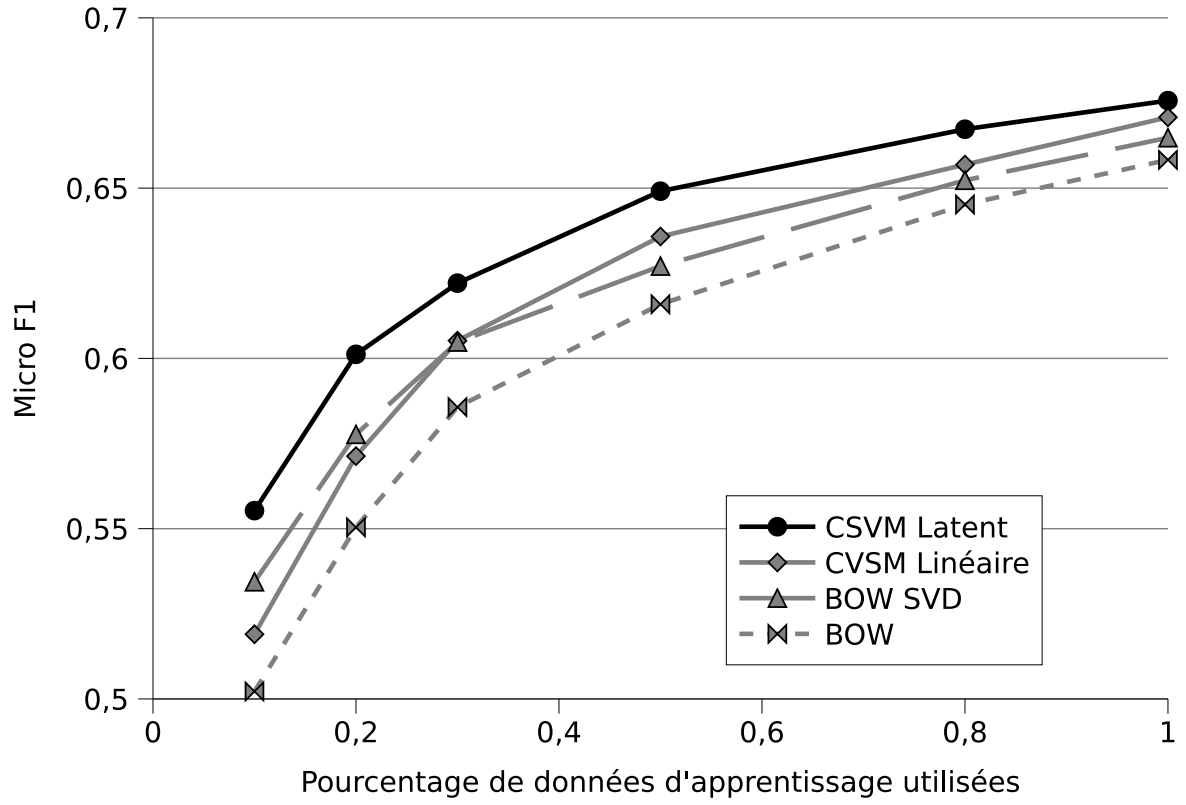


Fig. 5.9 – La variation de la micro-F1 en fonction du pourcentage de documents d'apprentissage utilisés.

5.3.4 **Discussion**

Dans cette seconde partie du chapitre, nous avons présenté un modèle d'espace vectoriel de concepts pour la représentation de documents. Ce modèle utilise des connaissances a priori pour capturer le sens des documents textuels. Nous avons montré une façon simple d'utiliser efficacement les ontologies pour les intégrer au modèle d'espace vectoriel, VSM, standard. Nous avons, aussi, adaptés le noyau linéaire et le noyau LSA pour le CVSM. Nous avons illustré l'utilisation du CVSM par une application au domaine biomédical. Le *Metathesaurus* de l'UMLS a été utilisé comme source a priori de connaissance pour définir le CVSM.

Les performances des noyaux CVSM ont été, expérimentalement, évaluées sur une tâche de catégorisation de documents biomédicaux. Les noyaux ont été comparés au noyau sac-de-mots (BOW) et au noyau LSA. Les résultats ont montré que les noyaux CVSM améliorent les performances de catégorisation de près de 2%. De plus, les expériences ont montré que l'utilisation des concepts latents, extraits à partir des concepts linguistiques par une SVD, permettent d'améliorer les résultats.

Pour un travail futur, nous pensons améliorer la représentation CVSM en intégrant une pondération d'attributs plus adaptée que la pondération IDF. Il a été montré dans [DS03] que la pondération supervisée des termes pouvait améliorer la catégorisation de documents. En effet, des recherches récentes, en matière de pondération des termes, ont donné des résultats prometteurs [SM05, LTL06, LTSL07].

5.4 Un noyau sémantique intégrant les concepts latents locaux

Les algorithmes d'apprentissage supervisé sont des méthodes capable d'extraire des relations à partir d'exemples et d'en déduire des informations en appliquant ces relations à tout ensemble de données. L'objectif de ces méthodes est d'une part de réduire l'intervention humaine au strict minimum, en étant le plus autonome possible, et d'autre part de traiter des volumes importants de données en minimisant le coût d'application.

L'application de l'apprentissage aux données textuelles nécessite un traitement particulier. En effet, les séquences de mots présentes dans ces données véhiculent des informations sémantiques qui ne sont pas accessibles au premier abord. La représentation de ces données dans un espace sémantique passe par la définition des concepts formant cet espace. L'approche usuelle consiste à utiliser une source de connaissance externe telle qu'un thésaurus. Cependant, cette approche implique une dépendance et une spécialisation de la méthode d'apprentissage. En effet, les thésaurus sont souvent spécifiques à certains domaines. L'approche alternative consiste à extraire de manière statistique les concepts. La méthode d'extraction faisant référence en la matière est l'analyse sémantique latente (LSA). Toutefois, la LSA extrait des concepts qui sont bien souvent généraux. Ainsi, lorsque des concepts spécifiques sont nécessaires pour traiter des documents, les méthodes basées sur la LSA voient leurs performances décroître.

Dans cette section, nous présentons une méthode basée sur la LSA permettant d'extraire des concepts statistiques. Pour palier à la faiblesse de la LSA en matière de granularité des concepts, nous proposons dans cette méthode d'utiliser des concepts spécifiques à certaines catégories. Ces concepts sont extraits en appliquant la LSA localement aux classes de documents. Les concepts locaux sont ensuite utilisés pour définir un espace sémantique global. Nous présentons un noyau sémantique enrichi par une pondération pour cet espace sémantique global. Nous montrons que l'utilisation de ce noyau permet de traiter efficacement les données textuelles dans un espace de concepts de faible dimension. Pour cela, le noyau est évalué expérimentalement pour une tâche de catégorisation de document.

5.4.1 Un espace sémantique de concepts locaux

Dans cette section, nous présentons un espace vectoriel sémantique défini par des concepts latents. L'analyse sémantique latente (LSA) permet, par une décomposition SVD, d'extraire des relations de co-occurrences entre les termes. Ces relations sont appelées "concepts latents". Bien que l'espace sémantique obtenu par une LSA globale permette d'améliorer les performances des systèmes, les concepts sont généraux. En effet, ils sont obtenus par une décomposition sur l'ensemble du corpus sans même tenir compte de l'information de la classe. La LSA locale quant à elle permet d'obtenir des concepts plus spécifique en se focalisant davantage sur des zones caractéristiques de chaque classe. Les concepts locaux à chaque classe permettent ainsi de représenter des documents d'une manière plus fine.

Nous proposons de définir un espace sémantique global à partir de concepts locaux. Les concepts locaux sont extraits par une LSA locale à une classe. Pour chaque classe, une matrice de termes par document est constituée à partir des documents de la classe et d'un échantillon de documents non-étiquetés. Dans le cadre de notre travail, l'échantillon non-étiqueté est constitué des documents de test. Les concepts locaux à une classe sont obtenus en appliquant la LSA à cette matrice locale. Ces concepts définissent alors un espace sémantique local à la classe. L'espace sémantique global est obtenu en imposant l'orthogonalité deux à deux des espaces locaux. Chaque concept local défini, ainsi, une dimension de l'espace global. La figure 5.10 illustre la définition d'un vecteur de l'espace sémantique global obtenu à partir des espaces locaux.

La représentation d'un document d dans l'espace sémantique global nécessite le calcul des vecteurs locaux. Un vecteur local, pour une classe c, est obtenu en projetant le document dans l'espace sémantique local avec la matrice de passage de termes par concept. Le vecteur local est ensuite plongé dans l'espace global en fixant toutes les composantes associées aux concepts externes à l'espace local de c, à zéro. Le vecteur global est obtenu en effectuant la somme des vecteurs locaux étendus.

$$\mathbf{\Phi}(d) = \sum_i g_{c_i}\left(\frac{\phi_{vsm}(d)I^{IDF}}{||\phi_{vsm}(d)I^{IDF}||}U_k^{(c_i)}\sqrt{\Sigma_k^{(c_i)}}\right) \tag{5.32}$$

avec I^{IDF} une matrice carré diagonale contenant les poids IDF pour chaque terme t_i calculé à partir d'un corpus $\mathcal{D}$ tel que :

$$I_{i,i}^{IDF} = \log\left(\frac{|\mathcal{D}|}{|\{d'|d' \in \mathcal{D}, w_i \in d'\}|}\right) \tag{5.33}$$

$\phi_{vsm}(d)$ la représentation de d dans le modèle de l'espace vectoriel (VSM), $U_k^{(c_i)}$ la matrice de passage du VSM vers l'espace sémantique local de la classe c_i défini par les k vecteurs propres les plus importants de la matrice de termes par document normalisé de c_i, $\Sigma_k^{(c_i)}$ les k valeurs singulières les plus importantes pour c_i et g_{c_i} la fonction permettant de plonger un vecteur de l'espace local de c_i à l'espace global.

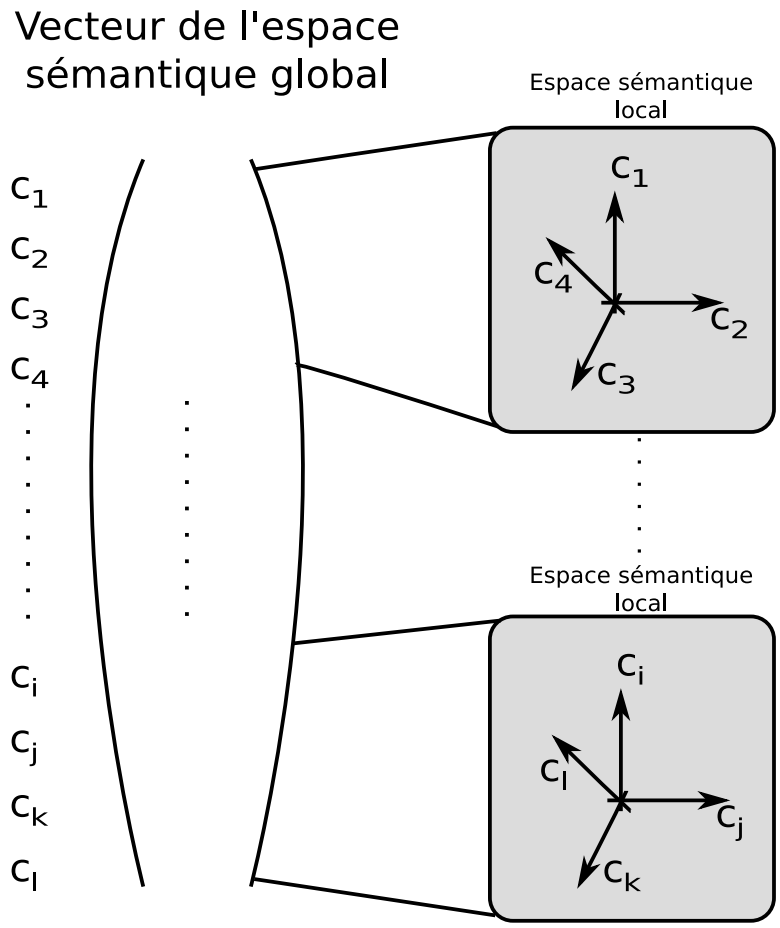


Fig. 5.10 – Espace sémantique global obtenu par des espaces locaux.

5.4.2 **Le noyau sémantique enrichi**

Le produit scalaire de deux documents d_1 et d_2 dans l'espace sémantique global est donné par :

$$k_g(d_1, d_2) = \langle \mathbf{\Phi}(d_1), \mathbf{\Phi}(d_2) \rangle \tag{5.34}$$

L'espace global étant composé de sous-espaces orthogonaux, nous obtenons :

$$k_g(d_1, d_2) = \sum_i \frac{\phi_{vsm}(d_1) I^{IDF}}{||\phi_{vsm}(d_1) I^{IDF}||} U_k^{(c_i)} \Sigma_k^{(c_i)} (U_k^{(c_i)})^T (\frac{\phi_{vsm}(d_2) I^{IDF}}{||\phi_{vsm}(d_2) I^{IDF}||})^T \tag{5.35}$$

Le produit scalaire dans l'espace global est la somme des produits scalaires dans les espaces

sémantiques locaux :

$$k_g(d_1, d_2) = \sum_i k_i(d_1, d_2) \tag{5.36}$$

avec

$$\begin{aligned} k_i(d_1, d_2) &= \langle \phi_{c_i}(d_1), \phi_{c_i}(d_2) \rangle \\ &= \frac{\phi_{vsm}(d_1) I^{IDF}}{||\phi_{vsm}(d_1) I^{IDF}||} U_k^{(c_i)} \Sigma_k^{(c_i)} (U_k^{(c_i)})^T (\frac{\phi_{vsm}(d_2) I^{IDF}}{||\phi_{vsm}(d_2) I^{IDF}||})^T \end{aligned} \tag{5.37}$$

La figure 5.11 illustre le calcul du produit scalaire à partir des espaces sémantiques locaux.

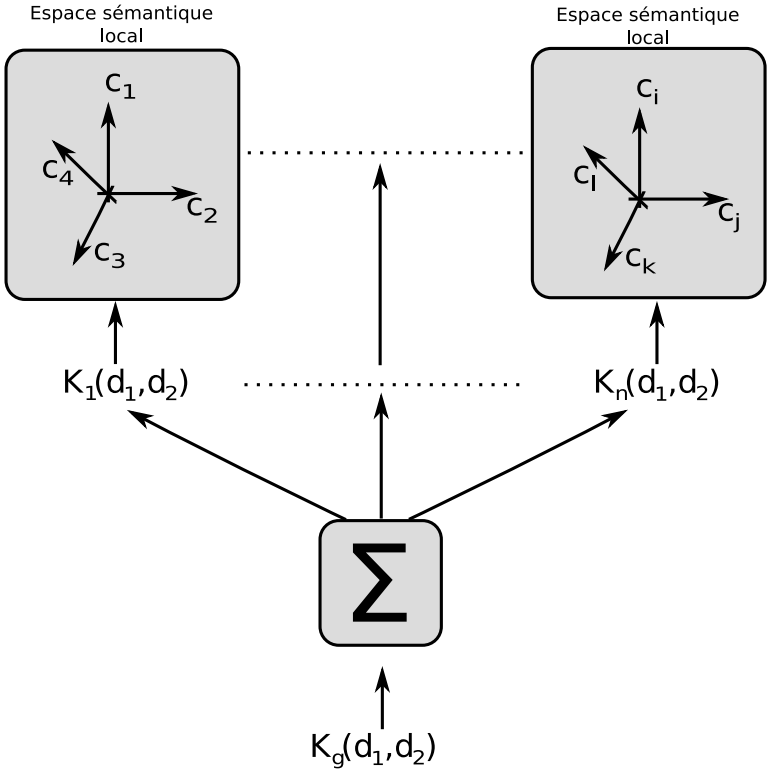


Fig. 5.11 – Noyau sémantique enrichi à partir des noyaux LSA locaux

Certains espaces sémantiques locaux peuvent être plus informatifs que d'autres selon la nature des documents. Ainsi, nous définirons un noyau sémantique enrichi en introduisant une pondération. Cette pondération permettra alors en fonction des documents d'accorder plus d'importance à certains espaces. L'expression du noyau enrichi est alors :

$$k(d_1, d_2) = \sum_i f_i(d_1) f_i(d_2) k_i(d_1, d_2) \tag{5.38}$$

avec $f_i(d)$ une fonction de pondération positive pour une classe c_i et un document d. Nous utiliserons de préférence un noyau normalisé tel que :

$$k(d_1, d_2) = \frac{1}{\sqrt{\sum_{i,j} f_i(d_1)^2 f_j(d_2)^2}} \sum_i f_i(d_1) f_i(d_2) k_i(d_1, d_2) \tag{5.39}$$

Dans le cadre de notre travail, nous proposons de nous concentrer sur les espaces dans lesquels la probabilité d'appartenance des documents est faible. Il s'agit essentiellement de vérifier que les documents se projettent dans les même zones. En effet, si deux documents sont similaires, ils se projetteront dans la même région quel que soit l'espace. Ainsi, nous définirons la fonction de pondération suivante pour la classe c_i :

$$f_i(d) = -\frac{\log(P(d, c_i))}{\sum_j \log(P(d, c_j))} \tag{5.40}$$

La probabilité $P(d, c_i)$ pourra être estimée empiriquement sur l'ensemble d'apprentissage. La section 3.2.5.2 décrit l'estimation multinomiale de $P(d, c_i)$ que nous avons utilisé pour nos travaux.

Il est à noter que pour accorder plus d'importance aux espaces les plus probables, il suffit de prendre une pondération telle que $f_i(d) = P(d, c_i)$.

Theorème 5.4.1. *Le noyau sémantique enrichi est un noyau valide.*

Démonstration. Le noyau sémantique enrichi est un produit scalaire dans l'espace sémantique global. En effet, les documents peuvent être projetés dans cet espace. Il suffit alors de pondérer, avec f_i (fonction positive), chaque concept (dimension) de l'espace global. Le noyau sémantique n'est alors que le produit scalaire classique des vecteurs pondérés de l'espace global. □

5.4.3 Évaluation expérimentale

Dans cette section, nous présentons les expériences que nous avons menées pour évaluer les performances du noyau sémantique enrichi.

5.4.3.1 La préparation des expériences

Les documents des corpus ont été pré-traités pour une représentation dans le modèle d'espace vectoriel (VSM). En effet, le VSM est la représentation initiale pour l'extraction des concepts et la projection dans les espaces sémantiques. Le pré-traitement des documents est composé des phases suivantes :

1. Nettoyage du document par élimination des symboles numériques et des caractères non latin,

2. Élimination des mots vides,
3. Lemmatisation des mots par *Stemming*.

Les problèmes de multi-catégories ont été gérés en utilisant la stratégie un-contre tous. En outre, les problèmes de multi-étiquetage qui consistent à attribuer à un document une ou plusieurs étiquettes ont été pris en charge en utilisant la stratégie *SCut* (voir section 3.2.3.3) avec tous les seuils des catégories fixés à zéros.

Le noyau sémantique enrichi est comparé au noyau linéaire "sac-de-mots" (noyau BOW) et au noyau de domaine décrit dans la section 5.2.5. Le noyau de domaine [GS05] est une méthode utilisant la LSA globale. Il se différencie de la LSA classique par une pondération qui améliore les performances des classifieurs par rapport au noyau LSA [CSTL02].

Par la suite, nous dénoterons le noyau sémantique enrichi par NSE, le noyau de domaine par ND et le noyau linéaire "sac-de-mots" par BOW.

5.4.3.2 Expériences sur le corpus Reuters 21578

Le corpus Reuters 21578 est une collection de 21578 articles de presse. les documents ont été étiquetés manuellement selon le sujet. Un ensemble de 135 catégories (sujets) a été défini, chaque document appartient à aucun ou plusieurs de ces catégories. Ce corpus a été préparé essentiellement pour des tâches de catégorisation. Plusieurs sous-ensembles de ce corpus ont été utilisés pour l'évaluation des systèmes de catégorisation. Parmi ces sous-ensembles, nous avons choisi le sous-ensemble "ModApte" qui est l'ensemble le plus utilisé. Cet ensemble est composé de 9603 documents d'apprentissage et de 3299 documents de test. Pour nos évaluations, nous nous sommes limités aux 10 catégories les plus importantes (contenant le plus de documents). Après pré-traitement pour ces 10 catégories, nous avons obtenu un ensemble de 6490 documents d'apprentissage et de 2545 documents de test. La figure 5.12 illustre un exemple de document issu du corpus Reuters 21578.

Le problème de catégorisation consiste à affecter une ou plusieurs des 10 catégories à chaque document.

Évaluation du nombre de concepts latents

Dans cette section, nous évaluons l'effet du nombre de concepts latents sur les performances du noyau.
La figure 5.13 montre la variation des performances du noyau enrichi (NSE) et du noyau de domaine (ND) en fonction du nombre de concepts latents extraits à partir de la LSA. Les performances du noyau enrichi sont nettement supérieures au noyau de domaine pour un nombre de concepts latents inférieurs à 250. Puis, les performances des deux noyaux convergent pour des dimensions élevées. En effet, certains concepts spécifiques sont associés à des valeurs propres faibles pour la LSA globale. Ainsi, en augmentant le nombre de concepts latents, le noyau de domaine tient compte des concepts spécifiques et converge vers les mêmes performances que le

DEBT DOWGRADED BY MOODY'S

Moody's Investors Service Inc said it lowered the debt and preferred stock ratings of USX Corp and its units. About seven billion dlrs of securities is affected.

Moody's said Marathon Oil Co's recent establishment of up to one billion dlrs in production payment facilities on its prolific Yates Field has significant negative implications for USX's unsecured creditors.

The company appears to have positioned its steel segment for a return to profit by late 1987, Moody's added.

Ratings lowered include those on USX's senior debt to BA-1 from BAA-3.

Reuter &#3;

Fig. 5.12 – Un exemple de document issu du corpus Reuters 21578.

NSE.

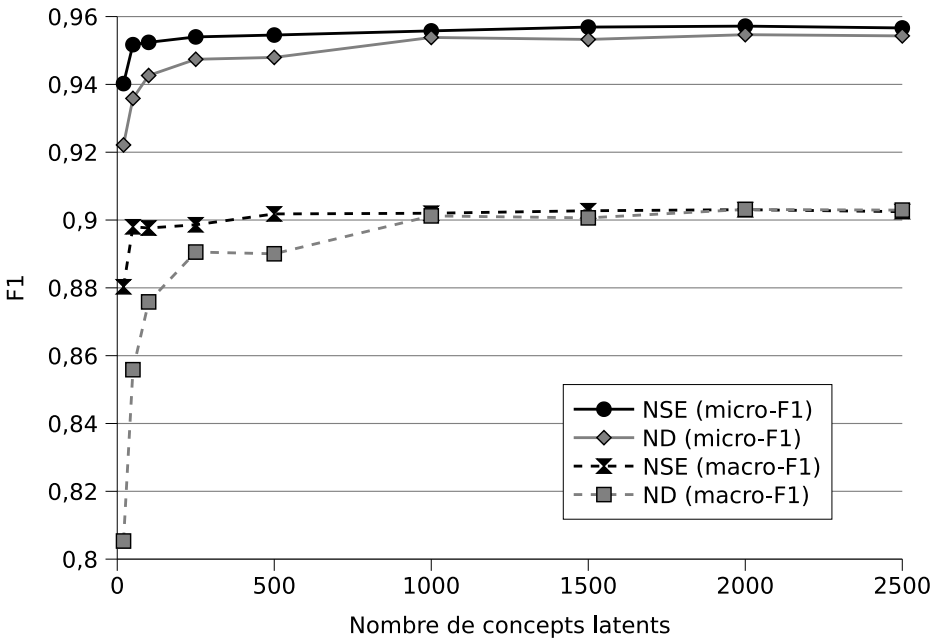


Fig. 5.13 – La variation de la F1 en fonction du nombre de concepts latents sur le corpus Reuters-21578.

Le tableau 5.4 indique les scores de performance des deux noyaux en fonction du nombre de concepts latents. Les différences sont surtout remarquables pour les faibles dimensions.

Dim.	Noyau	Micro			Macro		
		F1	Précision	Rappel	F1	Précision	Rappel
20	NSE	94.43%	94.02%	94.83%	88.03%	89.32%	87.44%
	ND	92.21%	92.01%	92.43%	81.03%	83.55%	80.63%
50	NSE	95.17%	95.22%	95.12%	89.80%	92.72%	88.01%
	ND	93.59%	92.68%	94.51%	85.58%	84.71%	85.58%
100	NSE	95.24%	95.72%	94.76%	89.76%	93.54%	87.08%
	ND	94.26%	93.63%	94.90%	87.58%	87.53%	87.99%

Tab. 5.4 – Les performances des noyaux LSA, en fonction du nombre de concepts latents, pour le corpus Reuters-21578.

Le tableau 5.5 montre les performances des noyaux LSA et du noyau sac-de-mots. La dimension des noyaux LSA, à savoir pour le noyau sémantique enrichi (NSE) et le noyau de domaine (ND) a été fixée à 500. Les noyaux LSA sont plus performants de 2 à 3% que le noyau linéaire sac-de-mots.

Noyau	Micro			Macro		
	F1	Précision	Rappel	F1	Précision	Rappel
NSE	95.46%	96.34%	94.58%	89.54%	93.79%	86.40%
ND	94.79%	94.83%	94.76%	89.00%	90.78%	87.70%
BOW	92.15%	93.08%	91.25%	87.29%	91.37%	83.56%

Tab. 5.5 – Les performances des noyaux pour le corpus Reuters-21578. Le nombre de dimensions a été fixé à 500 pour les LSA.

Influence de la quantité de données d'apprentissage

Dans cette section, nous nous intéressons à l'évolution des performances du noyau sémantique enrichi en fonction du nombre de documents d'apprentissage.
La figure 5.14 illustre l'évolution des performances en fonction de la taille du corpus d'apprentissage. Nous pouvons remarquer que même avec 10% de la base d'apprentissage, le noyau sémantique enrichi obtient de meilleurs performances que le noyau sac-de-mots.

5.4.3.3 Expériences sur le corpus 20NewsGroups

Le corpus 20NewGroups est une collection de 20000 e-mails provenant du groupe de discussion Usenet. Le corpus est segmenté en 20 catégories composées chacune de 1000 documents. Chaque catégorie représente un sujet de discussion. La figure 5.15 illustre un exemple de document issu du corpus 20 NewsGroups.
Pour nos expériences, seul le corps, le titre et l'adresse de provenance des e-mails ont été conser-

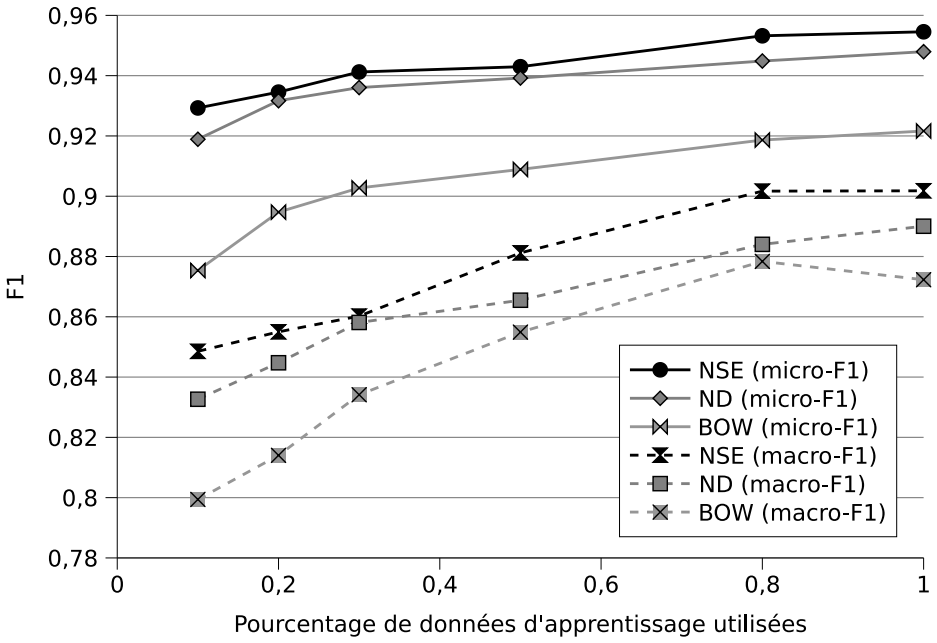


Fig. 5.14 – La variation de la F1 en fonction du pourcentage de documents d'apprentissage utilisés pour le corpus Reuters-21578.

vés. Les e-mails ayant été posté dans plus d'une catégorie ont été affectés aléatoirement à une seule catégorie. Les pré-traitements ont mené à un corpus composé de 18738 documents non vides. L'ensemble d'apprentissage a été composé aléatoirement (comme indiqué dans [GS05]) de 12492 documents (soit 2/3) et l'ensemble de test a été composé de 6246 documents.
Le problème de catégorisation est, ici, un problème de mono-étiquetage. En effet, chaque document doit être affecté à une unique catégorie.

Évaluation du nombre de concepts latents

Le tableau 5.6 présente l'évolution des performances des noyaux LSA en fonction du nombre de concepts. De même que précédemment, les différences entre les deux noyaux sont accentuées pour des faibles dimensions.

Baseball Stats

Hello, my friends and I are running the Homewood Fantasy Baseball League (pure fantasy baseball teams). Unfortunely, we are running the league using Earl Weaver Baseball II with the Comm. Disk II and we need the stats for the 1992 season. (Preferably the 1992 Major League Stat Disk) We have the '92 total stats but EWB2 needs the split stats otherwise we have 200 inning games because the Comm. Disk turns total stats into vs. L's stats unless you know both right and left -handed stats. So, if anyone has the EWB2 '92 Stat Disk please e-mail me!

|Admiral Steve C. Liu Internet Address: admiral@jhunix.hcf.jhu.edu|
| This sig has been brought to you by... Frungy! The Sport of Kings! |

Fig. 5.15 – Un exemple de document issu du corpus 20NewsGroups.

		Micro			Macro		
Dim.	**Noyau**	**F1**	**Précision**	**Rappel**	**F1**	**Précision**	**Rappel**
20	NSE	81.98%	81.08%	82.90%	81.44%	80.80%	82.34%
	ND	59.81%	59.49%	60.13%	55.94%	55.92%	58.70%
50	NSE	84.82%	83.92%	85.73%	84.53%	83.74%	85.40%
	ND	67.61%	66.97%	68.27%	65.24%	64.68%	67.31%
100	NSE	86.50%	85.90%	87.11%	86.29%	85.80%	86.85%
	ND	76.09%	74.85%	77.36%	74.71%	73.62%	76.31%

Tab. 5.6 – Les performances des noyaux LSA, en fonction du nombre de concepts latents, pour le corpus 20 NewsGroups.

En fixant le nombre de dimensions à 250 pour chaque LSA, nous obtenons les performances reportées dans le tableau 5.7. Pour cette expérience, le noyau sac-de-mots est plus performant, de 2%, que le noyau sémantique enrichi. Ceci s'explique essentiellement par la richesse de l'espace VSM qui est constitué pour ce corpus de près de 36000 termes. Des symboles spécifiques, avec une fréquence très faible, permettent ainsi d'améliorer la discrimination. Cependant, ces symboles, dans le cas de la LSA, ne seront associés qu'à des concepts avec des valeurs propres très faibles. Par conséquent, ces concepts ne seront pas utilisés dans des espaces de faible dimension.

	Micro			Macro		
Noyau	**F1**	**Précision**	**Rappel**	**F1**	**Précision**	**Rappel**
NSE	86.53%	85.91%	87.16%	86.37%	85.81%	87.00%
ND	81.87%	80.63%	83.16%	81.28%	80.16%	82.67%
BOW	88.12%	88.03%	88.21%	87.69%	88.07%	87.72%

Tab. 5.7 – Les performances des noyaux pour le corpus 20NewsGroups. Le nombre de dimensions a été fixé à 250 pour les LSA.

5.4.3.4 Expériences sur le corpus Ohsumed

Les résultats des expériences menées sur le corpus Ohsumed, reportés dans le tableau 5.8 confirment que le noyau sémantique enrichi est plus performant lorsque seuls les concepts les plus importants sont utilisés. Ainsi, lorsque la dimension des espaces sémantiques est faible, les différences entre les deux noyaux sont nettement visibles en faveur du noyau sémantique enrichi. Toutefois, lorsque qu'un nombre plus important de concepts est utilisé, les deux méthodes convergent. Ceci s'explique par le fait que les deux méthodes se basent sur la LSA. Par conséquent, la LSA locale fait surgir les concepts importants pour une catégorie. Des concepts proches peuvent aussi être extraits par une LSA globale mais ils seront associés à des valeurs propres faibles. Il sera, ainsi, nécessaire dans la LSA globale d'augmenter la dimension pour tenir compte de ces concepts d'où la convergence des deux méthodes dans des espaces de grande dimension.

		Micro			Macro		
Dim.	**Noyau**	**F1**	**Précision**	**Rappel**	**F1**	**Précision**	**Rappel**
20	NSE	57.78%	72.15%	48.17%	42.61%	66.97%	36.20%
	ND	35.75%	47.60%	28.63%	17.84%	28.76%	16.51%
50	NSE	60.31%	74.61%	50.61%	47.03%	69.81%	40.00%
	ND	48.35%	62.54%	39.41%	31.83%	46.72%	29.50%
100	NSE	61.85%	75.79%	52.23%	52.06%	69.94%	43.94%
	ND	56.71%	70.05%	47.64%	43.55%	60.21%	38.38%

Tab. 5.8 – Les performances des noyaux LSA, en fonction du nombre de concepts latents, pour le corpus Ohsumed.

Le tableau 5.9 montre les résultats obtenus pour les noyaux LSA et le noyau sac-de-mots. Ainsi, pour un espace sémantique de dimension 500, les trois noyaux ont des performances similaires.

	Micro			Macro		
Noyau	**F1**	**Précision**	**Rappel**	**F1**	**Précision**	**Rappel**
NSE	65.40%	76.84%	56.92%	60.26%	72.67%	51.47%
ND	64.37%	72.15%	58.11%	59.58%	69.05%	53.27%
BOW	65.83%	75.66%	58.26%	60.32%	76.01%	51.70%

Tab. 5.9 – Les performances des noyaux pour le corpus Ohsumed. Le nombre de dimension a été fixé à 500 pour les LSA.

5.4.3.5 Expériences sur le corpus ICD-9-CM

Les expériences sur le corpus ICD-9-CM, dont les résultats sont reportés dans le tableau 5.10, montrent que le noyau sémantique est légèrement plus performant que le noyau de domaine dans des espaces de faible dimension. Cependant, les performances macro-moyennées du noyau sémantique sont plus faible que les performances du noyau de domaine. Nous rappelons que la macro-moyenne donne une importance égale à chaque catégorie sans prendre en compte la taille de chaque catégorie. Ainsi, le noyau sémantique enrichi commet de multiples erreurs pour le classement des documents appartenant à des catégories faiblement représentées. En effet, le corpus ICD-9-CM est caractérisé par la présence de nombreuses catégories contenant très peu de documents. En outre, la LSA a été appliquée, pour chaque catégorie, sur une matrice contenant les documents de la catégorie et les documents de test. Par conséquent, pour les petites catégories, les documents de ces catégories sont "noyés" parmi les documents de test et la LSA ne réussit pas extraire les concepts pertinents.

		Micro			Macro		
Dim.	**Noyau**	**F1**	**Précision**	**Rappel**	**F1**	**Précision**	**Rappel**
20	NSE	62.13%	67.11%	57.85%	25.78%	29.21%	23.08%
	ND	59.36%	64.10%	55.27%	27.06%	31.28%	27.17%
50	NSE	72.16%	75.37%	69.22%	32.42%	32.80%	32.06%
	ND	69.42%	71.20%	67.72%	36.38%	41.97%	35.97%
100	NSE	77.22%	79.28%	75.27%	40.66%	41.87%	39.51%
	ND	75.14%	77.56%	72.86%	38.81%	42.93%	37.92%

Tab. 5.10 – Les performances des noyaux LSA, en fonction du nombre de concepts latents, pour le corpus ICD-9-CM.

Le tableau 5.11 indique les performances des noyaux LSA pour une dimension 500 par rapport au noyau sac-de-mots. Le noyau sémantique enrichi est légèrement meilleur que les autres noyaux en terme de micro-moyenne. Cependant, le noyau de domaine est plus performant que les autres en terme d'indicateurs macro-moyennés.

	Micro			Macro		
Noyau	**F1**	**Précision**	**Rappel**	**F1**	**Précision**	**Rappel**
NSE	81.33%	82.89%	79.83%	47.91%	53.07%	46.93%
ND	79.79%	82.03%	77.67%	49.94%	55.27%	48.94%
BOW	80.22%	82.47%	78.09%	47.54%	54.24%	45.98%

Tab. 5.11 – Les performances des noyaux pour le corpus ICD-9CM. Le nombre de dimension a été fixé à 500 pour les LSA.

5.4.4 Discussion

Dans cette dernière partie, nous avons présenté un espace sémantique global défini à partir de concepts locaux. Les concepts locaux sont extraits en appliquant une LSA localement aux documents d'une catégorie. Nous avons, ensuite, proposé un noyau sémantique enrichi pour l'espace global. Ce noyau permet de se focaliser sur certains sous-espaces sémantiques selon la nature des documents.

Les performances du noyau sémantique enrichi ont été évaluées expérimentalement sur une tâche de catégorisation de document. Les corpus bien connus de la littérature, tels que le Reuters-21578, ont été utilisés pour faciliter les comparaisons entre les méthodes de la littérature. Ainsi, les résultats montrent que le noyau sémantique est nettement plus performant, pour de faible dimension, qu'une méthode basée sur la LSA globale. Cependant, lorsque le nombre de concepts (la dimension de l'espace sémantique) est suffisamment important les deux méthodes convergent.

Bien que les performances du noyau sémantique enrichi soient intéressantes, la méthode souffrent d'un problème important qui est le temps de calcul. En effet, une décomposition SVD nécessite un temps non négligeable et, ainsi, effectuer des décompositions locaux peut s'avérer extrêmement coûteux en mémoire et en temps de calcul. Dans un travail futur, nous nous intéresserons à réduire le nombre de LSA locaux sans perte significative des performances. Pour cela, nous étudierons des critères statistiques permettant de fusionner certaines régions. En outre, une étude devra être menée sur la pondération utilisée dans le noyau enrichi afin de savoir si une autre pondération peut être mieux adaptée.

5.5 Conclusion

Dans ce chapitre, nous avons présenté des noyaux basés sur l'information latente extraite par l'analyse sémantique latente.

La première catégorie de noyaux est définie dans un espace sémantique construit à partir de concepts linguistiques provenant d'un thésaurus. Les expériences sur un corpus médical ont montré que l'utilisation de concepts latents extraits de l'espace sémantique linguistique permet d'améliorer, en comparaison au modèle d'espace vectoriel, les performances de catégorisation de près de 2%.

Le deuxième noyau utilise un espace vectoriel formé de concepts locaux. Les concepts locaux sont extraits en appliquant une LSA locale à certaine région de l'espace. Dans le cadre de notre travail, une région est définie par l'ensemble des documents d'une catégorie. Les concepts locaux de chaque région forment alors un sous-espace orthogonal aux autres sous-espaces. Les expériences montrent que ce noyau est performant lorsque les sous-espaces sont de faibles dimensions.

Notre étude sur les concepts latents montre que les concepts linguistiques restent plus expressifs que les concepts latents. En effet, la deuxième partie de ce chapitre a montré que l'espace de concepts linguistiques permet d'obtenir de meilleures performances que l'espace de concepts latents obtenu à partir du VSM. Toutefois, l'espace sémantique peut-être mieux défini en réduisant les LSA locaux.

CHAPITRE 6

Autres contributions : Extraction de motifs séquentiels

6.1 Introduction

Dans ce chapitre, nous présentons nos travaux de recherches effectués, en collaboration avec Dr. Aomar Osmani, durant la première année et demie de thèse. Ces travaux portent sur l'extraction de motifs séquentiels à partir de grandes bases de données. Ces travaux se situent dans le domaine de l'apprentissage symbolique et concernent la génération de règles d'association de données symboliques. En outre, l'extraction de motifs séquentiels peut aussi être appliqué au domaine textuel [KM03]. En effet, un document textuel peut être perçu comme une séquence de mots ou même, à un niveau plus bas, comme une séquence de lettres.

Nos travaux dans le domaine de l'extraction de motifs séquentiels ont donné lieu à une publication dans un "workshop" international [AO05] et à une publication dans une conférence internationale [AOV06].

Ce chapitre présente notre contribution majeure au domaine de l'extraction de motifs séquentiels. Nous débutons par présenter ce domaine notamment en détaillant les algorithmes principaux qui servent de références. Puis, nous présentons l'algorithme bitSPADE que nous avons proposé pour extraire efficacement les motifs séquentiels. Cet algorithme permet d'obtenir d'excellentes performances en matière de temps de traitement et d'utilisation de mémoire.

6.2 Problématique de l'extraction de motifs séquentiels

Le data-mining est devenu un processus majeur dans la gestion des relations clients (CRM). Il s'agit essentiellement d'un processus chargé d'extraire des connaissances spécifiques sur les

comportements des clients partant d'une série d'observations. L'objectif étant alors de dresser le profile comportemental d'un client. Pour cela, le data-mining utilise les observations relatives aux clients dans des situations spécifiques. Ces observations sont regroupées dans de larges bases de donnée qui serviront à alimenter les outils de data-mining. Ces outils devront alors mettre à jour les relations reliant les différentes variables de ces bases de données. Ces relations traduisent un lien entre les différents comportements relatifs à un type de client.
Les informations récoltés permettent ainsi de déterminer le profile d'un client, selon des caractéristiques définies. Ce profile peut alors être utilisé dans divers processus du CRM tel que l'aide à la décision etc.

Il existe différents type de techniques permettant d'extraire les relations entre variables provenant de larges bases de données. Parmi ces techniques, nous pouvons citer les réseaux de neurones et les méthodes probabilistes.
Dans ce document, nous nous concentrerons sur les règles d'associations relatives aux séquences temporelles qui peuvent être ensuite compilées pour produire un arbre de décision.
Ces règles permettent de déterminer dans un contexte précis défini par un enchaînement temporel d'actions ou événements passés, l'événement le plus probable de se produire.
Pour obtenir les règles d'associations, il est nécessaire d'extraire les motifs séquentiels les plus fréquents apparaissant dans la base de donnée. Ces motifs traduisent un enchaînement d'actions s'étant produit fréquemment et susceptible de se reproduire.

6.2.1 Formulation du problème

Étant donnée une base de donnée $\mathcal{D}$ de séquences temporels (séquence-exemple), le problème de l'extraction de motifs séquentiels consiste à déterminer l'ensemble $\mathcal{S}_{max}$ des séquences maximales. Ces séquences maximales proviennent de $\mathcal{S}_{freq}$ qui est l'ensemble des sous-séquences fréquentes de $\mathcal{D}$ tel que :

$$\mathcal{S}_{max} \subseteq \mathcal{S}_{freq}$$

$$\forall s \in \mathcal{S}_{freq}, \exists y \in \mathcal{D} \mid s \preceq y$$

Définition 6.2.1 (Séquence)**.** *Une séquence est définie comme étant un ensemble d'événements ordonnés dans le temps.*

Définition 6.2.2 (Événement)**.** *Un événement est un ensemble d'éléments partageant la même date d'occurrence. La cardinalité d'événement x notée $card(x)$ est le nombre d'élément le composant.*
Par soucis d'optimisation, l'opérateur d'ordre total $<$ sera défini sur les éléments d'événements.

Définition 6.2.3 (Taille d'une séquence)**.** *La taille d'une séquence s est définie par sa cardinalité à savoir le nombre d'événements composants la séquence s :*

$$taille(s) = card(s) = \sum_{x \in s} 1$$

Définition 6.2.4 (Taille totale d'une séquence)**.** *La taille totale d'une séquence est le nombre d'éléments composant les événements de la séquence s :*

$$taille_T(s) = \sum_{x \in s} card(x)$$

Définition 6.2.5 (Séquence-exemple)**.** *Une séquence est dite « séquence-exemple » si elle appartient à la base de donnée utilisée pour extraire les motifs.*

Définition 6.2.6 (Sous-séquence)**.** *Soient* $s =< s_1, ..., s_n >$ *et* $x =< x_1, ..., x_p >$ *deux séquences, s est dite sous-séquence de x si et seulement si tous les événements de s apparaissent dans le même ordre dans x. On dit qu'un événement* s_i *apparaît dans x si et seulement s'il existe un événement* x_j *de x tel que* s_i *est un sous-ensemble de* x_j*.*
Autrement dit,s est dite sous-séquence de x ($s \preceq x$*) s'il existe un ensemble d'entiers* $\{i_1, ..., i_n\}$ *tel que* $i_1 < ... < i_n$ *et* $\forall j \in \{1, ..., n\}, \exists x_{i_j} \in x \mid s_j \subseteq x_{i_j}$*.*
Une séquence s sera dite sous-séquence d'un ensemble $\mathcal{D}$ *de séquences si et seulement si :* $\exists s' \in \mathcal{D} \mid s \preceq s'$*.*

Définition 6.2.7 (séquence maximale)**.** *Une séquence s est dite maximale dans un ensemble* $\mathcal{S}$ *si* $\nexists x \in \mathcal{S} - \{s\} \mid s \preceq x$

Définition 6.2.8 (Support)**.** *Le support d'une séquence s, dans une base* $\mathcal{D}$*, est le nombre de séquences de* $\mathcal{D}$ *contenant la séquence s divisé par le nombre de séquences totales dans* $\mathcal{D}$ *:*

$$support_{\mathcal{D}}(s) = card(\{x \in \mathcal{D} \mid s \preceq x\})$$

Définition 6.2.9 (Séquence fréquente)**.** *Étant donnée la valeur d'un support seuil, min_support, une séquence s est dite fréquente si son support dans une base* $\mathcal{D}$ *est supérieur ou égale à min_support.*

Définition 6.2.10 (Opérateur d'accession sur les séquences)**.** *Soit une séquence s de taille n, le* $i^{ème}$ *événement de s (*$0 < i \leq n$*) est noté* s_i *ou encore* $s[i]$*.*

Définition 6.2.11 (Opérateur de concaténation $\cdot$ sur les séquences)**.** *L'opérateur* $\cdot$ *utilisé sur les séquences est un opérateur de concaténation étant donnée que les séquences sont des ensembles ordonnées d'événements.*
Soit deux séquences $s =< s_1, ..., s_n >$ *et* $x =< x_1, ...x_p >$ *:*

$$y = s \cdot x =< s_1, ..., s_n, x_1, ..., x_p > \neq x \cdot s$$

Les sous-séquences fréquentes de la base d'apprentissage forment les motifs séquentiels de cette base. Ne conserver que les sous-séquences maximales de ces sous-séquences fréquentes préserve la complétude de l'espace des motifs. En effet, toute sous-séquence d'une séquence maximale est une séquence fréquente (principe d'anti-monotonie).

Définition 6.2.12 (Anti-Monotonie)**.** *Une fonction f est dite anti-monotone si et seulement si :*

$$\forall x_1, x_2 \in \mathcal{D}_f \times \mathcal{D}_f,\ x_1 \leq x_2 \Rightarrow f(x_1) \geq f(x_2)$$

Ainsi, si s_1 et s_2 sont deux séquences alors, on a :

$$s_1 \preceq s_2 \Rightarrow support_{\mathcal{D}}(s_1) \geq support_{\mathcal{D}}(s_2)$$

L'anti-monotonie de la fonction *support* nous garantie que toutes les sous-séquences des séquences fréquentes sont fréquentes.

A partir de l'ensemble des sous-séquences maximales de la base de donnée d'apprentissage $\mathcal{D}$, il est alors possible d'en déduire les règles d'association dans le cadre des séquences en attribuant à chaque règle un degré de confiance.

Définition 6.2.13 (règle d'association)**.** *Une règle d'association est composée d'une prémisse et d'une conclusion. Pour les règles d'association de séquences, la prémisse et la conclusion sont deux séquences telles que si la séquence à eu lieu alors la séquence conclusion aura lieu immédiatement après.*

Définition 6.2.14 (Degré de confiance d'une règle)**.** *On associe à une règle d'association un degré de confiance. Ansi, pour une règle reliant les séquences s_1 et s_2 telle que $R : s_1 \Rightarrow s_2$, on définit le degré de confiance par la formule suivante :*

$$conf_{\mathcal{D}}(s_1 \Rightarrow s_2) = \frac{support_{\mathcal{D}}(s_1 \cdot s_2)}{support_{\mathcal{D}}(s_1)}$$

Étant donné, un seuil de confiance minimum, min_conf, une règle R est dite sûre si $conf_{\mathcal{D}}(R) \geq min\_conf$.

Définition 6.2.15 (prefix)**.** *Soit $s =< s_1, ..., s_n >$ et $k =< k_1, ..., k_p >$ deux séquences :*

$$k \in prefix(s) \Leftrightarrow 0 < p < n \text{ et } \forall i \leq p,\ k_i = s_i$$

Étant donnée $\mathcal{S}_{max}$, l'ensemble des sous-séquences maximales d'une base $\mathcal{D}$, et min_conf le degré de confiance minimal pour qu'une règle soit considérée comme étant « sûre », l'ensemble des règles sûres $\mathcal{R}_f$ est défini par :

$$\forall s \in \mathcal{S}_{max}, \forall l \in prefix(s), R_{l,s-l} : l \Rightarrow s - l \in \mathcal{R}_f \Leftrightarrow conf_{\mathcal{D}}(R_{s,s-l}) \geq min\_conf$$

6.2.2 Famille Apriori

6.2.2.1 Principe

L'algorithme "Apriori", proposé en 1995 par Agrawal et Srikant dans [AS95], est basée sur la génération a priori de motifs potentiels appelés candidats. Un motif étant une séquence apparaissant en tant que sous-séquence dans un nombre de séquence-exemples suffisant défini par un seuil. Ainsi, étant donnée une base d'apprentissage et un seuil, un motif est une séquence fréquente.

L'objectif de cet algorithme est de déterminer l'ensemble des motifs tout en ne conservant que les motifs maximums à savoir les motifs qui ne sont contenus dans aucun autre motif de l'ensemble.
Pour cela, Apriori utilise les principes suivantes :

- La génération, incrémentale, a priori de séquences candidates pouvant être des motifs
- Utilisation du principe d'anti-monotonie associé aux séquences pour l'élagage des candidats
- Le calcul du support pour chaque candidat i.e. le dénombrement des séquences-exemples contenant le motif candidat

Les algorithmes utilisant les principes ci-dessus seront classés dans la famille d'algorithme "Apriori".

Génération incrémentale des candidats

La génération incrémentale débute par déterminer l'ensemble $\mathcal{C}_1$ de toutes les séquences possibles de taille 1 [1]. Puis, lorsque l'ensemble L_1 des motifs de taille 1 a été déterminé, l'ensemble $\mathcal{C}_2$ de toutes les séquences possibles de taille 2 est généré. La génération s'arrête lorsque l'ensemble $\mathcal{C}_n$ est vide ce qui implique qu'il n'existe pas de motifs de taille n et par conséquent la taille maximale des motifs sera $n-1$.

Utilisation du principe d'anti-monotonie

Le principe d'anti-monotonie de la fonction *support* garantie que si une séquence est fréquente alors toutes ses sous-séquences sont fréquentes. Autrement dit, si une séquence n'est pas fréquente alors toutes ses sur-séquences ne sont pas fréquentes.
Cette propriété, étant donnée l'ensemble L_{n-1} des séquences fréquentes de taille $n-1$ permet de réduire l'espace des candidats $\mathcal{C}_n$ généré précédemment pour obtenir le sous-espace de candidat $C_n \subseteq \mathcal{C}_n$ tel que :

$$C_n = \{x \in \mathcal{C}_n \mid \forall s \prec x \; et \; taille(s) = n-1, \; s \in L_{n-1}\}$$

Calcul du support des candidats

Le calcul du support d'un candidat s'effectue en parcourant la base d'exemple. Il s'agit alors de "compter" le nombre de séquences exemples qui sont des sur-séquences du candidat. Étant donnée un seuil, un candidat devient un motif si et seulement si son support est supérieur ou égal au seuil.

[1] Par "taille d'une séquence" on désigne la taille totale de la séquence mais dans certains cas, comme pour les algorithmes "AprioriAll" et "AprioriSome", la taille désigne la cardinalité.

6.2.2.2 Famille d'algorithme AprioriAll

La particularité de cet algorithme est de générer, de manière itérative, les candidats pouvant être des motifs, à savoir, des séquences fréquentes. Une fois, les candidats générés, leurs supports est calculés en effectuant une lecture de la base d'exemples ; Les candidats ayant un support inférieur à un seuil sont, alors, éliminés. Une autre itération est, ensuite, effectuée pour générer des candidats de tailles supérieures jusqu'à ce que tous les candidats soient éliminés.
Le résultat de l'algorithme est l'ensemble des motifs maximum.
AprioriAll est donc un algorithme "multi-passe" , à savoir qu'il doit effectuer plusieurs passes (lectures de la base d'exemples), plus précisément une passe par itération pour déterminer les motifs.

Cet algorithme se décompose en cinq grandes phases :

1. Phase de préparation de la base de séquences
2. Phase de détermination des motifs d'évènements
3. Phase de transformation de la base de séquences
4. Phase de détermination des motifs de séquences
5. Phase d'élimination des motifs non maximums

Il est à noter qu'AprioriAll utilise la cardinalité en tant que définition pour la taille d'une séquence. Ainsi, la taille d'une séquence sera le nombre d'évènements composant la séquence.
Cette notion de taille est un élément très important pour cette famille d'algorithme. En effet, Cette définition est un point important pour la génération des candidats.

Phase de préparation de la base de séquences

Les bases de données, à partir desquels les motifs doivent être déterminés, sont souvent inadaptées pour l'extraction de motifs séquentiels. Il est alors nécessaire de préparer les données à l'application.
L'objectif est avant tout d'obtenir une base sous forme de séquence.
Les éléments des évènements composant les séquences doivent être triées dans un ordre croissant afin d'optimiser les traitements.

Phase de détermination des motifs d'évènements

Cette étape permet de déterminer les motifs évènements à savoir les sous-ensembles des évènements composants les séquences exemples tels que le support de chacun de ces sous-ensembles est supérieur ou égal à un seuil.
Nous définirons le support d'un évènement e comme étant le rapport entre le nombre de séquences s supportant e ($\exists x \in s \mid e \subseteq x$) et le nombre de séquences appartenant à la base d'exemples.

Le problème peut être reformulé comme étant la détermination des séquences fréquentes de taille 1.

Cette phase ne conservera que les évènements (ou séquences de taille 1) maximums étant donné que tous sous-ensemble d'un motif est un motif (principe d'anti-monotonie de la fonction support).

A cette étape, le problème est identique à celui rencontré dans la génération des règles d'associations d'éléments. En effet, la partie fondamentale de la génération des règles d'association d'éléments consiste dans la détermination des ensembles d'éléments qui apparaissent fréquemment dans la base d'exemples.
Ainsi, tout algorithme de calcul d'ensemble d'éléments (évènements) peut être utilisé. Cependant, il est à noter que la fonction *support* est relative, ici, aux séquences d'évènements et non aux évènements comme c'est le cas pour la génération de règles d'association "classique". Ainsi, la fonction *support* d'un évènement e devra compter le nombre de séquences dans lesquelles il existe un sur-évènement de e et non le nombre de sur-évènement de e.
Il est, ainsi, possible d'utiliser des algorithmes de recherche de motifs d'évènements tels qu' Apriori ([AS94]) en adaptant la fonction "support".

A la fin de cette étape, nous obtenons l'ensemble L_1 de tous les motifs de séquences de taille 1. Les motifs sont, ici, des séquences composées uniquement d'un seul évènement.
Ainsi, un motif de taille quelconque sera composé uniquement d'évènement appartenant à l'ensemble L_1. Cette propriété sera utilisée pour générer les différentes séquences candidates.

Phase de transformation de la base de séquences

L'ensemble, L_1, des évènements composant les motifs de séquences ayant été déterminé, les éléments des évènements de la base $\mathcal{D}$ n'apparaissant pas dans au moins un évènement de L_1 peuvent être éliminés. De plus, chaque évènement de $\mathcal{D}$ sera remplacé par l'ensemble de tous ses sous-évènements présents dans L_1. La base résultante $\mathcal{D}_\mathcal{T}$ sera, alors, constituée de séquences d'ensemble d'évènements appartenant à L_1.

L'algorithme 6.2.1 illustre la transformation.

De plus, la base $\mathcal{D}_\mathcal{T}$ est encodée de telle façon que chaque ensemble d'un évènement reçoit un identifiant unique. Ainsi, les séquences de $\mathcal{D}_\mathcal{T}$ seront composées d'évènements dont les éléments seront des identifiants uniques se référant aux évènements fréquents de L_1. L'avantage de cet encodage est qu'il permet d'associer à un ensemble d'éléments un numéro unique. Ainsi, pour savoir si deux ensembles sont identiques, il suffira juste de comparer les identifiants.
Cet encodage doit être pris en compte dans l'algorithme 6.2.1. Il suffira alors de remplacer l'instruction "$y \leftarrow y \cup \{x\}$" par une instruction du type "$y \leftarrow y \cup \{code(x)\}$".

Ainsi, à la fin de cette étape une séquence de $\mathcal{D}_\mathcal{T}$ n'est plus un ensemble ordonné d'ensembles d'éléments mais un ensemble ordonné d'ensembles d'identifiants représentant des évènements de L_1.

Algorithme 6.2.1 apriori_transform

Entrée: $\mathcal{D}$, Base de séquences
Entrée: L_1, Ensemble des séquences fréquentes de taille 1
Sortie: $\mathcal{D}_\mathcal{T}$, Base transformée

1: $\mathcal{D}_\mathcal{T} \leftarrow \emptyset$
2: **pour tout** $S \in \mathcal{D}$ **faire**
3: $S' \leftarrow \emptyset$
4: **pour tout** $s \in S$ **faire**
5: $y \leftarrow \emptyset$
6: **pour tout** $x \in L_1 \mid x \preceq s$ **faire**
7: $y \leftarrow y \cup \{x\}$
8: **fin pour**
9: $S' \leftarrow S' \cdot \{y\}$
10: **fin pour**
11: **si** $S' \neq \emptyset$ **alors**
12: $\mathcal{D}_\mathcal{T} \leftarrow \mathcal{D}_\mathcal{T} \cup \{S'\}$
13: **fin si**
14: **fin pour**

Phase de détermination des motifs de séquences

Pour la détermination des motifs de séquences, la base de donnée $\mathcal{D}_\mathcal{T}$ sera utilisée. Ainsi, les identifiants d'évènements seront utilisés à la place des évènements. Par conséquent, L_1 désignera l'ensemble des identifiants représentant les motifs des évènements.

Cette phase utilise le même principe que l'algorithme Apriori ([AS94]) sur les évènements. Ainsi, AprioriAll est un algorithme itératif qui, à l'étape $i > 1$), génère les séquences candidates (pour être des motifs) de taille i à partir des motifs de taille $i-1$ (L_{i-1}). Une fois ces candidates générées, Apriori détermine le support exacte de chacune des candidates en effectuant une passe sur la base $\mathcal{D}_\mathcal{T}$. Les candidates ayant un support inférieur au support seuil sont éliminées, les autres formant, alors, l'ensemble L_i des motifs de séquences de taille i. AprioriAll, réitère le processus en générant les candidates de tailles $i+1$ à partir de L_i et s'arrête lorsque que L_{i+1} est un ensemble vide à la fin de l'étape $i+1$.
L'algorithme 6.2.2 illustre le fonctionnement d'AprioriAll.
Il est à noter qu'un des moyens d'optimiser le calcul du support lors d'une passe sur la base $\mathcal{D}_\mathcal{T}$ est d'utiliser une structure "hash-tree" comme décrit dans [AS94]. Les candidates appartenant à C_k sont stockés dans une "hash-tree" ainsi pour chaque séquence s de $\mathcal{D}_\mathcal{T}$, il suffira de parcourir la "hash-tree" afin de réduire le nombre de séquences candidates pour lesquelles un test de sous-séquence doit être effectué.

La fonction de génération des séquences candidates de taille k à partir de l'ensemble, L_{k-1}, des séquences motifs de taille $k-1$ utilise le même principe que la fonction de génération utilisé

Algorithme 6.2.2 aprioriAll

Entrée: $\mathcal{D}_\mathcal{T}$, Base de séquences
Entrée: L_1, Ensemble des séquences fréquentes de taille 1
Sortie: L_f, Ensemble des motifs de séquences

```
1: k ← 2
2: tant que L_{k-1} ≠ ∅ faire
3:    C_k ← apriori − generate(L_{k-1})
4:    pour tout S ∈ D_T faire
5:       pour tout c ∈ C_k faire
6:          si c ⪯ S alors
7:             c.count + +
8:          fin si
9:       fin pour
10:   fin pour
11:   L_k ← {c ∈ C_k | c.support ≥ minsup}
12:   k ← k + 1
13: fin tant que
14: L_f ← ∪_k L_k
```

dans l'algorithme Apriori sur les ensembles ([AS94]). En effet, une séquence s peut être un motif si et seulement si toutes les sous-séquences de s sont des motifs (principe d'anti-monotonie de la fonction support). Ainsi, lors d'un processus itératif débutant par la génération de motifs de taille 1 ; A l'étape k, la génération d'une séquence s de taille k s'effectue en recherchant dans L_{k-1} deux séquences x et y tel que x et y partagent le même préfixe de taille $k-2$, s s'obtient alors en concaténant le dernier évènement de y à x. De plus, s n'est probable que si toutes les sous séquences de s de taille $k-1$ et possédant le même suffixe, de taille 2, que s appartiennent à L_{k-1}.
L'algorithme 6.2.3 illustre le déroulement de l'étape de génération.

Phase d'élimination des motifs non maximums

A la fin de l'étape précédente, l'ensemble L_f des motifs des séquences de la base $\mathcal{D}$ ont été déterminés. Toutefois, il devient inutile de conserver des sous-motifs étant donnée que les sous-ensembles de motifs sont aussi des motifs. Ainsi, cette étape se charge de déterminer les sous-motifs présents dans L_f (motifs non maximums) et de les éliminer de L_f.
Pour cela, un algorithme débutant à l'étape $k = max_{S_i \in L_f} card(S_i)$ jusqu'à l'étape $k = 2$, peut éliminer de manière itérative tous les sous-ensembles des séquences de taille k appartenant à L_f.
De plus, pour déterminer rapidement les sous-séquences d'une séquence une structure de type hash-tree ([AS94]) peut-être utilisée.

Algorithme 6.2.3 apriori-generate

Entrée: L_{k-1}, Ensemble des séquences fréquentes de taille $k-1$
Sortie: C_k, Ensemble des séquences candidates de taille k

1: $C_k \leftarrow \emptyset$
2: **pour tout** $x = \{x_1, ..., x_{k-1}\} \in L_{k-1}$ **faire**
3: **pour tout** $y = \{y_1, ..., y_{k-1}\} \in L_{k-1} - \{x\} \mid \forall 0 < i < k-1 x_i = y_i$ **faire**
4: $C_k \leftarrow C_k \cup \{\{x_1, ..., x_{k-1}, y_{k-1}\}\}$
5: **fin pour**
6: **fin pour**
7: {Élagage en utilisant le principe d'anti-monotonie de la fonction support}
8: **pour tout** $x = \{x_1, ..., x_k\} \in C_k$ **faire**
9: **si** $\exists s = \{s_1, ..., s_{k-3}, x_{k-1}, x_k\} \prec x \mid s \notin L_{k-1}$ **alors**
10: $C_k \leftarrow C_k - \{x\}$
11: **fin si**
12: **fin pour**

Dans certains algorithmes, cette étape est combinée avec l'étape précédente afin d'éviter de calculer le support inutilement pour les sous-motifs.

6.2.2.3 Algorithme AprioriSome

Le calcul du support des séquences étant le processus le plus coûteux dans la détermination des motifs maximaux, l'objectif d'AprioriSome est de limiter voire d'éviter le calcul du support des séquences non maximales.
Pour cela, AprioriSome se base sur l'observation que lorsqu'à une étape k, le ratio entre le nombre de motifs de tailles k et le nombre de candidats de taille k est élevé alors le pourcentage de chance que ces motifs soient des motifs non maximaux est élevé. Ainsi, une heuristique consisterait à sauter vers une étape $k+n$ où n doit être déduite de manière empirique et ad hoc. Ce saut devrait permettre d'atteindre plus rapidement les motifs maximaux en évitant le calcul des sous-motifs. Toutefois, il est possible que ces sauts aient éliminés des motifs maximaux de taille comprise dans l'intervalle $]k; n[$. Ainsi, il est nécessaire de revenir en arrière afin de déterminer les motifs maximaux qui auraient pu être éliminés.

L'algorithme AprioriSome se compose de deux étapes. La première étape consiste à déterminer, de manière incrémentale pour chaque sous étape $k \geq 0$, l'ensemble L_k des motifs de taille k en effectuant si nécessaire les sauts appropriés.
Pour déterminer si le saut peut être effectué ou non et pour connaître la taille du saut, Agrawal et Srikant fournissent, dans [AS94], l'heuristique décrite par l'algorithme 6.2.4 :

De plus, la première étape faisant apparaître des sauts, à savoir, des sous étapes dans lesquelles l'ensemble des motifs n'est pas déterminé étant donné que le calcul du support n'est pas effectué, la fonction de génération des candidats utilisée dans AprioriAll (voir algorithme 6.2.3)

Algorithme 6.2.4 next

Entrée: k, taille des séquences candidates
Entrée: C_k, Ensemble des candidates de taille k
Entrée: L_k, Ensemble des motifs de taille k
Sortie: $nextStep$, taille des prochaines séquences dont le support doit être calculé ($nextStep - k$ = taille du saut

1: $hit_k \leftarrow \frac{card(L_k)}{card(C_k)}$
2: **si** $hit_k < 0.666$ **alors**
3: $nextStep \leftarrow k + 1$
4: **sinon si** $hit_k < 0.75$ **alors**
5: $nextStep \leftarrow k + 2$
6: **sinon si** $hit_k < 0.80$ **alors**
7: $nextStep \leftarrow k + 3$
8: **sinon si** $hit_k < 0.85$ **alors**
9: $nextStep \leftarrow k + 4$
10: **sinon**
11: $nextStep \leftarrow k + 5$
12: **fin si**

n'est plus valide telle quelle. En effet, apriori-generate utilise, à l'étape k, l'ensemble L_{k-1} des motifs de taille $k-1$ afin de générer l'ensemble C_k des candidats de taille k. Cependant, cette fonction peut être utilisée en remplaçant L_{k-1}, lorsque celui-ci n'a pas été déterminé, par C_{k-1}. En effet, la complétude des motifs de taille k présent dans C_k est maintenu car $L_{k-1} \subseteq C_{k-1}$. Il est alors nécessaire, à chaque étape k, de calculer C_k.

La seconde étape de l'algorithme consiste à déterminer les motifs maximaux éliminés par les sauts de la première étape. Ainsi, on calcul les ensembles L_k, ignorés par la première étape, dans l'ordre des tailles k décroissants. Ce calcul s'effectue en utilisant l'ensemble C_k des candidats de taille k généré par la première étape. Les candidats appartenant à C_k et étant des sous-séquences d'une séquence appartenant à $L_j,\ \forall j > k$ est éliminé de C_k. Le calcul du support s'effectue pour les éléments restant de C_k afin de déterminer L_k contenant les séquences de C_k ayant un support supérieur ou égal au seuil minimum.
De plus, on peut remarquer que la première étape a généré non seulement des motifs maximaux mais aussi des sous motifs de ces derniers. Ainsi, lors de cette deuxième étape, lorsqu'un ensemble L_k a déjà été déterminé (dans la première étape), les motifs non maximum de L_k peuvent être éliminés. Pour cela, il suffit d'éliminer de L_k, les éléments étant des sous-séquences d'au moins une séquence appartenant à $L_j,\ \forall j > k$.

6.2.2.4 Algorithme DynamicSome

Dans le même esprit que l'algorithme AprioriSome, DynamicSome, défini dans [AS94], propose d'effectuer des "sauts" réguliers dans le but de trouver les motifs maximaux de séquences.

Ainsi, étant donné un entier, pas, strictement positif déterminant le pas de "saut", l'algorithme est constitué de quatre phases :

1. **la phase d'initialisation** qui permet de calculer l'ensemble L_k des séquences fréquentes de taille k ($\forall k \in [1; pas]$),
2. **la phase "forward"** qui permet de calculer tous les ensembles L_k des séquences fréquentes pour tout $k > 1$ multiple de pas,
3. **la phase intermédiaire** qui permet de calculer les ensembles C_k des séquences candidates ignorés dans les deux premières phases. Cette phase est nécessaire pour le passage à la prochaine phase. En effet, la prochaine phase utilise les ensembles de candidats pour déterminer les ensembles de motifs.
4. **la phase de "backtracking"** qui permet de calculer les ensembles L_k, des séquences fréquentes, qui ont été ignorés dans les phases précédentes à savoir les ensembles L_k tel que $k > pas$ et k n'est pas un multiple de pas. Cette phase est identique à la deuxième phase d'AprioriSome.

La phase "forward" est utilisée pour déterminer des séquences de taille k multiple de pas. Ainsi, il existe une discontinuité entre $k - pas$ et k. Cette discontinuité se traduit par l'absence de l'ensemble L_{k-1} ou C_{k-1} nécessaire pour générer l'ensemble L_k des candidats par l'algorithme apriorigenerate (algorithme 6.2.3). Il est alors nécessaire de définir une nouvelle fonction de génération de candidat de taille k (multiple de pas) à partir des ensembles L_{k-pas} et L_{pas}.
La nouvelle fonction de génération, $otfgenerate$, utilise l'observation que pour une séquence x de taille $k - pas$, une séquence y de taille pas et une séquence c, si x et y sont contenues dans c et x et y ne se chevauchent pas dans c alors $x \cdot y$ est un candidat de taille k Deux séquences x et y ne se chevauchent pas dans une séquence c, si le dernier élément de x apparaît, dans c, avant le premier élément de y. On a, ainsi, pour la fonction de génération, l'algorithme 6.2.5.

6.2.2.5 Apports

Agrawal et Srikant ont formalisé, dans [AS95], un nouveau problème entrant dans le cadre du Data-Mining à savoir l'extraction de motifs séquentiels pour la génération de règles d'associations. De plus, ils ont apporté une solution constituée de trois algorithmes dont les éléments serviront de bases pour les autres algorithmes créant ainsi la famille d'algorithme "Apriori".

6.2.2.6 Résultats

Les résultats mis à jours dans [AS95] indiquent que les algorithmes "AprioriAll" et "AprioriSome" sont plus rapides que "DynamicSome". Ces résultats s'expliquent essentiellement par l'utilisation de l'algorithme de génération de candidats "$otf - generate$". En effet, cet algorithme génère beaucoup plus de candidats que "$apriori - generate$". Ainsi, "DynamicSome" passe beaucoup de temps à déterminer les supports des candidats.
Les différences de résultats entre "AprioriAll" et "AprioriSome" sont négligeable. Bien que

Algorithme 6.2.5 otf-generate

Entrée: L_k, Ensemble des séquences fréquentes de taille k

Entrée: L_{pas}, Ensemble des séquences fréquentes de taille pas

Entrée: $c =< c_1, \ldots, c_n >$, Séquence principale

Sortie: Z_{k+pas}, Ensemble des séquences candidates de taille $k + pas$ apparaissant dans c et dont les préfixes de taille k appartiennent à L_k et les suffixes de taille pas appartiennent à L_{pas}.

1: $X_k \leftarrow \{x \mid x \in L_k \; et \; x \preceq c\}$
2: **pour tout** $x \in X_k$ **faire**
3: $\quad x.end \leftarrow \min(j \mid x \preceq < c_1, \ldots, c_j >)$
4: **fin pour**
5: $Y_{pas} \leftarrow \{y \mid y \in L_{pas} \; et \; y \preceq c\}$
6: **pour tout** $y \in Y_k$ **faire**
7: $\quad y.start \leftarrow \max(j \mid y \preceq < c_j, \ldots, c_n >)$
8: **fin pour**
9: $Z_{k+pas} \leftarrow \{z \mid \forall x, y \in X_k \times Y_{pas}, \; z = x \cdot y \; et \; x.end < y.start\}$

"AprioriSome" soit légèrement plus rapide dû au fait qu'il évite de déterminer les supports des séquences non-maximales, il génère beaucoup plus de candidats que "AprioriAll". En effet, "AprioriSome" génère, dans certains cas, ces candidats à partir de l'ensemble de candidats C_{k-1} alors que "AprioriAll" utilise l'ensemble des motifs L_{k-1}. Or étant donné que $L_{k-1} \subseteq C_{k-1}$, l'ensemble des candidats générés à partir de C_{k-1} sera plus grand que celui généré à partir de L_{k-1} . Ainsi, les performances de "AprioriSome" se dégradent dans certains cas par le fait qu'il consomme plus de mémoire et plus de temps à déterminer les supports de candidats que "AprioriAll".

6.2.3 Algorithme GSP

Srikant et Agrawal proposent, dans [SA96], l'algorithme GSP. Cet algorithme permet d'une part d'améliorer les performances d' "AprioriAll" et d'autre part de généraliser le problème de la recherche de motifs séquentiels. Cette généralisation du problème peut se résumer en trois points :

1. l'introduction de taxonomies pour les éléments des évènements d'un séquences,
2. la redéfinition de sous-évènements,
3. la redéfinition de sous-séquences,

6.2.3.1 Généralisation du problème

Taxonomies

La taxonomie permet de définir une classification, pour certains ou tous les éléments d'un évènement. Ainsi, elle permet de définir des relations de généralisations et de spécifications pour chaque élément possédant une taxonomie.
Ainsi, si deux éléments a et b sont reliés par une taxonomie et que a est l'ancêtre de b, dans cette classification, alors a est plus général que b.
En tenant compte de cette taxonomie, deux éléments a et b concordent s'ils sont identiques ou si l'un des deux est une généralisation de l'autre. Par conséquent, un évènement E contient un élément a si a ou, au moins, un de ces ancêtres (éléments plus généraux) appartient à E.

Redéfinition des sous-évènements

Définition 6.2.16 (Distance)**.** *Le terme distance est, ici, employé dans un sens large. Il peut être associé à la dimension temporelle comme ce sera souvent le cas dans le domaine des séquences ou encore à toutes autres dimensions quantifiables. Nous nous limiterons à une unique dimension $\mathcal{T}$ pour le calcul de la distance entre deux évènements.*
Ainsi, une distance entre deux évènements e_1 et e_2 peut être exprimée par la fonction :
$dist_{\mathcal{T}}(e_1, e_2) =\mid e_{2\mid\mathcal{T}} - e_{1\mid\mathcal{T}} \mid$ où $e_{\mid\mathcal{T}}$ est la valeur de la projection de e sur $\mathcal{T}$.
Nous utiliserons dist pour exprimer $dist_{\mathcal{T}}$ sans définir $\mathcal{T}$ qui sera implicite au domaine d'application.

Étant donné la taille d'une fenêtre *window_size* définit en terme de distance, un évènement e est dit sous-évènement d'une séquence $S =< s_1, \ldots, s_n >$, s'il existe deux entiers i et j tel que $1 \leq i \leq j \leq n$ et $e \subseteq \cup_{k=i}^{j} s_k$ et $dist(s_i, s_j) \leq window\_size$.

Redéfinition des sous-séquences

L'introduction d'un seuil minimum (*dist_min*) et maximum (*dist_max*) permet de définir des contraintes de distance entre les évènements adjacents d'une séquence.
Ainsi, une séquence $S =< s_1, \ldots, s_n >$ sera dite sous-séquence de $E =< e_1, \ldots, e_{p \geq n} >$ s'il existe une série d'entier $l_1 \leq u_1 < \ldots < l_n \leq u_n \leq p$ tel que $\forall i \in \{1, \ldots, n\}$:

1. $s_i \subseteq \cup_{k=l_i}^{u_i} e_k$,
2. $dist(e_{u_i}, e_{l_i}) \leq window\_size$,
3. $\forall i > 1,\ dist(e_{l_i}, e_{u_{i-1}}) > dist\_min$,
4. $\forall i > 1,\ dist(e_{u_i}, e_{l_{i-1}})) \leq dist\_max$.

6.2.3.2 L'algorithme

L'algorithme *GSP* fait partie de la famille des algorithmes "Apriori" par le fait qu'il possède les trois caractéristiques de cette famille :

- Génération incrémentale des candidats.
- Utilisation du principe d'anti-monotonie pour l'élagage des sur-séquences de séquences non-fréquentes.

– Calcul du support des candidats en effectuant une lecture de la base.

L'algorithme *GSP* (algorithme 6.2.6) est un algorithme itératif reposant sur deux grandes phases :

1. Génération de séquences "candidates" de taille totale k (séquences ayant k éléments) à partir des motifs de séquences de taille totale $k-1$.
2. Calcul du support des séquences "candidates" en effectuant un parcours de la base des séquences et élimination des séquences "candidates non fréquentes.

Algorithme 6.2.6 GSP

Entrée: $\mathcal{D}_\mathcal{T}$, Base de séquences
Entrée: L_1, Ensemble des séquences fréquentes de taille totale 1
Sortie: L_f, Ensemble des motifs de séquences

```
1: k ← 2
2: tant que L_{k-1} ≠ ∅ faire
3:    C_k ← gsp − generate(L_{k-1})
4:    pour tout S ∈ D_T faire
5:       gsp − count(S, C_k)
6:    fin pour
7:    L_k ← {c ∈ C_k | c.support ≥ minsup}
8:    k ← k + 1
9: fin tant que
10: L_f ← ∪_k L_k
```

Phase de génération des séquences candidates

La procédure de génération de candidats utilise la propriété des séquences contiguës. Cette propriété indique que si une séquence S est sous-séquence de S' alors toutes ses sous-séquences contiguës sont, elles aussi, des sous-séquences de S' en respect avec les contraintes de temps (fenêtre temporelle, seuils minimum et maximum des gaps entre les évènements adjacents d'une séquence).

Définition 6.2.17 (Séquence contiguë)**.** *Soit une séquence $S =< s_1, \ldots, s_n >$, une sous-séquence c est dite sous-séquence contiguë de S si au moins une des conditions suivantes est vérifiée :*

1. *c est obtenu en éliminant un élément de s_1 ou de s_n.*
2. *c est obtenu en éliminant un élément de s_i contenant au moins deux éléments.*
3. *c est une sous-séquence contiguë de c' avec c' étant une sous-séquence contiguë de c.*

Étant donné l'ensemble L_n des motifs de taille totale n, la propriété des séquences contiguës indique que si une séquence S de taille $n+1$ est fréquente alors toutes ses sous-séquences contiguës de taille totale n appartiennent à L_n.

Ainsi, pour générer l'ensemble des séquences candidates C_{n+1}, la procédure va effectuer une jointure de L_n avec L_n. Deux séquences $S_1 =< s_1, \ldots, s_p >$ et S_2 peuvent être jointes seulement si la sous-séquence obtenue en éliminant un élément du premier évènement de S_1, est identique à celle obtenu en éliminant un élément e du dernier évènement de S_2. Le résultat de la jointure est alors la séquence $S =< s_1, \ldots, s_p, \{e\} >$ si e est l'unique élément du dernier évènement de S_2 ou $S =< s_1, \ldots, s_p \cup \{e\} >$ sinon [2]. Une séquence, de taille totale $n+1$, obtenue par jointure ne peut être candidate (i.e. appartenir à C_{n+1}) que si toutes ses sous-séquences contiguës de taille totale n sont fréquentes (i.e. appartiennent à L_n). L'algorithme 6.2.7 décrit la génération des séquences candidates.

Algorithme 6.2.7 gsp-generate

Entrée: $\mathcal{D}_\mathcal{T}$, Base de séquences
Entrée: L_k, Ensemble des séquences fréquentes de taille totale k
Sortie: C_{k+1}, Ensemble des motifs candidats

1: $C_{k+1} \leftarrow \emptyset$
2: **si** $k = 1$ **alors**
3: **pour tout** $x =< \{x_1\} > \in L_k$ **et** $y =< \{y_1\} > \in L_k$ **faire**
4: $C_{k+1} \leftarrow C_{k+1} \cup \{< \{x_1\}, \{y_1\} >\}$
5: **si** $x \neq y$ **alors** $C_{k+1} \leftarrow C_{k+1} \cup \{< \{x_1, y_1\} >\}$
6: **fin pour**
7: **sinon**
8: **pour tout** $x =< x_1 \cup \{e_x\}, \ldots, x_p > \in L_k$ **et** $y \in L_k$ **faire**
9: $S \leftarrow \emptyset$
10: **si** $y =< x_1, \ldots, x_p \cup \{e_y\} >$ **alors**
11: $S \leftarrow < x_1 \cup \{e_x\}, \ldots, x_p \cup \{e_y\} >$
12: **sinon**
13: **si** $y =< x_1, \ldots, x_p, \{e_y\} >$ **alors** $S \leftarrow < x_1 \cup \{e_x\}, \ldots, x_p, \{e_y\} >$
14: **fin si**
15: **si** $S \neq \emptyset$ **et** $\forall s_i \in S \mid card(s_i) > 1$
et $\forall e \in s_i, < s_1, \ldots, s_i - \{e\}, \ldots, s_{card(S)} > \in L_k$ **alors**
16: $C_{k+1} \leftarrow C_{k+1} \cup \{S\}$
17: **fin si**
18: **fin pour**
19: **fin si**

Calcul du support

Le problème principal pour le calcul du support est de déterminer si une séquence $S =< s_1, \ldots, s_n >$ est une sous-séquence d'une séquence D tout en tenant compte des contraintes de

[2] La jointure de deux séquences de taille totale 1 produira deux séquences : $S =< s_1, \ldots, s_p, \{e\} >$ et $S' =< s_1, \ldots, s_p \cup \{e\} >$. En effet, pour ce cas particulier, S_1 et S_2 sont des sous-séquences contiguës de S et de S'.

temps. La procédure proposée pour déterminer si S est une sous-séquence de $D =< d_1, \ldots, d_p >$ repose sur deux étapes :

1. **étape "forward" :** L'algorithme recherche successivement les évènements s_i de S dans D tant que $(end\_time(s_i) - start\_time(s_{i-1})) \leq dist\_max$ [3].
   Si $(end\_time(s_i) - start\_time(s_{i-1}) > dist\_max$, l'algorithme passe alors à l'étape "backward". Si par contre l'évènement s_i n'apparaît pas dans D alors S n'est pas une sous-séquence de D.
2. **étape "backward" :** La phase de "backward" est atteinte lorsque :

   $$(end\_time(s_i) - start\_time(s_{i-1}) > dist\_max$$

   Dans ce cas, l'algorithme recherche l'évènement s_{i-1} dans D de telle sorte que

   $$start\_time(s_{i-1}) \geq end\_time(s_i) - dist\_max$$

   Si la projection de s_{i-1} sur D a pu être rajustée, l'algorithme réitère, si nécessaire, l'étape "backward" pour réajuster les évènements précédents. Si le réajustement a pu être effectué, l'algorithme repasse à l'étape "forward" sinon la séquence D ne contient pas S.

Pour la recherche d'un élément d'un évènement s_i d'une séquence S dans une séquence D, un tableau ayant autant de case que d'éléments (d'évènements) est utilisé. Pour un élément donné, la case du tableau associée contiendra les "dates", dans l'ordre chronologique, des évènements de D contenant l'élément.

De plus, afin d'éviter d'effectuer le test de sous-séquence pour chaque séquence candidate appartenant à C_k et pour chacune des séquences de la base. L'ensemble C_k peut être représenté sous forme d'arbre de hachage. Chaque feuille contenant un sous-ensemble de C_k. Les noeuds intérieurs contiennent une table de hachage associant un élément à un sous-arbre. Pour une séquence donnée, le parcours s'effectue à un noeud n en hachant l'élément n de la séquence.

Taxonomies

L'idée pour prendre en compte les taxonomies dans l'algorithme GSP est de remplacer chaque séquence de la base par une séquence dite étendue. La séquence contiendra, en plus de la séquence elle même, dans chaque évènement les éléments de cet évènement ainsi que tous les ancêtres de chaque élément.

6.2.3.3 Apports

Srikant et Agrawal ont étendu le problème de la recherche de motifs séquentiels en introduisant des contraintes temporelles comme la définition de gap et de fenêtre temporelle et les

[3] Soit $s_i \subseteq \cup_{k=u}^{v} d_k$, $start\_time(s_i) = d_{u|\mathcal{T}} = time(d_u)$ et $end\_time(s_i) = d_{v|\mathcal{T}} = time(d_v)$.

taxonomies.
De plus, l'algorithme GSP proposé améliore de façon considérable l'algorithme AprioriAll.

6.2.3.4 Résultats

Les résultats expérimentaux indiqués dans [SA96] montrent que GSP est 3 à 5 fois plus rapide que l'algorithme AprioriAll. Ce résultat provient essentiellement du fait que GSP génère beaucoup moins de séquences candidates qu'AprioriAll. En effet, GSP génère ses candidates en étendant une séquence d'un élément alors qu'AprioriAll génère en étendant la séquence candidate d'un motif d'évènements. Or l'espace des motifs d'évènements est beaucoup plus vaste que l'espace des éléments fréquents (motifs de taille 1).

6.3 bitSPADE : un nouvel algorithme d'extraction de séquences fréquentes

Sequential pattern mining is an active field of research since it was, originally, introduced by Agrawal and Srikant [AS95]. Given a dataset of sequences, the mining task is to discover sequential patterns expressing temporal relationships between items shared by a sufficient number of sequences to be considered as relevant. The extracted patterns can be used for generating association rules. For example assume a customer dataset, if "bought a television" and "bought a television then a DVD player" are patterns contained in, respectively, 80% and 72% of the customer sequences of the dataset then we can generate the rule "a customer who buy a television will buy a DVD player" with a confidence of $\frac{0.72}{0.80} = 90\%$. Although the problem was motivated by applications such as customer satisfaction, it has been successfully applied to many other domains including medical and genetics.
Usually, sequential patterns need to be extracted from very large databases. Thus, the mining algorithms must take into account the space-time tradeoff.

In this section, we propose to merge the best features of SPADE [Zak98], a memory efficient algorithm, and SPAM [AGYF02], a speed efficient algorithm, into bitSPADE a new mining algorithm. Our main contributions are the following : 1) We introduce the concept of a semi vertical database using the bitmap representation of SPAM. Combining the semi vertical database with the SPADE's lattice decomposition into independent equivalence classes allows a fast and efficient enumeration of frequent patterns. 2) We present a new pruning strategy that can be applied, independently, to each equivalence class. 3) bitSPADE outperforms SPADE and SPAM in terms of memory usage and ensures the best space-time tradeoff between the existing approaches.

The rest of this section is organized as follows. In section 6.3.1, we define the sequential pattern mining problem. In section 6.3.1, we briefly discuss the related work. Section 6.3.3 focuses on the SPADE algorithm since our method is based on. This is followed by a section describing the bitSPADE algorithm. Then, we present the experimental comparisons and the performance results of the algorithm.

6.3.1 Preliminaries

The problem of sequence mining can be stated as follow : Let $\mathcal{I} = i_1, \ldots, i_m$ be an alphabet of m distinct items. A transaction is an unordered set of items (without loss of generality, we can assume that the set is lexicographically ordered) and a sequence S is a time ordered list of transactions denoted by $S = s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_n$ with $s_i \subseteq \mathcal{I}$ for $i \in \{1, \ldots, n\}$. The size of the sequence S, denoted by $|S|$, is the number of transactions in S and the length of S is the number of items composing the sequence ($l = \sum |s_i|$ where $|s_i|$ is the number of items in the transaction s_i). A sequence of length l is called an l-sequence. A sequence $S = s_1 \rightarrow \ldots \rightarrow s_n$ is called a supersequence of another sequence $U = u_1 \rightarrow \ldots \rightarrow u_p$ (S contains U) and U a subsequence of S (U is contained in S), denoted by $U \preceq S$, if there exists a one-to-one order-preserving function $f : \{u_1, \ldots, u_p\} \rightarrow \{s_1, \ldots, s_n\}$ such that, 1) $\forall i \in \{1, \ldots, p\}$, $u_i \subseteq f(u_i)$, 2) $\forall i, t \in \{1, \ldots, p\}$ with $i < t$, $\exists j, k \in \{1, \ldots, n\}$ with $j < k$ such that $f(u_i) = s_j$ and $f(u_t) = s_k$. A sequence dataset $\mathcal{D}$ is a set of tuples (sid, S) where sid is the unique sequence id of the sequence S. A tuple (sid, S) is said to contain a sequence U if U is a subsequence of S. The support or frequency of a sequence S in a dataset $\mathcal{D}$, denoted by $\sigma_{\mathcal{D}}(S)$, is the number of tuples in $\mathcal{D}$ containing S. Given a user-defined threshold called the minimum support, denoted by min_supp, a sequence S is said to be a frequent sequence or pattern in a dataset $\mathcal{D}$ if $\sigma_{\mathcal{D}}(S) \geq min\_supp$. A maximal frequent sequence is a sequential pattern that has no frequent supersequence. Given a frequent sequence β in $\mathcal{D}$, the set of association rules that can be generated from β is $\{\alpha \Rightarrow \beta \mid \alpha \preceq \beta\}$ and the confidence for a rule $\alpha \Rightarrow \beta$ is $conf_{\mathcal{D}}(\alpha \Rightarrow \beta) = \frac{\sigma_{\mathcal{D}}(\beta)}{\sigma_{\mathcal{D}}(\alpha)}$. The association rule generation being straightforward, given a dataset $\mathcal{D}$ and a minimum support threshold min_supp the sequence mining problem lies in finding the set of all frequent sequences in $\mathcal{D}$.

6.3.2 Related work

The problem of sequence mining was introduced in [AS95], in which three algorithms were presented. The three methods were based on the same approach which is the Apriori approach. The Apriori strategy is based on an iterative method composed of two main stages. The first stage is to generate candidate sequences of size k from frequent sequences of size $k-1$. The second stage is to test and prune the candidates by evaluating their support by a pass over the dataset. In subsequent work, the same authors proposed the GSP algorithm [SA96]. GSP improved the previous algorithms by generating candidates of length k from sequential patterns of length $k-1$. Many other algorithms have adopted this approach [MCP98, Zak98, AGYF02, LRBE03, OPS04]. The other approach is the pattern growth method [HPMA$^+$00, PHMA$^+$01, LLW02, HPY05]. This method recursively grows patterns by performing database projection according to the frequent items.
SPADE [Zak98] and SPAM [AGYF02] are two algorithms that belong to the Apriori family. SPADE is one of the most memory efficient algorithm. It uses a lattice to efficiently enumerate frequent sequences. The lattice can be partitioned into equivalence classes that can be explored independently. Thus, SPADE can handle large databases with low memory consumption. Moreover, SPADE uses a vertical database for fast candidate support counting. SPAM is one

of the fastest algorithm to mine frequent sequential patterns. It uses a lexicographic sequence tree with a pure depth first traversal to navigate through the sequence space. This frequent sequences enumeration assumes that the whole dataset fit in memory. For fast counting, SPAM uses a vertical bitmap representation of the database. Our new method, based on SPADE, uses the best features of both algorithms and thus it ensures the best space-time tradeoff.

6.3.3 The SPADE Algorithm

SPADE is a candidate-generation-and-test algorithm. Thus, it needs to enumerate all the potential frequent sequences and to test each candidate by counting his support. The main features of SPADE are an efficient sequence enumeration and the use of a vertical database which allows fast support counting by simple join operations.

6.3.3.1 Sequence Enumeration

The sequence lattice used by SPADE allows to efficiently order the sequence space. Given an alphabet of items $\mathcal{I}$ and the subsequence relation $\preceq$, $(L, \preceq)$, with L the set of all sequences, is a complete lattice. The bottom of the lattice is the empty sequence denoted by $\emptyset$ and the top is undefined since the lattice is infinite. The join of a set S of sequences, denoted by $\bigvee S$, is the set of minimal common supersequences ($\bigvee S = \{\alpha \mid \forall \beta \in S, \beta \preceq \alpha$ and $\forall \gamma \in L, \beta \preceq \gamma \preceq \alpha \Rightarrow \alpha = \gamma\}$). The meet of the set S , denoted by $\bigwedge S$, is the set of maximal common subsequences ($\bigwedge S = \{\alpha \mid \forall \beta \in S, \alpha \preceq \beta$ and $\forall \gamma \in L, \alpha \preceq \gamma \preceq \beta \Rightarrow \alpha = \gamma\}$). In fact, $(L, \preceq)$ is an hyper-lattice since the meet and the join operations do not, necessarily, produce unique elements. Indeed, the join of two items, A and B, will produce the set $\{\{A, B\}, \{A\} \rightarrow \{B\}, \{B\} \rightarrow \{A\}\}$. Any k-sequence of the lattice can be obtained with a join on two of its $k-1$ length subsequences. Moreover, SPADE, assumes, that each set of k-sequences of the lattice, i.e. the set of sequences at the level k of the lattice, are lexicographically sorted with the partial relation order $\leq$. Given two k-sequences α and β, sharing a common prefix of length $k-1$, $\alpha \leq \beta$ if $|\alpha| < |\beta|$ or, in the case of $|\alpha| = |\beta|$, if α lexicographically smaller or equal than β.
Using the lattice approach, sequences can be enumerated with a bottom-up traversal. However, with very large data-bases, the lattice may not fit in memory. To solve the problem, the lattice can be decomposed into equivalence classes that can be processed, independently, in memory. An equivalence class $[X]_{\theta_k}$ is a set of sequences that share the k-sequence X as a common prefix. Given two sequences $U = u_1 \rightarrow \ldots \rightarrow u_n$ and $X = x_1 \rightarrow \ldots \rightarrow x_k$ X is said to be a prefix of U if $k \leq n$ and $\forall i \in \{1, \ldots, k-1\}, x_i = u_i$ and $x_k \subseteq u_k$. Thus, the equivalence class $[X]_{\theta_k}$ is a sub-lattice with the sequence X as the bottom of the lattice. Sequences of length $k+1$ are said to be the atoms of $[X]_{\theta_k}$ and atoms with the same size as X are called itemset atoms whereas the others are called sequence atoms. An equivalence class can recursively be decomposed into smaller equivalence classes. This property will be used in a depth first search limiting the join operation to atoms.

Enumerate-Frequent-Seq(S)

Input: S : The lexicographically ordered atoms of $[X]_{\theta_k}$
Output: F : The set of frequent sequences

1: **for all** atoms $A_i \in S$ **do**
2: $T_i \leftarrow \emptyset$
3: **for all** atoms $A_j \in S$ **do**
4: $R_1 \leftarrow R_2 \leftarrow \emptyset$
5: **if** A_i an itemset atom **then**
6: **if** A_j an itemset atom and $i < j$ **then**
7: $R_1 \leftarrow$ equality-join(A_i,A_j)
8: **else if** A_j a sequence atom **then**
9: $R_1 \leftarrow$ temporal-join(A_i,A_j)
10: **end if**
11: **else if** A_j a sequence atom **then**
12: **if** $i < j$ **then**
13: $R_1 \leftarrow$ equality-join(A_i,A_j)
14: **end if**
15: $R_2 \leftarrow$ temporal-join(A_i,A_j)
16: **end if**
17: **for all** $R_p \neq \emptyset$ and $support(R_p) \geq min\_supp$ **do**
18: $T_i \leftarrow T_i \cup \{R_p\}$
19: **end for**
20: **end for**
21: **if** Depth-First-Search **then**
22: $F \leftarrow F \cup$ Enumerate-Frequent-Seq(T_i)
23: **end if**
24: **end for**
25: **if** Breadth-First-Search **then**
26: **for all** $T_i \neq \emptyset$ **do**
27: $F \leftarrow F \cup$ Enumerate-Frequent-Seq(T_i)
28: **end for**
29: **end if**

Fig. 6.1 – Pseudo-code for SPADE

6.3.3.2 Sequence joining

SPADE begins by computing the frequent sequences of length 1 and 2 by reading the dataset. Then, it recursively enumerates all the frequent sequences within each equivalence class starting with classes $[X]_{\theta_1}$ and the corresponding atoms of length 2. Figure 6.1 shows the pseudo-code for frequent sequence enumeration within a class defined by his set of atoms. Assume P is the prefix sequence of the current class, PA and PB two sequences obtained by adding items A and B to the last transaction of P, when joining two atoms α and β, there can be up to three cases : 1) $\alpha = PA$ and $\beta = PB$ are two itemset atoms with $A < B$, the join will be an equality-join producing PAB, 2) $\alpha = PA$ an itemset atom and $\beta = P \rightarrow \{B\}$ a sequence atom, a temporal join will produce $PA \rightarrow \{B\}$ and 3) $\alpha = P \rightarrow \{A\}$ and $\beta = P \rightarrow \{B\}$ two sequence atoms, the join will be composed of an equality-join, if $A < B$, producing $P \rightarrow \{AB\}$ and of a temporal-join producing $P \rightarrow \{A\} \rightarrow \{B\}$

6.3.3.3 Pruning

The Apriori property states that all the subsequences of a frequent sequence are frequents [AS95]. Thus, the property leads to the efficient Apriori pruning strategy that can be defined as follows : given a sequence S, if there exists, at least, a subsequence of S that is not frequent then S and his supersequences can not be frequents. According to this property, if a sequence X is not frequent then the equivalence class $[X]_{\theta_k}$ can be pruned and X doesn't need to be used in any join operations since all the sequences that can be obtained won't be frequent.

6.3.3.4 Data Representation

Traditional algorithms, such as AprioriAll, that use conventional horizontal databases, like the one shown in table 6.1, need to make a pass over the whole dataset to compute the sequence supports. This technique is clearly inefficient because either it needs sufficient memory to store the dataset or it must make several disk accesses. A better representation is introduced in SPADE which is a vertical database. A vertical database is a set of id-lists and to each item of the original dataset is associated an id-list. The id-list for an item A, denoted by $\mathcal{L}(A)$, is a set of tuples (SID, TID) where TID is the id of the transaction of sequence SID in which A appears. Table 6.2 shows the vertical database for sequences of table 6.1. Given a vertical database, joining two $(k+1)$-sequence atoms of class $[X]_{\theta_k}$ $\alpha = \alpha_1 \rightarrow \ldots \rightarrow \alpha_m$ $(m \in \{k, k+1\})$ and $\beta = \beta_1 \rightarrow \ldots \rightarrow \beta_n$ $(n \in \{k, k+1\}$ and I_β being the last item of $\beta_n)$ with id-lists, respectively, $\mathcal{L}(\alpha)$ and $\mathcal{L}(\beta)$ is straightforward. The equality-join of α with β, if defined, will produce $\gamma = \alpha_1 \rightarrow \ldots \rightarrow (\alpha_m \cup \{I_\beta\})$ with $\mathcal{L}(\gamma) = \{(s,t) \mid (s,t) \in \mathcal{L}(\alpha)\ and\ (s,t) \in \mathcal{L}(\beta)\}$ and the temporal join of α with β, if defined, will produce $\delta = \alpha_1 \rightarrow \ldots \rightarrow \alpha_m \rightarrow \{I_\beta\}$ with $\mathcal{L}(\delta) = \{(s,t) \mid (s,t) \in \mathcal{L}(\beta)\ and\ \exists (s,t_0) \in \mathcal{L}(\alpha)\ with\ t_0 < t\}$. The support for a sequence S is simply the number of tuples with different SID in $\mathcal{L}(S)$.

SID	Sequence
1	$\{A, B\} \rightarrow \{C\} \rightarrow \{B, C\}$
2	$\{B\} \rightarrow \{A, B, C\}$
3	$\{A, B\} \rightarrow \{C\}$

Tab. 6.1 – A sequence dataset

A	
SID	TID
1	1
2	2
3	1

B	
SID	TID
1	1
1	3
2	1
2	2
3	1

C	
SID	TID
1	2
1	3
2	2
3	2

Tab. 6.2 – The vertical dataset for table 6.1

6.3.4 The bitSPADE Algorithm

In this section, we describe how the lattice concept of SPADE is used in bitSPADE. We, also, discuss about a new method to prune sub-lattices within an equivalence class. Finally, we show how to speed up supports counting by using an appropriate semi-vertical bitmap representation.

6.3.4.1 Sequence Enumeration

bitSPADE uses the same lattice decomposition as SPADE since it is the main factor of the memory performance of SPADE. Thus, it recursively explores the equivalence classes as shown in figure 6.1. However, bitSPADE uses only a depth first traversal of the lattice since our aim is to reduce the memory usage. Indeed, a breadth first traversal, needs to store, in memory, all the sequences, with their id-list, produced by joining each atom of the class before moving to another class. Moreover, bitSPADE will explore each class in a reverse lexicographic order. Thus, given two equivalence classes $[X]_{\theta_k}$ and $[Y]_{\theta_k}$, if $X < Y$ then $[Y]_{\theta_k}$ will completely be processed before exploring $[X]_{\theta_k}$. This heuristic is introduced for pruning purposes and it will be explained in the next paragraph.

6.3.4.2 Pruning

bitSPADE uses a new pruning method. Since bitSPADE processes equivalence classes independently with a depth first traversal, for a given k-sequence, it doesn't have access to the whole set of $(k-1)$-sequences to perform an efficient pruning. Nevertheless in this section we show how processed classes can be used to prune the currently explored class.
Assume bitSPADE is processing the class $[X]_{\theta_k}$ with $k > 1$, let S be the sequence obtained by

dropping the $(k-1)$'th item (if the item is within a transaction of size 1 then the whole transaction is dropped) and U the sequence obtained by dropping the last item, bitSPADE first checks if the class $[S]_{\theta_{k-1}}$ has completely been processed. If so, $[S]_{\theta_{k-1}}$ will be used to prune $[X]_{\theta_k}$. For any $k+1$-sequence V with V an atom of $[X]_{\theta_k}$, V is a candidate (potentially frequent) if the sequence V_S, obtained by dropping the $(k-2)$'th item of V, is a frequent sequence of $[S]_{\theta_{k-1}}$ and if the sequence V_U, obtained by dropping the $(k-1)$'th item of U, is a frequent sequence of $[U]_{\theta_{k-1}}$. If V is a frequent sequence then bitSPADE will recursively process and prune $[V]_{\theta_{k+1}}$ using the processed class $[V_S]_{\theta_k}$.

Moreover we assume that event atoms of a class will be processed before any sequence atoms and atoms will be ordered according to the last items using a reverse lexicographic order. Given two same sized k-sequences S_1 and S_2 of the class $[S]_{\theta_{k-1}}$, with $S_1 < S_2$, the heuristic assures that S_2 will completely be processed before S_1. This heuristic is used for efficiency since the candidate event atom set of $[S_2]_{\theta_k}$ will always be smaller or equal than those of $[S_1]_{\theta_k}$ and thus processing $[S_2]_{\theta_k}$ will require less time than for $[S_1]_{\theta_k}$.

Figure 6.2 illustrates the bitSPADE pruning strategy. When processing the class $[S_1 = \{C\} \rightarrow \{BC\}]_{\theta_3}$, bitSPADE checks if the class $[S_0 = \{C\} \rightarrow \{C\}]_{\theta_2}$ has already been completely processed (it's always the case since it processes classes in a reverse lexicographic order). If $[\{C\} \rightarrow \{C\}]_{\theta_2}$ has been processed, then the frequent sequence sub-lattice induced by this class will be used to explore $[S_1]_{\theta_3}$. Thus, for an item Y, $\{C\} \rightarrow \{BC\} \rightarrow \{Y\}$ (resp. $\{C\} \rightarrow \{BCY\}$ if $C < Y$) is a relevant candidate if $\{C\} \rightarrow \{C\} \rightarrow \{Y\}$ (resp. $\{C\} \rightarrow \{CY\}$) is a frequent sequence atom (resp. event atom) of $[S_0]_{\theta_2}$ and if $\{C\} \rightarrow \{B\} \rightarrow \{Y\}$ (resp. $\{C\} \rightarrow \{BY\}$) is a frequent sequence atom (resp. event atom) of $[S_1 = \{C\} \rightarrow \{B\}]_{\theta_2}$. Recursively, if $\{C\} \rightarrow \{BC\} \rightarrow \{Y\}$ (resp. $\{C\} \rightarrow \{BCY\}$) is frequent then $[\{C\} \rightarrow \{BC\} \rightarrow \{Y\}]_{\theta_4}$ (resp. $[\{C\} \rightarrow \{BCY\}]_{\theta_4}$) will be processed using the explored class $[\{C\} \rightarrow \{C\} \rightarrow \{Y\}]_{\theta_3}$ (resp. $[\{C\} \rightarrow \{CY\}]_{\theta_3}$).

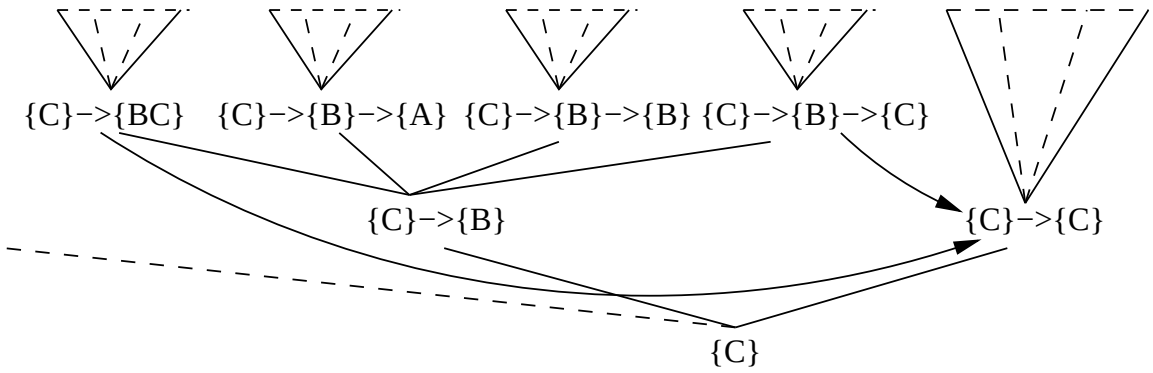


Fig. 6.2 – The partial sub-lattice of class $[C]_{\theta_1}$

A	
SID	Bitmap
1	100
2	01
3	10

B	
SID	Bitmap
1	101
2	11
3	10

C	
SID	Bitmap
1	011
2	01
3	01

Tab. 6.3 – The semi-vertical dataset for table 6.1

6.3.4.3 Data Representation

An efficient data representation is crucial for sequence mining algorithms since most of the running time is spent in counting sequence supports. Thus, bitSPADE uses a semi-vertical database. The semi-vertical database differs from the vertical database by the fact that for a given item A, we associate a hid-list, denoted by $\mathcal{L}_{\mathcal{H}}(A)$, which is a set of tuples $(SID, hTID)$ where $hTID$ is the set of all transaction ids of sequence SID in which A appears. Moreover $\forall (S_0, hTID_0), (S_1, hTID_1) \in \mathcal{L}_{\mathcal{H}}(A), S_0 = S_1 \Rightarrow hTID_0 = hTID_1$.
With the semi-vertical dataset, the transaction set $hTID$, for a given item A and a sequence S, can be represented by a bitmap B_{hTID} composed of a number of bits equals to the size of S. We define a one-to-one order-preserving function f_b such that for each transaction with id $i \in hTID$, we associate to it the bit numbered $f_b(i)$ of B_{hTID}, denoted by $B_{hTID}[f_b(i)]$. For the bitmap B_{hTID}, the following property holds $\forall f_b(i), B_{hTID}[f_b(i)] = 1$ iff $i \in hTID$. Table 6.3 shows the semi-vertical dataset associated to the dataset of the table 6.1.
Using the bitmap structure, bitSPADE uses the I-Step and the S-Step processes of SPAM [AGYF02] to compute respectively the equality join and the temporal join. The I-Step process can be done with an *and* operation. The S-Step of two sequences α and β is performed, for each different SID, by looking for the first transaction with tid t_0 in which the last item of α appears and dropping all transactions of β with a tid lower or equal to t_0. Assume δ a sequence obtained by an equality join of α with β, $\mathcal{L}_{\mathcal{H}}(\alpha)$ and $\mathcal{L}_{\mathcal{H}}(\beta)$ the respective hid-lists of α and β, the hid-list of δ is then $\mathcal{L}_{\mathcal{H}}(\delta) = \{(s, b) \mid (s, b_\alpha) \in \mathcal{L}_{\mathcal{H}}(\alpha),\ (s, b_\beta) \in \mathcal{L}_{\mathcal{H}}(\beta)\ and\ b = AND(b_\alpha, b_\beta)\ and\ b \neq 0\}$. And if γ is a sequence obtained by a temporal join of α with β then the hid-list of γ is $\mathcal{L}_{\mathcal{H}}(\gamma) = \{(s, b) \mid (s, b_\alpha) \in \mathcal{L}_{\mathcal{H}}(\alpha),\ (s, b_\beta) \in \mathcal{L}_{\mathcal{H}}(\beta)\ and\ b = AND(\mathcal{M}(b_\alpha), b_\beta)\ and\ b \neq 0\}$ where $\mathcal{M}(b_\alpha)$ is a bitmap in which all the bits with indexes superior to the index of the first non-zero bit of b_α are set to one and the rest are set to zero.

6.3.5 Experimental Results

In this section, we present the results of our experiments on the performance of bitSPADE, SPADE [Zak98] and SPAM [AGYF02]. The source code of SPADE was obtained from M. J. Zaki [4] and the source code of SPAM from J. Gehrke [5]. All the experiments were performed on a 3GHz Intel Pentium 4 PC machine with one gigabyte of RAM and using the Debian-

[4] http ://www.cs.rpi.edu/∼zaki/software/
[5] http ://himalaya-tools.sourceforge.net/Spam/

Symbol	Meaning	Value
D	Number of customers(i.e. sequences) in the dataset (unit :thousand)	15
C	Average number of transactions per customer	15
T	Average number of items per transaction	15
S	Average length of maximal sequences	15
I	Average length of maximal transactions	15
N	Number of different items	100

Tab. 6.4 – Parameters used to generate the synthetic dataset.

Linux 2.6.14 operating system. All of the three programs were compiled with g++ 3.3 with the optimization flag set to -O3. The execution time of the programs was measured using the *time* shell command. The peak memory usage was measured with the *memusage* program. During the experiments, the output of frequent patterns was turned off.

All the experiments were performed on a synthetic dataset generated with the IBM AssocGen program[6] [AS95]. The synthetic datasets were widely used in the domains of frequent sequence and item mining [AS95, Zak98, AGYF02]. Therefore, they became suitable for algorithms comparison. The parameters used to generate the dataset are summarized in table 6.4. The parameters were selected according to the literature. Nevertheless, we ran the experiments, including data generation with different parameter settings, several times in order to ensure that our results were not due to a particular dataset.

Figures 6.3 to 6.8 shows that, in terms of speed, SPAM is still the fastest algorithm outperforming bitSPADE by an average factor of 2. Nevertheless bitSPADE outperforms SPADE by an average factor of 3.7. The execution speed of SPAM can be explained by the usage of the efficient vertical bitmap representation of the dataset which allows fast sequence joining operations. Since bitSPADE uses the same joining operation but with a semi-vertical dataset, it outperforms SPADE. In terms of memory usage, bitSPADE is the most efficient algorithm. It outperforms SPAM by a factor up to more than an order of magnitude and SPADE by a factor of 3.4.

To study the scale-up of bitSPADE, we performed a number of experiments by varying, independently, each parameters shown in table 6.4. The results are shown in figures 6.4 to 6.8. Figure 6.4 shows that bitSPADE scales linearly as the number of customers in the dataset increase. Moreover, independently of the parameters, bitSPADE's memory usage always scales almost linearly. In terms of speed, results show that bitSPADE has quite the same scale-up as

[6] http ://www.almaden.ibm.com/software/projects/hdb /Resources/datasets/syndata.html

SPAM.

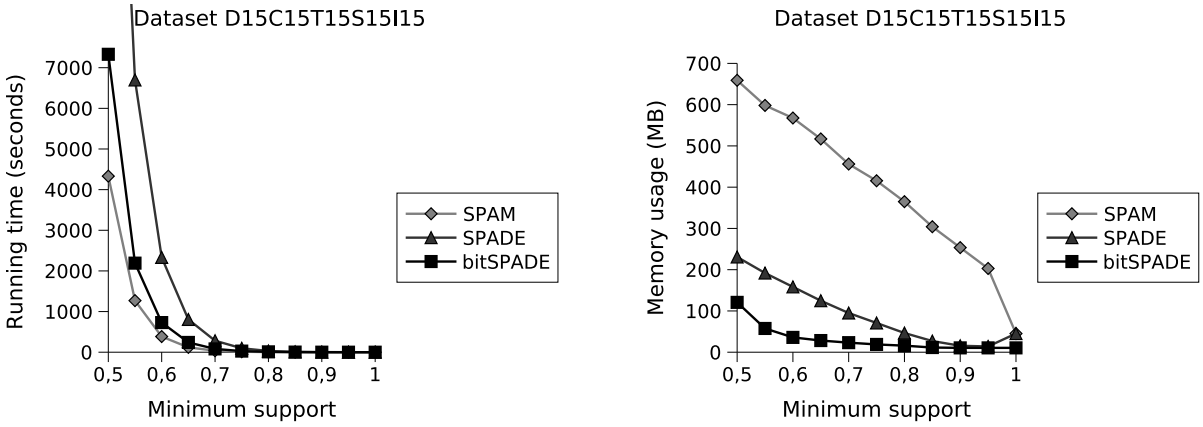


Fig. 6.3 – Performance Comparison : varying minimal support

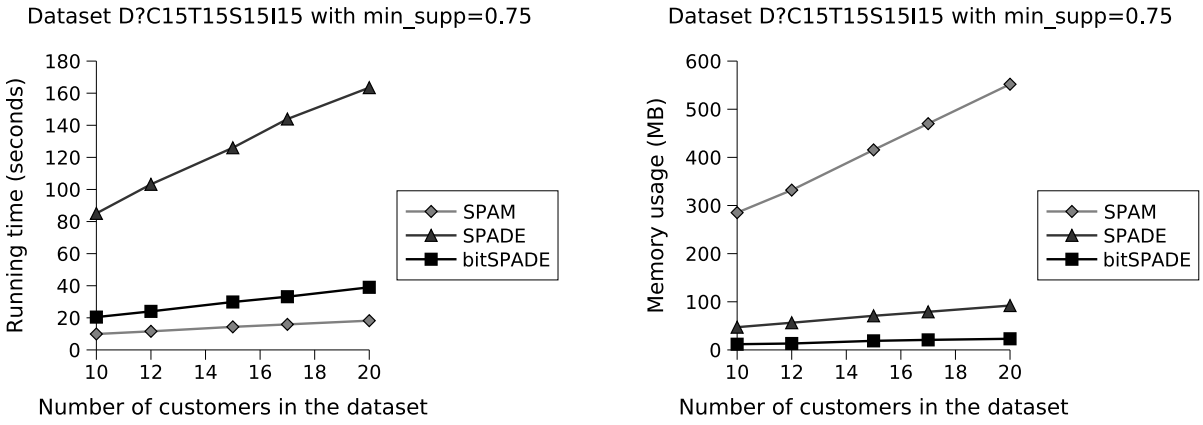


Fig. 6.4 – Scale-up Comparison : varying number of customers

6.4 Conclusion

Dans ce chapitre, nous avons présenté bitSPADE un algorithme performant pour l'extraction de motifs séquentiels. Cet algorithme résulte de la fusion de l'algorithme SPADE, connu pour ses performances en matière d'utilisation de mémoire lors de la phase d'énumération de séquences, et de l'algorithme SPAM, connu pour ses performances en terme de vitesse d'exécution. Cette fusion permet à bitSPADE d'extraire des motifs à partir de larges bases de données en les décomposant en petites classes d'équivalences. L'extraction s'effectue efficacement en terme de vitesse par l'utilisation d'une représentation binaire semi-verticale de la base de données. De plus, nous avons proposé une nouvelle stratégie d'élagage qui permet de diriger bitSPADE lors du parcours de treillis. Les expériences ont montré que bitSPADE obtient un bon compromis entre le temps de traitement et la consommation de mémoire. En effet, bitSPADE surpasse

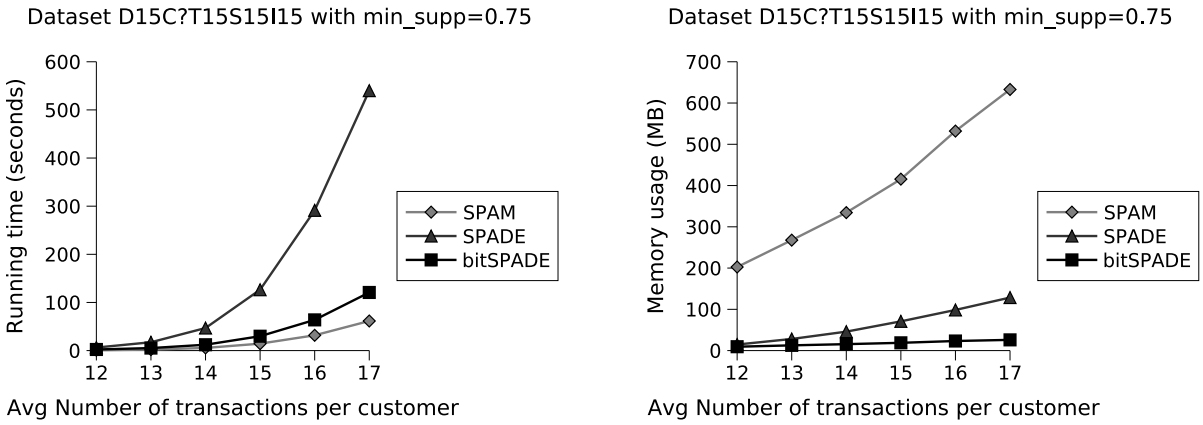


Fig. 6.5 – Scale-up Comparison : varying number of transactions per customer

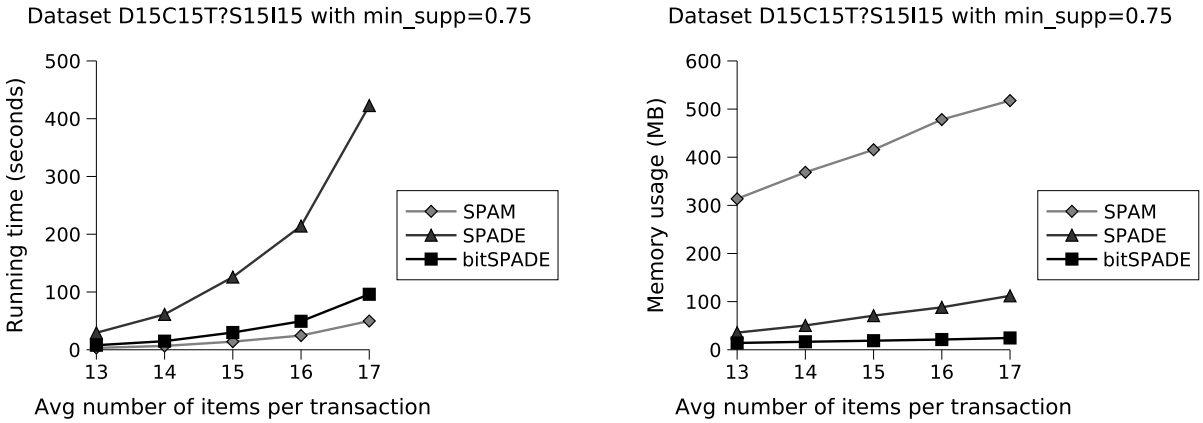


Fig. 6.6 – Scale-up Comparison : varying number of items per transaction

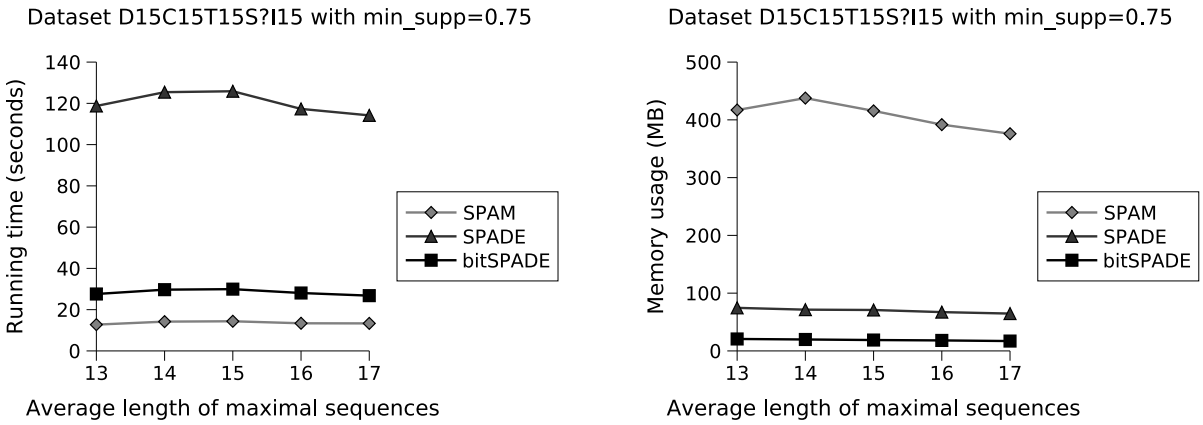


Fig. 6.7 – Scale-up Comparison : varying length of frequent sequences

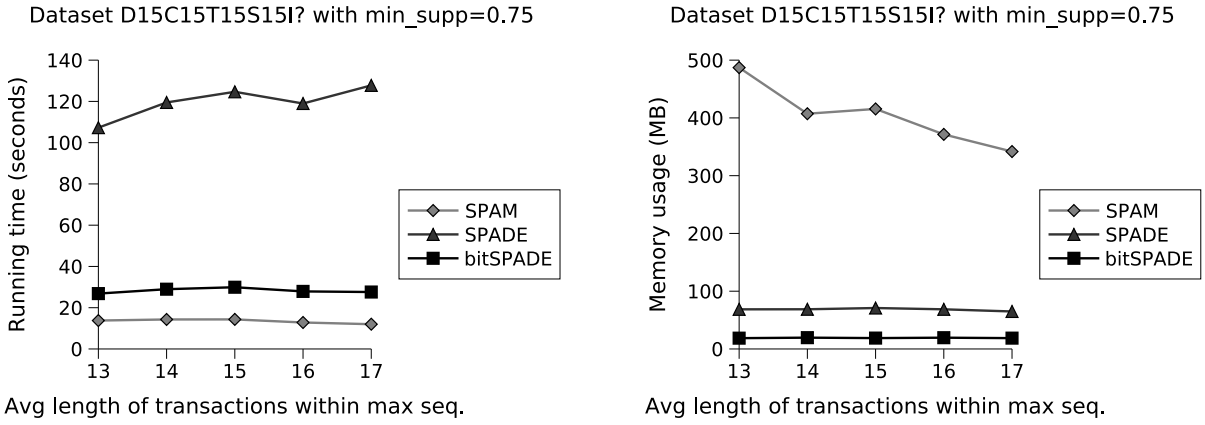


Fig. 6.8 – Scale-up Comparison : varying length of maximal transactions

SPADE et SPAM en terme d'utilisation de mémoire. De plus, bitSPADE obtient des temps d'exécution qui sont largement inférieur à SPADE.

Dans les applications de la vie courante, il est souvent nécessaire d'extraire des motifs séquentiels en prenant en compte des contraintes temporelles telles que des fenêtres temporelles et des sauts temporelles entre les événements d'une séquence. De telles contraintes ont été introduites dans [SA96] mais très peu de travaux ont été réalisés dans ce domaine [SA96, MCP98, Zak00, LLW02, OPS04]. Par conséquent, nous étudions la possibilité d'incorporer ces contraintes dans bitSPADE.

CHAPITRE 7

Conclusion et perspectives

Dans cette thèse, nous avons présenté plusieurs fonctions noyaux pour documents textuels. Tous ces noyaux se basent sur l'information sémantique pour estimer la similarité entre les documents. Dans cette section, nous présentons les principaux apports de cette thèse ainsi que les perspectives de recherche associées.

1. *Noyau sémantique pour documents semi-structurés* : Nous avons proposé un noyau sémantique pour les documents textuels structurés en sections indépendantes. L'objectif du noyau est de tenir compte des parties indépendantes composant un document. Ainsi, lors du calcul de similarité, seules les sections pertinentes sont comparées. En outre, le noyau se base non pas sur les termes composants le document mais sur le sens sémantique du document. Pour cela, une source de connaissance externe est utilisée pour associer chaque groupe de mot du document à un concept sémantique. Le noyau utilise ensuite ces concepts pour calculer une distance sémantique en utilisant une taxonomie de concepts. Le noyau sémantique a été évalué sur un corpus de documents médicaux. Chaque document de ce corpus est un dossier patient, formatté en XML, regroupant deux sections indépendantes. Une des deux sections est une observation clinique du patient et l'autre est une observation radiologique. Le thésaurus médical UMLS a été utilisé pour définir les concepts et pour effectuer l'association avec les mots. L'évaluation s'est faite dans le cadre de la compétition internationale 2007 de catégorisation de documents médicaux organisée par le centre informatique de l'hôpital pour enfant de Cincinnati. Le noyau sémantique a été classé dixième sur 44 méthodes.

   Bien que le noyau sémantique ait obtenu de bonne performance, la modélisation des documents semi-structurés peut être améliorée. En effet, nous pensons, dans un travail futur, organiser les unités lexicales selon des liens syntaxiques. Ainsi, les unités lexicales représentant des adjectifs seront reliées à des noms. Les arcs reliants ces unités lexicales seront pondérées selon une probabilité de liaison afin de gérer l'ambiguïté. Cette probabi-

lité pourra être calculée en fonction de la distance séparant l'adjectif du nom. Nous ferons de même pour les verbes. Cette représentation permettra de comparer les unités lexicales syntaxiquement compatibles. Par conséquent, nous ne comparerons plus, par exemple, un adjectif et un nom. Au contraire, les noms seront comparés entre eux en prenant en compte les adjectifs les qualifiants. Nous pensons qu'une telle représentation évitera de générer des informations non pertinentes.
En outre, notre méthode n'effectue pas de désambiguïsation lors de l'annotation sémantique. Ceci a pour effet de créer du bruit. Ainsi, il peut être nécessaire d'introduire une phase de désambiguïsation. Pour cela, nous pensons que l'utilisation d'une fenêtre de contexte permettra d'améliorer l'annotation. La fenêtre devant capturer les co-occurrences d'un mot et des concepts associés. En fonction des co-occurrences de concepts fréquentes pour un mot, nous pouvons en déduire si un concept est plausible ou non.

2. *Noyaux sémantiques basés sur l'information latente* : Dans une seconde partie, nous nous sommes intéressés aux concepts latents extraits par des méthodes statistiques. L'utilisation des concepts latents dans un algorithme d'apprentissage permet de le rendre autonome, à savoir indépendant de l'extérieur, et de le rendre générique, à savoir non dépendant du corpus. En effet, quelque soit le domaine du corpus, une phase d'extraction des concepts latents pourra être utilisée pour rendre la méthode indépendante du domaine d'application. Ainsi, dans un premier temps, nous avons proposé un espace sémantique de concepts linguistique. Cet espace est défini par des concepts linguistiques provenant d'un thésaurus externe. Puis, nous avons présenté un noyau, pour cet espace, utilisant l'analyse sémantique latente (LSA) pour extraire des concepts abstraits à partir des concepts linguistiques. Nous avons montré expérimentalement que la combinaison des concepts linguistiques et des concepts statistiques permet d'améliorer les performances de catégorisation. Ces expériences ont été menées sur un corpus médical bien connu pour sa difficulté de catégorisation. Comme précédemment, nous avons utilisé la source de connaissances médicales UMLS pour obtenir les concepts linguistiques. Nous pensons que les performances peuvent être améliorées en choisissant une pondération adaptée pour les concepts. En effet, des travaux récents en matière de pondération de termes ont apporté des améliorations intéressantes. Ainsi, une étude empirique sur ces méthodes de pondération appliquées à notre espace de concepts permettrait sans nul doute d'améliorer nos résultats. En outre, aucune étape de désambiguïsation n'a été utilisée lors de l'association entre les concepts linguistiques et les mots. Ainsi, nous avons introduit inévitablement du bruit dans notre espace. Comme dans notre travail précédent, sur le noyau sémantique pour les données semi-structurées, l'introduction d'une étape de désambiguïsation permettrait d'avoir une représentation plus fidèle du sens des documents.

   Dans un deuxième temps, nous nous sommes focalisés sur une méthode d'extraction de concepts purement statistique. Nous avons proposé d'utiliser la LSA localement à chaque catégorie pour en extraire les concepts latents relatifs à ces régions. Ces concepts sont ensuite utilisés pour former un espace sémantique global. L'espace global est constitué,

alors, de sous-espaces sémantiques orthogonaux mis en évidence par les LSA locaux. Par la suite, nous avons présenté un noyau sémantique enrichi pour cet espace global. L'une des particularités de ce noyau est qu'il est capable de mettre l'accent sur certains sous-espaces selon les documents utilisés. Pour cela, une pondération a été utilisée. Les expériences sur les différents corpus de la littérature montrent que le noyau sémantique enrichi améliore les performances de catégorisation par rapport à une méthode LSA globale. Les différences sont notamment visibles lorsque le nombre de concepts latents est faible. Toutefois, lorsque le corpus contient de nombreuses classes peu représentées, le taux d'erreur du noyau enrichi, pour ces classes, augmente. Ce phénomène est perceptible en prenant en compte l'indicateur F1 macro-moyenné. En effet, bien que la micro-F1 reste supérieure à la moyenne, la macro-F1 est relativement faible. Nous pensons que la pondération utilisée ne convient pas à ce type de corpus. En effet, dans ce cas, il est préférable de se baser sur le sous-espace le plus probable en lui accordant un poids plus important. Pour un travail futur, il serait raisonnable d'effectuer une étude comparative des différentes pondérations notamment celles basées sur la probabilité conditionnelle de Bayes et le gain d'information.

Nous avons montré, dans cette thèse à travers nos différentes propositions, que les concepts statistiques pouvaient être une bonne complémentarité, voire même une alternative, aux concepts linguistiques issus de source de connaissances externes. En effet, bien que les concepts statistiques n'aient pas le même pouvoir descriptif que les concepts linguistiques, ils permettent d'atteindre d'excellentes performances de traitement. Le problème principal réside essentiellement dans le fait que le sens sémantique induit par les concepts statistiques sont difficilement interprétables au niveau humain. En effet, un document textuel représenté dans un espace sémantique de concepts latents peut sembler être dénué de sens "humain".

Outre les différentes améliorations que nous avons cité pour nos méthodes, nous pouvons dégager de cette thèse deux points d'extension de nos contributions :

1. le mélange des concepts statistiques et linguistiques a permi d'améliorer les performances de notre deuxième noyau (étudié à la section 5.3). Par conséquent, nous supposons que ces deux types de concepts apportent des informations complémentaires. Ainsi, nous pensons qu'une méthode d'apprentissage semi-supervisé basé sur le co-apprentissage peut améliorer les relations extraites lors de l'apprentissage. En effet, nous pouvons utiliser deux visions d'un même document pour adopter le co-apprentissage. Un système d'apprentissage utilisera les concepts linguistiques et l'autre les concepts statistiques.

2. bien que les concepts statistiques soient prometteurs, ils ont une lacune importante qui est l'absence de lien entre ces concepts. En effet, construire automatiquement une taxonomie voir même une ontologie de concepts permettrait d'améliorer, considérablement, la précision des calculs de similarité. Dans un premier temps, la construction d'une taxonomie

pourrait être envisagée. Pour cela, nous pouvons appliquer successivement des décompositions SVD dans un espace sémantique latent, afin d'obtenir des relations entre les concepts latents. Ces relations pourront permettre d'établir une structure hiérarchique.

Liste des publications

1. S. Aseervatham, E. Viennet and Y. Bennani. Semi-Structured Document Categorization with a Semantic Kernel. Soumis à the International Journal of Computational Intelligence and Applications (IJCIA), World Scientific. 19 p. 2007.
2. S. Aseervatham. A Concept Vector Space Model for Semantic Kernels. Soumis à the International Journal of Artificial Intelligence Tools (IJAIT), World Scientific. 23 p. 2007.
3. S. Aseervatham. Un modèle d'espace vectoriel de concepts pour noyaux sémantiques. In Proceedings of the Extraction et Gestion des Connaissances (EGC) 2008. Accepté, à paraître.
4. S. Aseervatham and E. Viennet. Méthodes à noyaux appliquées aux textes structurés. In Revue des Nouvelles Technologies de l'Information (RNTI), pages 185-205, 2007. Accepté, à paraître.
5. S. Aseervatham, E. Viennet, and Y. Bennani. A Semantic Kernel for Semi-Structured Documents. In Proceedings of the 2007 IEEE International Conference on Data Mining (ICDM), Omaha NE, USA, october 2007.
6. S. Aseervatham, E. Viennet and Y. Bennani. Noyau Sémantique pour les documents Semi-Structurés. Poster présenté au projet Infomagic-Thalès, juillet 2007.
7. S. Aseervatham and E. Viennet. Méthodes à noyaux appliquées aux textes structurés. Rapport d'avancement 1, ST3.12, Projet Infom@gic, Pôle Cap Digital. LIPN. Juillet 2006.
8. S. Aseervatham, A. Osmani, and E. Viennet. BitSPADE : A lattice-based sequential pattern mining algorithm using bitmap representation. In Proceedings of the 2006 IEEE International Conference on Data Mining (ICDM), pages 792-797, Hong-Kong, China, december 2006.
9. S. Aseervatham and A. Osmani. Mining short sequential patterns for hepatitis type detection. In Proceedings of the Discovery Challenge Workshop (ECML/PKDD), Porto, Portugal, october 2005.
10. T. Gaudré, V. Joloboff, Y. Elmrabet, S. Aseervatham, P. Robin, P. Dague and A. Osmani. Réalisation d'un Environnement Système Évolutif pour le Diagnostic Automobile. Projet RESEDA RNTL-2001-2004, LIPN, décembre 2004.

11. Y. Elmrabet, S. Aseervatham, P. Dague and A. Osmani. Réalisation du Système de Diagnostic pour le Système EMB du Vèhicule. Projet RESEDA RNTL-2001-2004, LIPN, décembre 2004.

Bibliographie

[AC00] C. Angulo and A. Català. K-SVCR. A Multi-class Support Vector Machine. In *ECML '00 : Proceedings of the 11th European Conference on Machine Learning*, pages 31–38, London, UK, 2000. Springer-Verlag.

[AG02] M.-R. Amini and P. Gallinari. The use of unlabeled data to improve supervised learning for text summarization. In *SIGIR '02 : Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 105–112, New York, NY, USA, 2002.

[AGYF02] J. Ayres, J. Gehrke, T. Yiu, and J. Flannick. Sequential pattern mining using a bitmap representation. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 429–435, New York, NY, USA, 2002. ACM Press.

[AO05] S. Aseervatham and A. Osmani. Mining short sequential patterns for hepatitis type detection. In *Discovery Challenge (ECML/PKDD)*, 2005.

[AOV06] S. Aseervatham, A. Osmani, and E. Viennet. BitSPADE : A lattice-based sequential pattern mining algorithm using bitmap representation. In *Proceedings of the 2006 IEEE International Conference on Data Mining.*, pages 792–797, dec 2006.

[APC03] C. Angulo, X. Parra, and A. Català. K-SVCR. A support vector machine for multi-class classification. *Neurocomputing*, 55(1-2) :57–77, 2003.

[Aro50] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68 :337–404, 1950.

[AS94] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[AS95] R. Agrawal and R. Srikant. Mining Sequential Patterns. In *Proceedings of the 11th IEEE International Conference on Data Engineering*, pages 3–14, 1995.

[AV06] S. Aseervatham and E. Viennet. Méthodes à Noyau appliquées aux textes structurés (Rapport d'avancement 1, ST3.12, Projet Infom@gic, Pôle Cap Digital). Technical report, LIPN, July 2006.

[AV07] S. Aseervatham and E. Viennet. Méthodes à noyaux appliquées aux textes structurés. *Revue des Nouvelles Technologies de l'Information*, pages 185–205, 2007. Accepté, à paraitre.

[AVB07] S. Aseervatham, E. Viennet, and Y. Bennani. A Semantic Kernel for Semi-Structured Documents. In *Proceedings of the ICDM'07, IEEE International Conference on Data Mining*, Omaha NE, USA, october 2007.

[BB99] E. J. Bredensteiner and K. P. Bennett. Multicategory Classification by Support Vector Machines. *Computational Optimizations and Applications*, 12(1-3) :53–79, 1999.

[BB06] K. Benabdeslem and Y. Bennani. Dendogram-based SVM for multi-class classification. *Journal of Computing and Information Technology*, 14(4) :288–291, 2006.

[BC99] C. Burges and D. Crisp. Uniqueness of the SVM solution. In *Advances in Neural Information Processing Systems*, pages 223–229. The MIT Press, 1999.

[BCM06] R. Basili, M. Cammisa, and A. Moschitti. A Semantic Kernel to classify texts with very few training examples. *Informatica*, 30(2) :163–172, 2006.

[BD98] K. P. Bennett and A. Demiriz. Semi-supervised support vector machines. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 368–374, Cambridge, MA, USA, 1998. MIT Press.

[Ben06] Y. Bennani. Séparateurs connexionnistes linéaires. In Y. Bennani, editor, *Apprentissage Connexionniste*, pages 21–46. Editions Hermès Science, 2006.

[BHHSV01a] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik. A Support Vector Method for Clustering. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems (NIPS 13)*, pages 367–373. MIT Press, 2001.

[BHHSV01b] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. N. Vapnik. Support Vector Clustering. *Journal of Machine Learning Research*, 2 :125–137, 2001.

[Bis95] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[BLT06] F. Badran, M. Lebbah, and S. Thiria. Perceptron Multi-Couches (PMC). In Y. Bennani, editor, *Apprentissage Connexionniste (sous la direction de Y. Bennani)*, pages 47–82. Editions Hermès Science, 2006.

[BM98a] L. Douglas Baker and A. K. McCallum. Distributional clustering of words for text classification. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 96–103, Melbourne, AU, 1998. ACM Press, New York, US.

[BM98b] A. Blum and T. Mitchell. Combining Labeled and Unlabeled Data with Co-training. In *COLT : Proceedings of the Workshop on Computational Learning Theory*. Morgan Kaufmann Publishers, 1998.

[Bor07] K. M. Borgwardt. *Graph Kernels*. PhD thesis, Ludwig-Maximilians-University Munich, 2007.

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW7 : Proceedings of the seventh international conference on World Wide Web 7*, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[BPd+92] P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4) :467–479, 1992.

[BS04] U. Brefeld and T. Scheffer. Co-EM support vector learning. In *ICML '04 : Proceedings of the twenty-first international conference on Machine learning*, page 16, New York, NY, USA, 2004. ACM Press.

[BSA94] C. Buckley, G. Salton, and J. Allan. The effect of adding relevance information in a relevance feedback environment. In *SIGIR '94 : Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 292–300, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[BSAS94] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic Query Expansion Using SMART : TREC 3. In *Proceedings of the third Text REtrieval Conference (TREC 3)*, 1994.

[Bun02] W. L. Buntine. Variational Extensions to EM and Multinomial PCA. In *ECML '02 : Proceedings of the 13th European Conference on Machine Learning*, pages 23–34, London, UK, 2002. Springer-Verlag.

[Bur98] C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2) :121–167, 1998.

[CD02] M. Collins and N. Duffy. Convolution Kernels for Natural Language. In *NIPS : Advances in Neural Information Processing Systems 14*, pages 625–632. MIT Press, 2002.

[CG92] G. Celeux and G. Govaert. A Classification EM algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, 14(3) :315–332, 1992.

[CILR06] M. Cottrell, S. Ibbou, P. Letremy, and P. Rousset. Cartes auto-organisatrices de Kohonen. In Y. Bennani, editor, *Apprentissage Connexionniste*, pages 141–183. Editions Hermès Science, 2006.

[CL01] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM : a library for support vector machines*, 2001.

[CMC+06] Fabrizio Costa, Sauro Menchetti, Alessio Ceroni, Andrea Passerini, and Paolo Frasconi. Decomposition Kernels for Natural Language Processing. In *Learning Structured Information in Natural Language Applications Workshop, Trento,*

Italy, April 3rd 2006, hosted in conjunction with the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006), April 5–7, 2006, 2006.

[CMR03] S. Canu, X. Mary, and A. Rakotomamonjy. Functional learning through kernel. *Advances in Learning Theory : Methods, Models and Applications, NATO Science Series III : Computer and Systems Sciences*, 190 :89–110, 2003.

[Col02] Michael Collins. Discriminative Training Methods for Hidden Markov Models : Theory and Experiments with Perceptron Algorithms. In *Proceedings of EMNLP*, 2002.

[CSTL02] N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent Semantic Kernels. *Journal of Intelligent Information Systems*, 18(2-3) :127–152, 2002.

[CV95] C. Cortes and V. N. Vapnik. Support-Vector Networks. *Machine Learning*, 20(3) :273–297, 1995.

[DBK+97] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. N. Vapnik. Support Vector Regression Machines. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 155. The MIT Press, 1997.

[DBL06] G. Divita, A.C. Browne, and R. Loanne. dTagger : A POS Tagger. In *Proceedings AMIA Fall Symposium*, pages 201–203, 2006.

[DDL+90] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6) :391–407, 1990.

[DGLT03] F. Denis, R. Gilleron, A. Laurent, and M. Tommasi. Text Classification and Co-training from Positive and Unlabeled examples. In *Proceedings of the ICML Workshop : the Continuum from Labeled Data to Unlabeled Data in Machine Learning and Data Mining*, pages 80 – 87, 2003.

[DIR03] P. Dai, U. Iurgel, and G. Rigoll. A Novel Feature Combination Approach for Spoken Document Classification with Support Vector Machines. In *Proceedings of the ACM SIGIR Workshop on Multimedia Information Retrieval*, 2003.

[DS03] F. Debole and F. Sebastiani. Supervised term weighting for automated text categorization. In *SAC '03 : Proceedings of the 2003 ACM symposium on Applied computing*, pages 784–788, New York, NY, USA, 2003. ACM Press.

[Fel98] C. Fellbaum. *WordNet : An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998.

[Fle87] R. Fletcher. *Practical Methods of Optimization*, chapter 8.7 : Polynomial time algorithms, pages 183–188. John Wiley & Sons, New York, second edition, 1987.

[Gär03] T. Gärtner. A survey of kernels for structured data. *SIGKDD Explor. Newsl.*, 5(1) :49–58, 2003.

[GDR03] T. Gärtner, K. Driessens, and J. Ramon. Graph Kernels and Gaussian Processes for Relational Reinforcement Learning. In *ILP*, pages 146–163, 2003.

[GEPM00] Y. Guermeur, A. Elisseeff, and H. Paugam-Moisy. A New Multi-Class SVM Based on a Uniform Convergence Result. In *IJCNN '00 : Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)-Volume 4*, pages 183–188, Washington, DC, USA, 2000. IEEE Computer Society.

[GG05] E. Gaussier and C. Goutte. Relation between PLSA and NMF and implications. In *SIGIR '05 : Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 601–602, New York, NY, USA, 2005. ACM Press.

[GLS06] T. Gärtner, Q. V. Le, and A. J. Smola. A Short Tour of Kernel Methods for Graphs, 2006.

[GS05] A. Gliozzo and C. Strapparava. Domain Kernels for Text Categorization. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 56–63, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[Haf05] G. Haffari. A Survey on Inductive Semi-Supervised Learning. Technical report, School of Computing Science, Simon Fraser University, 2005.

[Hau99] D. Haussler. Convolution Kernels on Discrete Structures. Technical Report UCS-CRL-99-10, UC Santa Cruz, 1999.

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

[HMB07] S. Hassan, R. Mihalcea, and C. Banea. Random-Walk Term Weighting for Improved Text Classification. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2007)*. IEEE Computer Society, 2007.

[Hof99a] T. Hofmann. Probabilistic Latent Semantic Analysis. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 289–296, San Francisco, CA, 1999. Morgan Kaufmann.

[Hof99b] T. Hofmann. Probabilistic Latent Semantic Indexing. In *SIGIR '99 : Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57, New York, NY, USA, 1999. ACM Press.

[Hof00] T. Hofmann. Learning the Similarity of Documents : An Information-Geometric Approach to Document Retrieval and Categorization. In *Advances in Neural Information Processing Systems (NIPS'12)*, pages 914–920. MIT Press, 2000.

[Hof01] T. Hofmann. Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 42(1/2) :177–196, 2001.

[HP98] T. Hofmann and J. Puzicha. Unsupervised Learning from Dyadic Data. Technical Report TR-98-042, International Computer Science Insitute, Berkeley, CA, 1998.

[HPJ99] T. Hofmann, J. Puzicha, and M. I. Jordan. Learning from Dyadic Data. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 466–472, Cambridge, MA, USA, 1999. MIT Press.

[HPMA+00] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu. Freespan : frequent pattern-projected sequential pattern mining. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 355–359, New York, NY, USA, 2000. ACM Press.

[HPY05] J. Han, J. Pei, and X. Yan. Sequential Pattern Mining by Pattern-Growth : Principles and Extensions. In W. Chu and T. Lin, editors, *Foundations and Advances in Data Mining*, volume 180, pages 183–220. Springer Verlag, studies in fuzziness and soft computing edition, 2005.

[Hul94] D. Hull. Improving Text Retrieval for the Routing Problem using Latent Semantic Indexing. In *SIGIR '94 : Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 282–291, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[Hul95] D. A. Hull. *Information Retrieval using Statistical Classification*. PhD thesis, Stanford University, 1995.

[IK02] H. Isozaki and H. Kazawa. Efficient Support Vector Classifiers for Named Entity Recognition. In *COLING*, 2002.

[JC97] J. J. Jiang and D. W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th International Conference on Research in Computational Linguisics*, pages 19–33, Taipei, Taiwan, 1997.

[Joa97] T. Joachims. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In *ICML '97 : Proceedings of the Fourteenth International Conference on Machine Learning*, pages 143–151, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[Joa98] Th. Joachims. Text categorization with support vector machines : learning with many relevant features. In *Proc. of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Heidelberg, DE, 1998. Springer Verlag.

[Joa99a] T. Joachims. Transductive Inference for Text Classification using Support Vector Machines. In *ICML '99 : Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[Joa99b] Th. Joachims. Transductive Inference for Text Classification using Support Vector Machines. In I. Bratko and S. Dzeroski, editors, *Proceedings of ICML-99, 16th International Conference on Machine Learning*, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers, San Francisco, US.

[Joa02] T. Joachims. *Learning to Classify Text Using Support Vector Machines : Methods, Theory and Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[Jon88] K. Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Document retrieval systems*, pages 132–142, 1988.

[KI02] H. Kashima and A. Inokuchi. Kernels for Graph Classification. In *ICDM '02 : Proceedings of the First International Conference On Data Mining, Workshop on Active Mining*, 2002.

[KK02] H. Kashima and T. Koyanagi. Kernels for Semi-Structured Data. In *ICML '02 : Proceedings of the Nineteenth International Conference on Machine Learning*, pages 291–298, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[KM03] T. Kudo and Y. Matsumoto. Fast methods for kernel-based text analysis. In *ACL '03 : Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 24–31, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[Koh97] T. Kohonen, editor. *Self-organizing maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997.

[KT04] H. Kashima and Y. Tsuboi. Kernel-based discriminative learning algorithms for labeling sequences, trees, and graphs. In *ICML '04 : Proceedings of the twenty-first international conference on Machine learning*, page 58, New York, NY, USA, 2004. ACM Press.

[KTI03] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized Kernels between Labeled Graphs. In T. Faucett and N. Mishra, editors, *Proceedings of the 20th International Conference on Machine Learning*, pages 321–328. AAAI Press, Aug 2003.

[LC98] C. Leacock and M. Chodorow. Combining local context and WordNet similarity for word sense identification. *WordNet : An Electronic Lexical Database*, pages 265–283, 1998.

[LCZ+04] T. Liu, Z. Chen, B. Zhang, W. Ma, and G. Wu. Improving Text Classification using Local Latent Semantic Indexing. In *ICDM '04 : Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 162–169, Washington, DC, USA, 2004. IEEE Computer Society.

[Lee05] D. Lee. An Improved Cluster Labeling Method for Support Vector Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3) :461–464, 2005.

[LEN02] C. S. Leslie, E. Eskin, and W. S. Noble. The Spectrum Kernel : A String Kernel for SVM Protein Classification. In *Pacific Symposium on Biocomputing*, pages 566–575, 2002.

[Lin98] D. Lin. An information-theoretic definition of similarity. In *Proc. of the 15th International Conf. on Machine Learning (ECML)*, pages 296–304. Morgan Kaufmann, San Francisco, CA, 1998.

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

[LLW02] M.-Y. Lin, S.-Y. Lee, and S.-S. Wang. DELISP : Efficient Discovery of Generalized Sequential Patterns by Delimited Pattern-Growth Technology. In *Proceedings of the Sixth Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD)*, pages 198–209, London, UK, 2002. Springer-Verlag.

[LLW03] H.-T. Lin, C.-J. Lin, and R. C. Weng. A Note on Platt's Probabilistic Outputs for Support Vector Machines. Technical report, National Taiwan University, May 2003.

[LRBE03] M. Leleu, C. Rigotti, J.-F. Boulicaut, and G. Euvrard. GO-SPADE : Mining Sequential Patterns over Datasets with Consecutive Repetitions. In *Proceedings of the Third IAPR International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*, pages 293–306. Springer, 2003.

[LSST+02] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. J. C. H. Watkins. Text Classification using String Kernels. *Journal of Machine Learning Research*, 2 :419–444, 2002.

[LTL06] M. Lan, C. L. Tan, and H. B. Low. Proposing a new term weighting scheme for text categorization. In *AAAI'06 : Proceedings of the 21st National Conference on Artificial Intelligence*, 2006.

[LTSL07] M. Lan, C. L. Tan, J. Su, and H. B. Low. Text representations for text categorization : a case study in biomedical domain. In *IJCNN'07 : International Joint Conference on Neural Networks*, 2007.

[Mac67] J. B. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, 1967.

[MCP98] F. Masseglia, F. Cathala, and P. Poncelet. The PSP Approach for Mining Sequential Patterns. In *Proceedings of the Second European Conference on Principles of Data Mining and Knowledge Discovery in Databases (PKDD'98)*, pages 176–184. Springer Verlag, 1998.

[MK97] G. J. McLachlan and T. Krishan. *The EM Algorithm and Extension*. Wiley Sons, NY, 1997.

[MN98] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[NG00] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *CIKM '00 : Proceedings of the ninth international conference on Information and knowledge management*, pages 86–93, New York, NY, USA, 2000. ACM Press.

[NMM06] K. Nigam, A. McCallum, and T. Mitchell. Semi-Supervised Text Classification Using EM. In O. Chapelle, A. Zien, and B. Schölkopf, editors, *Semi-Supervised Learning*. MIT Press : Boston, 2006.

[NMTM00] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 39(2-3) :103–134, 2000.

[OPS04] S. Orlando, R. Perego, and C. Silvestri. A new algorithm for gap constrained sequence mining. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 540–547, New York, NY, USA, 2004. ACM Press.

[Par62] E. Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33 :1065–1076, 1962.

[PBM+07] J.P. Pestian, C. Brew, P. Matykiewicz, D.J. Hovermale, N. Johnson, K. Bretonnel Cohen, and W. Duch. A Shared Task Involving Multi-label Classification of Clinical Free Text. In ACL, editor, *Proceedings of ACL BioNLP*, Prague, June 2007. Association of Computational Linguistics.

[PCST00] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large Margin DAGS for Multiclass Classification. In S.A. Solla, T.K. Leen, and K.-R. Muller, editors, *Advances in Neural Information Processing Systems*. MIT Press, 2000.

[PHMA+01] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan : Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. In *Proceedings of the Seventeenth ICDE International Conference on Data Engineering*, pages 215–226, Washington, DC, USA, 2001. IEEE Computer Society.

[Pla99] J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In A.J. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

[Poi99a] T. Poibeau. Le repérage des entités nommées, un enjeu pour les systèmes de veille. In *Terminologies Nouvelles, actes du colloque Terminologie et Intelligence Artificielle (TIA'99)*, volume 2, pages 43–51, 1999.

[Poi99b] T. Poibeau. L'évaluation des systèmes d'extraction d'information : une expérience sur le français. In *Langues*, volume 2, pages 110–118, 1999.

[Por80] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3) :130–137, July 1980.

[PPPC06] T. Pedersen, S. Pakhomov, S. Patwardhan, and C. Chute. Measures of semantic similarity and relatedness in the biomedical domain. *Journal of Biomedical Informatics*, In Press, Corrected Proof, June 2006.

[Rak03] A. Rakotomamonjy. Variable selection using SVM based criteria. *Journal of Machine Learning Research*, 3 :1357–1370, 2003.

[Rak07] A. Rakotomamonjy. Analysis of SVM regression bound for variable ranking. *Neurocomputing*, 70 :1489–1491, 2007.

[RC05] A. Rakotomamonjy and S. Canu. Frame, Reproducing Kernel, Regularization and Learning. *Journal of Machine Learning Research*, pages 1485–1515, 2005.

[Res95] P. Resnik. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proc. of the 14th International Joint Conf. on Artificial Intelligence (IJCAI)*, pages 448–453, 1995.

[Roc71] J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System : Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall, 1971.

[SA96] R. Srikant and R. Agrawal. Mining Sequential Patterns : Generalizations and Performance Improvements. In *EDBT*, pages 3–17, 1996.

[SB87] G. Salton and C. Buckley. Term Weighting Approaches in Automatic Text Retrieval. Technical report, Cornell University, Ithaca, NY, USA, 1987.

[SBV95] B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, Menlo Park, CA, 1995. AAAI Press.

[SdB00] G. Siolas and F. d'Alché Buc. Support Vector Machines Based on a Semantic Kernel for Text Categorization. In *Proc. of IJCNN*. IEEE Computer Society, 2000.

[Seb99] F. Sebastiani. A Tutorial on Automated Text Categorisation. In *Proceedings of the 1st Argentinian Symposium on Artificial Intelligence (ASAI'99)*, pages 7–35, Buenos Aires, AR, 1999.

[Seb02] F. Sebastiani. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, 34(1) :1–47, 2002.

[See00] M. Seeger. Learning with Labeled and Unlabeled Data. Technical report, Institute for ANC, Edinburgh, UK, 2000.

[SHP95] H. Schütze, D. A. Hull, and J. O. Pedersen. A Comparison of Classifiers and Document Representations for the Routing Problem. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*, pages 229–237, Seattle, Washington, USA, 1995. ACM Press.

[SHSM03] J. Suzuki, T. Hirao, Y. Sasaki, and E. Maeda. Hierarchical directed acyclic graph kernel : methods for structured natural language data. In *ACL '03 : Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 32–39, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[SLL04] T. Solorio and A. López-López. Learning Named Entity Classifiers Using Support Vector Machines. In *CICLing*, pages 158–167, 2004.

[SM05] P. Soucy and G. W. Mineau. Beyond TFIDF Weighting for Text Categorization in the Vector Space Model. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI'05 : International Joint Conf. on Artificial Intelligence*, pages 1130–1135. Professional Book Center, 2005.

[Sol04] T. Solorio. Improvement of Named Entity Tagging by Machine Learning. Technical Report CCC-04-004, Coordinación de Ciencias Computacionales INAOE, 2004.

[SPST+99] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the Support of a High-Dimensional Distribution. Technical report, Microsoft Research, 1999.

[SPST+01] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7) :1443–1471, 2001.

[SSM03] J. Suzuki, Y. Sasaki, and E. Maeda. Kernels for Structured Natural Language Data. In *NIPS : Advances in Neural Information Processing Systems 16*. MIT Press, 2003.

[SSWB00] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New Support Vector Algorithms. *Neural Computation*, 12(5) :1207–1245, 2000.

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

[SWY75] G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for automatic indexing. *Communications of the ACM*, 18(11) :613–620, 1975.

[TC02] K. Takeuchi and N. Collier. Use of support vector machines in extended named entity. In *Proceedings of CoNLL-2002*, pages 119–125. Taipei, Taiwan, 2002.

[TD99a] D. M. J. Tax and R. P. W. Duin. Data Domain Description by Support Vectors. In *Proceedings of European Symp. Artificial Neural Networks (ESANN)*, pages 251–256, 1999.

[TD99b] D. M. J. Tax and R. P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13) :1191–1199, 1999.

[TD04] D. M. J. Tax and R. P. W. Duin. Support Vector Data Description. *Machine Learning*, 54(1) :45–66, 2004.

[TKR+02] K. Tsuda, M. Kawanabe, G. Rätsch, S. Sonnenburg, and K.-R. Müller. A new discriminative kernel from probabilistic models. *Neural Computation*, 14(10) :2397–2414, 2002.

[TMY04] H. Takamura, Y. Matsumoto, and H. Yamada. Modeling Category Structures with a Kernel Function. In *Proceedings of CoNLL-2004*, pages 57–64. Boston, MA, USA, 2004.

[Vap82] V. N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, New York, USA, 1982.

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

[VGS96] V. N. Vapnik, S. E. Golowich, and A. J. Smola. Support Vector Method for Function Approximation, Regression Estimation and Signal Processing. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, pages 281–287. The MIT Press, 1996.

[WPW95] E. Wiener, J. O. Pedersen, and A. S. Weigend. A Neural Network Approach to Topic Spotting. In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95) (pp. 317-332)*, pages 317–332, Las Vegas, NV, US, 1995.

[WW98] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, Egham, TW20 0EX, UK, 1998.

[WZW85] S. K. M. Wong, W. Ziarko, and P. C. N. Wong. Generalized Vector Spaces Model in Information Retrieval. In *SIGIR '85 : Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 18–25, New York, NY, USA, 1985. ACM Press.

[Yan99] Y. Yang. An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval*, 1(1-2) :69–90, 1999.

[Yan01] Y. Yang. A Study on Thresholding Strategies for Text Categorization. In *Proc. of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 137–145. ACM Press, New York, US, 2001.

[YECC02] J. Yang, V. Estivill-Castro, and S. K. Chalup. Support vector clustering through proximity graph modelling. In *Proceedings of the 9th International Conference on Neural Information Processing (ICONIP '02)*, volume 2, pages 898–903, 2002.

[YP97] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 412–420, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.

[Zak98] M. J. Zaki. Efficient Enumeration of Frequent Sequences. In *CIKM '98 : Proceedings of the seventh international conference on Information and knowledge management*, pages 68–75, 1998.

[Zak00] M. J. Zaki. Sequence Mining In Categorical Domains : Incorporating Constraints. In *Proceedings of the Ninth International Conference on Information and Knowledge Management*, pages 422–429, New York, NY, USA, 2000. ACM Press.

[ZF06] P. Zhong and M. Fukushima. A new multi-class support vector algorithm. *Optimization Methods and Software*, 21(3) :359–372, June 2006.

[Zhu05] X. Zhu. Semi-Supervised Learning Literature Survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[ZM98] J. Zobel and A. Moffat. Exploring the similarity space. *SIGIR Forum*, 32(1) :18–34, 1998.

[ZPZ04] L. Zhang, Y. Pan, and T. Zhang. Focused named entity recognition using machine learning. In *SIGIR '04 : Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 281–288, New York, NY, USA, 2004. ACM Press.