Analyse des propriétés stationnaires et des propriétés émergentes dans les flux d'informations changeant au cours du temps

Randa Kassab

Département de formation doctorale en informatique **École doctorale IAEM Lorraine**
UFR STMIA

Analyse des propriétés stationnaires et des propriétés émergentes dans les flux d'informations changeant au cours du temps

THÈSE

présentée et soutenue publiquement le 11 mai 2009

pour l'obtention du

Doctorat de l'université Henri Poincaré – Nancy 1

(spécialité informatique)

par

Randa Kassab

Composition du jury

Président :	Anne Boyer	Professeur, Université Nancy 2
Rapporteurs :	Younès Bennani	Professeur, Université Paris 13
	Patrick Gallinari	Professeur, Université Paris 6
Examinateurs :	Nacer Boudjlida	Professeur, Université Nancy 1
	Éric Gaussier	Professeur, Université Joseph Fourier
Directeur :	Frédéric Alexandre	Directeur de recherche, INRIA

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thloria.

Remerciements

En premier lieu, je tiens à remercier mon directeur de thèse, Frédéric Alexandre, pour son aide et tous ses conseils, son écoute, sa patience et sa disponibilité. Je le remercie particulièrement pour la liberté qu'il m'a laissée pour mener à terme ce travail, le soutien qu'il m'a apporté dans les moments les plus difficiles, et le temps qu'il a consacré à lire et à corriger ce manuscrit.

Mes remerciements s'adressent également aux membres du jury de thèse pour le temps qu'ils ont consacré à l'évaluation de ce travail. Je remercie Anne Boyer d'avoir bien voulu présider ce jury et d'avoir toujours montré son soutien et son intérêt à mon travail. Je tiens à remercier tout particulièrement Younès Bennani et Patrick Gallinari, d'avoir accepté la charge d'être les rapporteurs de ce travail de thèse, malgré toutes les responsabilités qu'ils assument. Leurs commentaires et suggestions me furent très précieux. De même, je tiens à remercier Nacer Boudjlida et Éric Gaussier, pour leur intervention en tant qu'examinateurs et pour l'intérêt qu'ils ont manifesté à l'égard de mon travail.

Je tiens ensuite à remercier l'ensemble des membres actuels et anciens de l'équipe CORTEX pour leur gentillesse, leur aide et leur soutien durant toutes ces années. De même, je tiens à remercier tous les partenaires du projet Sat-N-Surf qu'ils soient ingénieurs, étudiants, ou chercheurs, pour leur collaboration et les nombreuses discussions.

Je voudrais aussi remercier tous mes professeurs de l'université de Damas qui ont contribué, peut-être de manière indirecte mais importante, à la réalisation de cette thèse. Je pense, particulièrement, à mon superviseur Faysal Al-Abbas. Je le remercie pour ses conseils, ses encouragements et son aide précieuse sur le plan administratif.

Mes pensés vont également à ceux de mes proches, ma famille, et mes amis qui ont toujours été là pour moi et qui savent combien ils comptent pour moi. Je destine un remerciement tout spécial à mes parents à qui je dois beaucoup de ce que je suis aujourd'hui.. et je leur dédie cette thèse.

Résumé

De nombreuses applications génèrent et reçoivent des données sous la forme de flux continu, illimité, et très rapide. Cela pose naturellement des problèmes de stockage, de traitement et d'analyse de données qui commencent juste à être abordés dans le domaine des flux de données. Il s'agit, d'une part, de pouvoir traiter de tels flux à la volée sans devoir mémoriser la totalité des données et, d'autre part, de pouvoir traiter de manière simultanée et concurrente l'analyse des régularités inhérentes au flux de données et celle des nouveautés, exceptions, ou changements survenant dans ce même flux au cours du temps.

L'apport de ce travail de thèse réside principalement dans le développement d'un modèle d'apprentissage — nommé ILoNDF — fondé sur le principe de la détection de nouveauté. L'apprentissage de ce modèle est, contrairement à sa version de départ, guidé non seulement par la nouveauté qu'apporte une donnée d'entrée mais également par la donnée elle-même. De ce fait, le modèle ILoNDF peut acquérir constamment de nouvelles connaissances relatives aux fréquences d'occurrence des données et de leurs variables, ce qui le rend moins sensible au bruit. De plus, doté d'un fonctionnement en ligne sans répétition d'apprentissage, ce modèle répond aux exigences les plus fortes liées au traitement des flux de données.

Dans un premier temps, notre travail se focalise sur l'étude du comportement du modèle ILoNDF dans le cadre général de la classification à partir d'une seule classe en partant de l'exploitation des données fortement multidimensionnelles et bruitées. Ce type d'étude nous a permis de mettre en évidence les capacités d'apprentissage pures du modèle ILoNDF vis-à-vis de l'ensemble des méthodes proposées jusqu'à présent. Dans un deuxième temps, nous nous intéressons plus particulièrement à l'adaptation fine du modèle au cadre précis du filtrage d'informations. Notre objectif est de mettre en place une stratégie de filtrage orientée-utilisateur plutôt qu'orientée-système, et ceci notamment en suivant deux types de directions. La première direction concerne la modélisation utilisateur à l'aide du modèle ILoNDF. Cette modélisation fournit une nouvelle manière de regarder le profil utilisateur en termes de critères de spécificité, d'exhaustivité et de contradiction. Ceci permet, entre autres, d'optimiser le seuil de filtrage en tenant compte de l'importance que pourrait donner l'utilisateur à la précision et au rappel. La seconde direction, complémentaire de la première, concerne le raffinement des fonctionnalités du modèle ILoNDF en le dotant d'une capacité à s'adapter à la dérive du besoin de l'utilisateur au cours du temps. Enfin, nous nous attachons à la généralisation de notre travail antérieur au cas où les données arrivant en flux peuvent être réparties en classes multiples.

Mots-clés: apprentissage automatique, réseaux de neurones, classification supervisée et non supervisée, détection de nouveauté, flux de données, dérive de concept, filtrage basé sur le contenu, modélisation utilisateur, analyse des données multidimensionnelles, applications en Intelligence Artificielle

Abstract

Many applications produce and receive continuous, unlimited, and high-speed data streams. This raises obvious problems of storage, treatment and analysis of data, which are only just beginning to be treated in the domain of data streams. On the one hand, it is a question of treating data streams on the fly without having to memorize all the data. On the other hand, it is also a question of analyzing, in a simultaneous and concurrent manner, the regularities inherent in the data stream as well as the novelties, exceptions, or changes occurring in this stream over time.

The main contribution of this thesis concerns the development of a new machine learning approach — called ILoNDF — which is based on novelty detection principle. The learning of this model is, contrary to that of its former self, driven not only by the novelty part in the input data but also by the data itself. Thereby, ILoNDF can continuously extract new knowledge relating to the relative frequencies of the data and their variables. This makes it more robust against noise. Being operated in an on-line mode without repeated training, ILoNDF can further address the primary challenges for managing data streams.

Firstly, we focus on the study of ILoNDF's behavior for one-class classification when dealing with high-dimensional noisy data. This study enabled us to highlight the pure learning capacities of ILoNDF with respect to the key classification methods suggested until now. Next, we are particularly involved in the adaptation of ILoNDF to the specific context of information filtering. Our goal is to set up user-oriented filtering strategies rather than system-oriented in following two types of directions. The first direction concerns user modeling relying on the model ILoNDF. This provides a new way of looking at user's need in terms of specificity, exhaustivity and contradictory profile-contributing criteria. These criteria go on to estimate the relative importance the user might attach to precision and recall. The filtering threshold can then be adjusted taking into account this knowledge about user's need. The second direction, complementary to the first one, concerns the refinement of ILoNDF's functionality in order to confer it the capacity of tracking drifting user's need over time. Finally, we consider the generalization of our previous work to the case where streaming data can be divided into multiple classes.

Keywords: machine learning, neural networks, supervised and unsupervised classification, novelty detection, data streams, concept drift, content-based filtering, user modeling, multidimensional data analysis, Artificial Intelligence applications

Table des matières

Introduction

Chapitre 1
Définitions et éléments clés d'analyse de données

1.1 Qu'est qu'un flux de données ? 5
1.2 Représentation du temps dans un flux de données 7
1.2.1 Types de datation 7
1.2.2 Notion de fenêtrage 8
1.3 Modèles de flux de données : cas spéciaux 9
1.4 Généralités sur l'analyse de données multidimensionnelles 11
1.4.1 Les données et types de variables 11
1.4.2 Analyse statistique de données 12
1.4.3 Apprentissage 13
1.4.4 Clustering 18
1.4.5 Classification 20
1.4.6 Synthèse 23
1.5 Modalités et critères d'évaluation 24
1.5.1 Évaluation supervisée 24
1.5.2 Évaluation non supervisée 28
1.6 Conclusion 29

Chapitre 2
Panorama sur les méthodes d'analyse de flux de données

2.1 Structures de synthèse de flux de données 31
2.1.1 Échantillonnage 32
2.1.2 Histogrammes, Sketches, et Ondelettes 33
2.1.3 Maintenance en ligne des micro-clusters 36
2.1.4 Synthèse 40
2.2 Clustering de flux de données 41

2.2.1 Méthodes de type k-means . . . 42
2.2.2 Les cartes auto-organisatrices temporelles . . . 44
2.2.3 Les réseaux neuronaux à topologie adaptative . . . 49
2.2.4 Les réseaux de type ART . . . 55
2.2.5 Autres méthodes . . . 56
2.2.6 Synthèse . . . 57
2.3 Classification de flux de données . . . 58
2.3.1 La "dérive de concept" . . . 58
2.3.2 Méthodes à apprentissage adaptatif en ligne . . . 59
2.3.3 Méthodes par sélection des données . . . 61
2.3.4 Méthodes par pondération des données . . . 64
2.3.5 Méthodes à base d'ensemble de classifieurs . . . 65
2.3.6 Synthèse . . . 66
2.4 Analyse de changements dans les flux de données . . . 67
2.4.1 Classification à partir d'une seule classe . . . 68
2.4.2 Clustering et méthodes neuronales . . . 70
2.4.3 Méthodes des flux de données . . . 71
2.4.4 Synthèse . . . 71
2.5 Conclusion . . . 72

Chapitre 3
Aspects spécifiques à l'analyse des flux documentaires

3.1 Prétraitement et représentation des documents . . . 74
3.1.1 Indexation . . . 75
3.1.2 Modèles de représentation des documents . . . 77
3.1.3 Réduction de dimensionnalité . . . 81
3.2 Grandes familles de filtrage d'informations . . . 85
3.2.1 Filtrage basé sur le contenu . . . 85
3.2.2 Filtrage collaboratif . . . 88
3.2.3 Filtrage hybride . . . 89
3.3 Spécificités fonctionnelles du filtrage basé sur le contenu . . . 90
3.3.1 Tâches de filtrage actuelles . . . 90
3.3.2 Méthodes d'apprentissage du profil utilisateur . . . 91
3.3.3 Méthodes de seuillage dans les systèmes de filtrage d'informations . . . 97
3.4 Détection et suivi d'événements dans les flux documentaires . . . 101
3.4.1 Segmentation . . . 102
3.4.2 Détection et suivi d'événements . . . 102

3.5 Conclusion 104

Chapitre 4
Modèle d'apprentissage fondé sur le principe de la détection de nouveauté

4.1 Le modèle de filtre détecteur de nouveauté 106
4.1.1 Principe 106
4.1.2 Implémentation du modèle NDF 107
4.2 Réflexion sur les forces et les faiblesses du modèle NDF 110
4.3 Le modèle d'apprentissage incrémental ILoNDF 113
4.4 Utilisation du modèle ILoNDF pour la classification 117
4.4.1 Méthode de projection directe 117
4.4.2 Synthèse élémentaire de l'apprentissage 118
4.4.3 Méthode combinatoire 119
4.5 Illustration 120
4.6 Méthode de seuillage 122
4.7 Méthodes typiques pour la classification à partir d'une seule classe 124
4.7.1 Modèles statistiques multivariés 124
4.7.2 Réseaux de neurones auto-associatifs de type MLP 126
4.7.3 Les SVM monoclasses 127
4.8 Expérimentations 129
4.8.1 Résultats pour différentes conditions expérimentales 131
4.8.2 Une comparaison directe des méthodes de classification des données fortement multidimensionnelles 144
4.8.3 Synthèse des résultats expérimentaux 147
4.9 Conclusion 148

Chapitre 5
Personnalisation et modélisation des profils utilisateurs

5.1 Le modèle ILoNDF comme modèle utilisateur 151
5.1.1 Motivation 151
5.1.2 Principe 152
5.1.3 Évaluation 154
5.2 Analyse synthétique du besoin de l'utilisateur 163
5.2.1 Analyse de la pertinence des termes 164
5.2.2 Spécification du type de besoin de l'utilisateur 165
5.2.3 Adaptation de la fonction de décision 167
5.2.4 Évaluation 167

5.3 Adaptation à la dérive du besoin de l'utilisateur . . . 173
5.3.1 Motivation . . . 173
5.3.2 Méthode . . . 174
5.3.3 Évaluation . . . 175
5.4 Le système CASABLANCA : Nouvelles fonctionnalités . . . 181
5.4.1 Contexte . . . 181
5.4.2 Indexation conceptuelle indépendante de la langue des sites web . . . 182
5.4.3 Modélisation du profil utilisateur . . . 186
5.4.4 Gestion de l'intégration de nouveautés . . . 187
5.4.5 Combinaison de filtrage par contenu et de filtrage collaboratif . . . 188
5.5 Conclusion . . . 189

Chapitre 6
Méthodologie d'analyse temporelle de flux de données

6.1 Nouveaux indices de validité de méthodes de clustering . . . 192
6.1.1 Identification des variables de type noyau . . . 193
6.1.2 Coefficient de corrélation intra-cluster . . . 195
6.1.3 Coefficient d'isolation inter-clusters . . . 196
6.1.4 Coefficient global de validité (OqC) . . . 196
6.1.5 Évaluation . . . 197
6.2 Approche neuronale à des niveaux d'abstraction multiples . . . 201
6.2.1 Principe de mise en œuvre du modèle A-CLN . . . 202
6.2.2 Algorithme . . . 204
6.2.3 Évaluation . . . 207
6.3 Étiquetage des clusters . . . 211
6.4 Suivi de l'évolution des clusters à travers le temps . . . 213
6.4.1 Mise en correspondance . . . 213
6.4.2 Quantification de changements . . . 214
6.5 conclusion . . . 215

Conclusion

Annexes

Annexe A
Présentation des corpus de test

A.1 Reuters-21578 . . . 225
A.2 WebKB . . . 227

A.3 Google 229

Annexe B
Publications

Bibliographie **233**

Introduction

Les progrès de la technologie actuelle ont donné naissance à de nombreuses applications innovantes, qui se caractérisent par une croissance très rapide du volume et du débit des données à générer, à transporter, et puis à stocker et à traiter. Cette inflation concerne en premier lieu des données numériques de nature de plus en plus diverse : des images, des vidéos, des e-mails, des pages web, des messages instantanés, des appels téléphoniques, des fichiers de transactions commerciales et financières, des enregistrements de capteurs de surveillance et de contrôle, etc. Bien que les systèmes de stockage soient eux-aussi de plus en plus puissants et de moins en moins chers, une récente étude réalisée par IDC (International Data Corporation)[1] estime que la quantité des données numériques produites dans le monde dépasse désormais les capacités de stockage. Selon IDC, 281 milliards de gigaoctets de données numériques ont été créées et copiées en 2007. Cette quantité augmente très rapidement et devrait atteindre 1,800 milliards de gigaoctets en 2011, soit un taux de croissance annuel moyen d'environ 60%. Par ailleurs, IDC estime que les capacités de stockage mondiales approchent 264 milliards de gigaoctets en 2007 et sont donc inférieures aux volumes estimés des données produites durant la même période. Cet écart entre production et stockage devrait d'ailleurs rapidement se creuser dans les prochaines années (cf. Figure 1). Cette situation ouvre de nouveaux horizons à la gestion des données et fait naturellement appel à de nouvelles réflexions de la part de différentes communautés de recherche. En particulier, celles d'intelligence artificielle, d'apprentissage automatique, des statistiques, des bases de données, et des réseaux de communication. Une des perspectives de recherche s'est orientée vers la conception et le développement de nouvelles méthodes et techniques permettant le traitement des données potentiellement infinies, provenant de diverses sources de façon continue. C'est ce qu'on appelle aujourd'hui le domaine des "flux de données". Dans ce contexte, les flux sont si volumineux qu'il n'est plus possible d'envisager de stocker les données dans leur intégralité pour des fins de traitement, d'analyse ou d'interprétation. Même lorsque les données peuvent être stockées leur taille peut être si grande qu'il n'est pas possible d'effectuer des passages multiples sur les données. En outre, l'accès aux données ne peut être que séquentiel dans de nombreux cas et la durée du traitement des données doit être compatible avec leur débit d'arrivée. Un autre point à considérer concerne l'aspect évolutif et dynamique des flux de données qui constitue une source riche d'informations dans ce nouveau contexte. Il s'agit plus précisément de la prise en compte de la dimension temporelle des données de manière à pouvoir traiter de manière simultanée et concurrente l'analyse des régularités inhérentes au flux de données et celle des nouveautés, exceptions, ou changements survenant dans ce même flux au cours du temps. Peu de méthodes permettent aujourd'hui d'envisager de tels traitements bien que ceux-ci interviennent de manière centrale dans des applications aussi nombreuses que variées, parmi lesquelles il est possible de mentionner la surveillance de trafic dans les réseaux, l'analyse de données de transactions, le suivi d'actions sur des pages web dynamiques, ou encore l'analyse

[1] http ://www.idc.com/

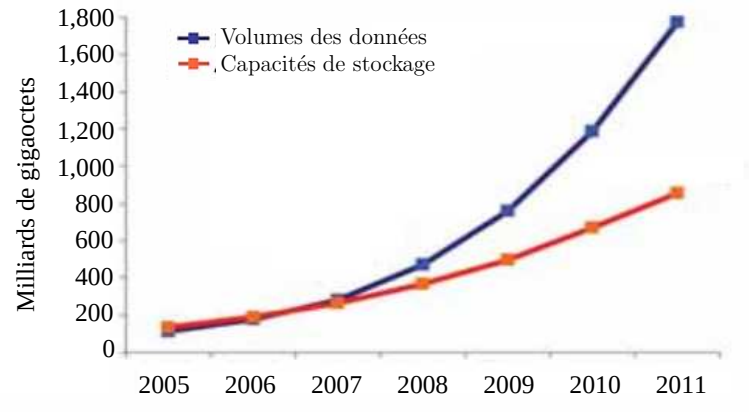


FIG. 1 – Évolution anuelle des volumes des données versus les capacités de stockage mondiales. D'après (Gantz et al., 2008).

des données en provenance de capteurs, mais aussi le filtrage d'informations, la veille scientifique et technologique, ou l'analyse temporelle des données expérimentales, telles que celles des puces ADN, en bioinformatique. Il est également probable qu'à terme, la grande majorité des systèmes informatiques seront confrontés à la gestion de flux importants de données de nature très diverse.

Ce travail de thèse propose un cadre fédérateur pour l'analyse des flux de données changeant au cours du temps. Le cadre principal d'application envisagé est celui du traitement des données fortement multidimensionnelles, à l'image des données en provenance du web ou d'autres sources documentaires accessibles en ligne. Les flux documentaires constituent un domaine en plein essor qui suscite actuellement un intérêt grandissant en raison de la profusion très rapide de l'édition numérique en ligne : articles, dépêches, brevets, rapports, études, mais aussi e-mails, sites web, messages de forums, listes de diffusion, etc. Quelques chiffres en témoignent : Selon Netcraft[2], ce sont près de 186 millions sites désormais enregistrés sur Internet, soit 30 millions de plus qu'en 2007. Bien sûr, cette inflation des ressources s'est inévitablement accompagnée d'un développement des méthodes d'analyse et de fouille de données pour extraire à partir de quantités de données de plus en plus importantes celles qui sont les plus pertinentes. À titre d'exemple, il y avait déjà plus de 26 millions de pages web référencées sur Google en 1998, puis 1 milliard en 2000. Aujourd'hui, ce sont quelque 1000 milliards de pages qui seraient accessibles depuis Google (Alpert et Hajaj, 2008). Toutefois, peu des méthodes actuelles sont adaptées aux contraintes opérationnelles et aux aspects dynamiques inhérents aux flux de données qui commencent juste ces dernières années à être abordés dans le domaine documentaire. Le travail que nous présentons dans ce manuscrit concerne donc l'élaboration d'un paradigme général d'analyse des flux de données multidimensionnelles qui permet, d'une part, de s'affranchir des limites inhérentes aux méthodes classiques et, d'autre part, d'intégrer la dimension temporelle des données si essentielle dans le contexte des flux évolutifs. De manière générale, nous plaçons notre travail dans le cadre de la détection automatique et du suivi de changements survenant dans les flux documentaires. Parmi les diverses applications porteuses du domaine, nous nous sommes en particulier intéressés au domaine du filtrage d'informations. Le filtrage d'informations peut être

[2]Une société britannique spécialisée dans l'étude et l'analyse du trafic Internet. http ://news.netcraft.com/

vu comme un processus consistant à adapter de manière dynamique la distribution d'informations en tenant compte à la fois de l'évolution des besoins des utilisateurs et des informations nouvelles entrant dans le flux. Ce type de processus semble en effet promis à un grand avenir, en particulier pour la diffusion personnalisée d'informations multimédia ou encore pour la diffusion d'informations ciblée à haute valeur ajoutée. Ainsi, une importante partie de notre travail a pu faire l'objet d'une implémentation dans un contexte industriel spécifique, en l'occurrence, le système CASABLANCA dont le rôle est la distribution ciblée de sites web multilingues par satellite.

Dans une première partie du manuscrit, qui se divise en trois chapitres, nous nous attachons à faire un état de l'art recouvrant les développements réalisés dans le domaine de l'analyse des flux de données. Pour le réaliser, nous commençons dans le premier chapitre par donner quelques généralités caractérisant les tendances, les défis, et les aspects fondamentaux relatifs à l'analyse des flux de données. Le deuxième chapitre est dédié à la présentation des différentes méthodes utilisées dans le domaine de l'analyse des flux de données, en insistant plus particulièrement sur le cas des flux de données multidimensionnelles. Nous terminons cette première partie en présentant dans un troisième chapitre les applications des flux de données dans le domaine documentaire, à savoir, le filtrage d'informations et la détection et le suivi de thèmes.

La seconde partie du manuscrit s'attache à décrire les principales contributions de notre travail de thèse dans le cadre de l'analyse des flux de données multidimensionnelles. De fait, notre contribution majeure réside dans le développement d'un modèle original d'apprentissage basé sur le principe de détection de nouveauté, nommé ILoNDF (Incremental data-driven Learning of Novelty Detector Filter), qui sera principalement présenté dans le chapitre 4. L'objectif principal de la détection de nouveauté est de souligner la nouveauté apparaissant dans des données encore inconnues, en exploitant la connaissance extraite à partir d'un ensemble de données de référence. Les données de référence se limitent à des exemples positifs des données normales ou familières, du fait de la difficulté, voire l'impossibilité dans certains cas, d'identifier a priori ce qui constituerait une nouveauté par rapport aux données déjà connues (ce qui amène au problème de la classification à partir d'une seule classe). L'idée fondamentale est donc d'apprendre un modèle des données normales disponibles et de l'employer pour identifier des données entrantes dérivant du modèle appris. Le modèle d'apprentissage que nous proposons est une adaptation d'un modèle antérieur de détection de nouveauté proposé dès les années 1976. Le modèle d'origine (NDF), doté d'un fonctionnement en ligne sans répétition d'apprentissage, présente un intérêt spécifique pour le traitement des flux de données. Toutefois, l'adaptation du modèle aux contraintes liées, d'une part, à l'élaboration d'une classification avec un taux considérable de recouvrement entre les classes (normale et nouvelle) et, d'autre part, au caractère évolutif des données arrivant en flux, a nécessité une modification en profondeur du modèle initial. De nouvelles méthodes d'initialisation, de nouvelles règles d'apprentissage, ainsi que des nouvelles stratégies pour l'exploitation du modèle pour des fins de classification ont donc été proposées. Le grand avantage du modèle ILoNDF est lié à sa capacité d'acquérir constamment de nouvelles connaissances relatives aux fréquences d'occurrence des variables descriptives des données utilisées pour faire l'apprentissage et leurs dépendances de co-occurrence dans ces mêmes données, ce qui rend le modèle robuste au bruit qui pourrait être présent dans la description des données. En outre, le modèle ILoNDF ne comporte aucun paramètre à régler avant ou pendant l'apprentissage ; il n'y a donc aucun besoin de faire des calculs supplémentaires et coûteux en matière d'optimisation de paramètres.

L'adaptation du modèle ILoNDF au cadre précis du filtrage d'informations est présentée dans le chapitre 5. Cette adaptation comporte trois volets distincts mais complémentaires, ayant pour

objectif commun la mise en place d'une stratégie de filtrage orientée-utilisateur plutôt qu'orientée-système. Le premier volet concerne la modélisation du profil utilisateur par l'intermédiaire de deux modèles de type ILoNDF appris respectivement à partir d'exemples positifs et négatifs du besoin d'informations de l'utilisateur. Cette modélisation présente l'originalité de reposer sur une analyse non paramétrée du contenu de ces modèles, qui se déroule naturellement de manière postérieure à l'apprentissage des modèles précités. Une telle analyse permet de mieux comprendre la nature du besoin d'informations de l'utilisateur en termes de critères de précision, d'exhaustivité et de contradiction. Ceci permet, entre autres, de définir une méthode de seuillage optimisant les résultats de filtrage en se basant directement sur le type du besoin de l'utilisateur et ses caractéristiques intrinsèques. Le deuxième volet est celui de la détection et du suivi de l'évolution du besoin de l'utilisateur. Très souvent, l'exploitation du retour de pertinence de l'utilisateur sur les documents fournis par le système permet la mise en place d'un processus d'auto-adaptation du profil utilisateur qui, s'il est bien mené, doit permettre d'améliorer l'efficacité du processus de filtrage. Cependant, le besoin de l'utilisateur est censé être figé ou faiblement évolutif dans le temps pour permettre au système de ne pas répéter certaines erreurs. Or, aujourd'hui, un défi important restant à relever par les systèmes de filtrage est de pouvoir s'adapter à la dérive du besoin de l'utilisateur qui peut être aussi bien graduelle que brusque. C'est donc dans cette perspective que nous étendons les fonctionnalités du modèle ILoNDF pour renforcer sa capacité d'oubli des données déjà apprises mais devenues périmées avec le temps. Toujours dans le même esprit d'orientation, le troisième volet vise à élaborer une stratégie de combinaison mettant en jeu différents modes de filtrage à l'aide du modèle ILoNDF.

Enfin, nous procédons à la généralisation de notre travail antérieur de manière à prendre en compte de manière explicite et unifiée la dimension temporelle des données. En effet, le modèle ILoNDF prend déjà en compte, par construction, la détection de changements susceptibles d'intervenir dans les flux de données. Ce modèle doit cependant être intégré dans un système de classification (supervisée ou non supervisée) lorsque les données arrivant en flux peuvent être réparties en classes multiples. Notre stratégie consiste donc à découper le flux de données en fenêtres temporelles sur lesquelles on applique une approche neuronale de classification. L'analyse temporelle est ensuite faite en comparant les classes construites pour chaque fenêtre de temps. Cependant, cette comparaison est une tâche délicate qui soulève de nombreux problèmes concernant le choix et le dimensionnement du modèle neuronal, les paramètres à ajuster, l'évaluation de la validité des classes obtenues, les relations entre les classes d'une même fenêtre ainsi celles-ci entre les classes de différentes fenêtres etc. Pour y faire face, de nouvelles mesures de validation et de comparaison, ainsi de nouvelles méthodes de classification ont été mises au point et seront présentées dans le chapitre 6.

Le manuscrit se conclut par un bilan des apports de notre travail et de nos résultats, ainsi que par les différentes perspectives qu'ouvre ce travail pour l'avenir proche et plus lointain.

Chapitre 1

Définitions et éléments clés d'analyse de données

Depuis très longtemps, les systèmes informatiques ont déjà dû traiter des données en quantités très importantes, comme par exemple dans les bases de données traditionnelles ou dans les entrepôts de données, à l'aide de techniques d'analyse et de fouille de données. Cependant, de nombreuses applications se trouvent confrontées aujourd'hui à gérer des flux importants de données à haut débit. Les méthodes qui sont aujourd'hui suffisamment matures pour traiter efficacement les problèmes classiques d'analyse de données, doivent être étendues de façon à faire face à de nouveaux défis liés à ce nouveau contexte. Ce chapitre s'attache d'abord à la description conceptuelle du paradigme de flux de données, les principaux défis qui leur sont propres, ainsi que les techniques de base pour les manipuler. Il se poursuit ensuite par un rappel succinct des méthodes classiques d'analyse de données, en se focalisant sur les méthodes numériques destinées à l'analyse de données multidimensionnelles. Le chapitre se termine par la question de l'évaluation des différentes méthodes d'analyse à l'aide de critères de validité formels.

1.1 Qu'est qu'un flux de données ?

Un flux de données est conceptuellement défini comme un ensemble ordonné de données, potentiellement infini, provenant de façon continue. Les données sont en général représentées par des n-uplets ayant tous la même structure. L'accès aux données est strictement séquentiel et l'ordre d'arrivée des données n'est pas maîtrisé. Les données, en raison de l'importance de leur volume et de leur débit d'arrivée, ne peuvent pas être exhaustivement stockées de manière persistante : les données passent et doivent être traitées "à la volée". De même, les flux de données peuvent encore être statiques et dynamiques. Dans un flux statique, les données sont vues comme un échantillon aléatoire d'une distribution fondamentalement stationnaire, alors que la distribution des données dans les flux dynamiques ou non stationnaires varie avec le temps.

Les modèles de flux de données capturent les caractéristiques essentielles des données de façon continue, ce qui les rend particulièrement utiles pour se substituer à des données massives issues des flux de données. Cependant, ces modèles sont confrontés à de nouvelles contraintes et de nouveaux défis qui se résument comme suit :

- L'infinité des flux introduit des problèmes pour les méthodes ayant besoin en entrée d'un ensemble fini de données. Ce type de méthodes dites "hors ligne" permet de revenir autant de fois que nécessaire sur les données. L'une des solutions possibles pour contourner ces

problèmes est de créer, à partir d'un flux infini, un ensemble fini de données afin de permettre l'utilisation de telles méthodes. Ces ensembles finis sont souvent appelés "*fenêtres temporelles*". Une autre solution est de faire appel aux méthodes dites "en ligne" au sens où les données ne peuvent être vues qu'une fois, dans un ordre séquentiel déterminé.

- L'incapacité de stocker la totalité des données issues d'un flux suggère très souvent l'utilisation des structures de résumés approximatives, connues dans la littérature sous le nom des synopsis (cf. Section 2.1). En conséquence, les modèles de flux de données ne sont pas en mesure de fournir des solutions exactes aux problèmes abordés.
- L'évolution temporelle des données dans le cas des flux non stationnaires exacerbe les difficultés liées à la construction de modèles régulièrement mis à jour de sorte à maintenir une connaissance actualisée des informations contenues dans les flux. Dans certaines situations, les changements dans un flux pourraient être dû à l'évolution des concepts déjà appris par le système ; ce problème est connu sous le nom de "*dérive de concepts*" où la distribution des données change inévitablement dans le temps. Dans d'autres situations, les changements dans un flux de données pourraient définir un nouveau concept, pas encore connu, ou bien pourraient refléter des anomalies affectant ledit flux (fraudes, intrusions ou encore erreurs de mesure, de fonctionnement, etc). Généralement, un système doit être capable de traiter de manière simultanée l'analyse des régularités inhérentes au flux de données et celle des nouveautés, exceptions, ou changements survenant dans un flux de données au cours du temps.
- Des contraintes "temps réel" peuvent s'ajouter dans le cadre des applications générant des flux de données à très haut débit. Dans tous les cas — même si ces contraintes ne sont pas critiques dans toutes les applications — le traitement appliqué à un élément du flux doit au moins être compatible avec le taux d'arrivée des éléments.
- L'accès aux flux de données est séquentiel et les données sont parfois redondantes et bruitées. Les techniques d'optimisation doivent donc être continues et dynamiques pour s'adapter aux conditions variables de ces systèmes. Néanmoins, l'exactitude des réponses peut parfois être sacrifiée afin d'optimiser l'utilisation de la mémoire ou de réduire le temps de réponse.

Un des enjeux principaux est donc de pouvoir apporter des solutions adaptées aux caractéristiques spécifiques de flux de données. Dans ce cadre, de nombreux travaux scientifiques ont été menés tout au long de ces dernières années et se sont poursuivis à ce jour. Il est possible de distinguer deux familles de solutions, chaque famille étant caractérisée par ses domaines d'applications et ses limites :

- Approches orientées données : Ces approches raisonnent en termes d'efficacité et partent du besoin de trouver un compromis entre l'exactitude des résultats et le coût des calculs. L'idée consiste à construire des résumés pour se substituer aux informations originales issues d'un flux de données. Ces résumés sont soit un sous-ensemble des données du flux choisi en utilisant certaines techniques comme l'échantillonnage, soit des transformations verticales ou horizontales des données en une ou plusieurs représentations approximatives de taille plus réduite. Une telle approche rend donc possible l'application des techniques d'analyse de données classiques au cas des flux de données (cf. Section 2.1).
- Approches orientées opérations : Ce sont les approches qui modifient les techniques classiques existantes ou proposent des techniques nouvelles afin de résoudre les défis liés aux calculs massifs en espace et en temps lors du traitement des flux de données. Parmi ces techniques, on peut citer, par exemple, l'utilisation des fenêtres temporelles (cf. Section 1.2.2).

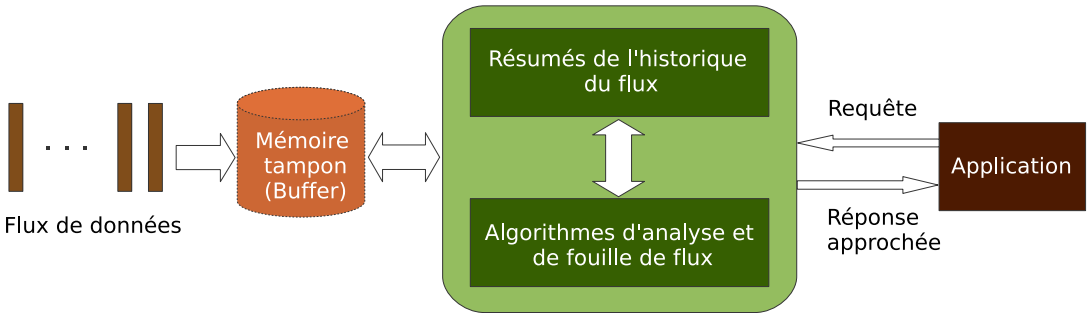


FIG. 1.1 – Architecture typique d'un modèle de flux de données.

La figure 1.1 montre l'architecture typique d'un modèle de flux de données.

1.2 Représentation du temps dans un flux de données

La représentation du temps est l'un des enjeux majeurs pour tout système informatique tendant à décrire un monde réel en prenant en compte son aspect dynamique. Différents aspects du temps peuvent être utilisés dans les problèmes du traitement des flux de données. Dans la plupart des cas, le temps est considéré comme une simple relation d'ordre (un flux de données est ainsi une séquence de vecteurs ordonnés selon le temps)[3]. La représentation du temps peut être "discrète" ou "continue". Dans la représentation discrète, le temps est réduit à une succession de données (ou événements) discrètes, alors que la représentation continue s'appuie sur un échantillonnage très fin du temps, sans tenir compte d'unité temporelle. Bien que cette dernière reflète mieux la perception physiologique du temps, elle est moins fréquemment adoptée dans les modèles dédiés au traitement des flux de données. Nous nous intéressons donc ici uniquement à la représentation discrète du temps.

1.2.1 Types de datation

Le temps étant une information primordiale dans les systèmes orientés flux de données, différentes méthodes de mise en correspondance du temps ont été développées. La plupart d'entre elles s'appuient sur des mécanismes de datation. La représentation du temps sous forme d'estampilles temporelles (timestamps) est la forme la plus souple d'un point de vue conceptuel. Les estampilles temporelles précisent simplement à quel moment (date et heure) les données ont été produites ou envoyées par flux. Ainsi, le marquage temporel des données fait partie du schéma du flux de données, et la datation est dite "explicite". Lorsque les estampilles ne sont pas ajoutées par la source du flux de données, le marquage temporel est fait par le système à l'entrée des données. Dès lors, les estampilles désignent le temps d'arrivée des données, et la datation est dite "implicite".

Notons que les estampilles explicites sont préférables quand chaque donnée du flux correspond à un événement réel produit à un moment particulier qui est d'importance pour la signification de

[3]D'autres modèles, rares, prennent en compte en plus de la relation d'ordre la durée entre deux éléments consécutifs d'un flux. Comme, par exemple, le temps considéré dans le traitement de la parole où la durée est significative.

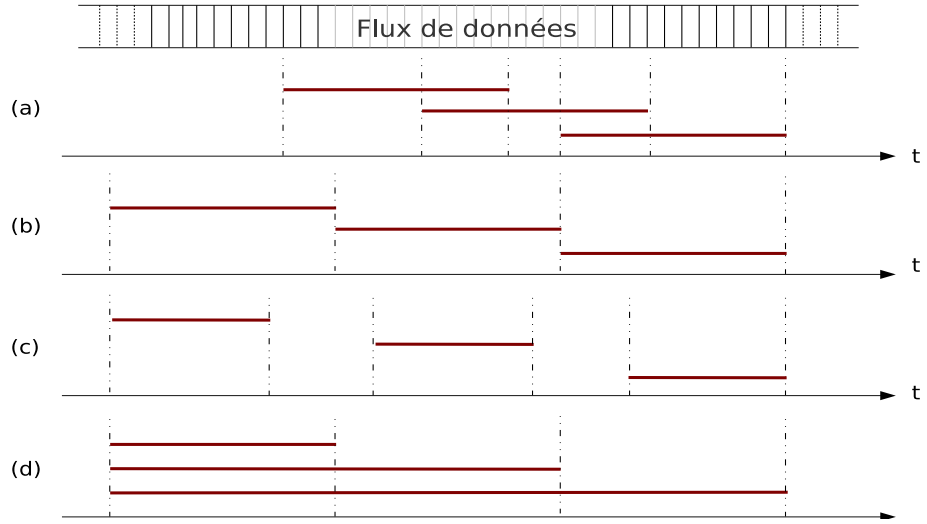


FIG. 1.2 – Exemples de différents types de fenêtrage. (a) fenêtres glissantes (b) fenêtres sautantes (c) fenêtres bondissantes (d) fenêtres fixes avec points de repère.

cet événement. Néanmoins, leur inconvénient est qu'aucune garantie n'est généralement accordée sur l'ordre d'arrivée des données. Autrement dit, les données peuvent arriver dans un ordre différent de l'ordre de leurs estampilles. Ce qui rend difficile de réaliser un traitement basé sur l'ordre des données, tel que le calcul des fenêtres temporelles.

1.2.2 Notion de fenêtrage

Un flux étant potentiellement infini — tel qu'il n'est plus possible d'envisager de stocker l'ensemble du flux — il est souvent indispensable de considérer des fenêtres d'intérêt sur le flux. Le but de fenêtrage temporel est donc de convertir le flux de données sous forme d'un ensemble fini de données. La définition des fenêtres temporelles est réalisable aisément par un découpage du flux en portions finies spécifiées par leur taille et leur décalage temporel. La spécification de la taille de la fenêtre temporelle est donnée soit en termes d'unité de temps (c.-à-d. intervalle de temps : seconde, minute, heur, jour, mois, année), soit en termes de nombre de données à considérer dans une fenêtre. On parle alors respectivement des *fenêtres physiques* et des *fenêtres logiques* (Golab et Lukasz, 2003). Principalement, quatre types de fenêtres peuvent être utilisés :

- Les fenêtres mobiles qui considèrent les N données les plus récentes dans le flux. Avec le temps, de nouvelles données font leur apparition dans la fenêtre, et d'autres données, plus anciennes, font leur disparition. Plusieurs modèles de fenêtrage peuvent être choisis en fonction du pas de décalage entre deux fenêtres successives. Le modèle le plus fréquemment utilisé est celui des fenêtres glissantes (Sliding windows) où le pas de décalage est inférieur à la taille de la fenêtre. On peut aussi citer les fenêtres sautantes (Tumbling windows) où le pas de décalage est égal à la taille de la fenêtre et les fenêtres bondissantes où le pas de décalage est supérieur à la taille de la fenêtre (cf. Figure 1.2).
- Les fenêtres fixes avec point de repère (Landmark windows) : Ces modèles considèrent toutes les données du flux à partir d'un point de repère, par exemple, au début de chaque heure. Ainsi, seulement une des bornes de la fenêtre se déplace, l'autre borne reste fixe (cf.

Figure 1.2).

- Les fenêtres amorties (Damped windows) où le poids de données décroît exponentiellement avec le temps. Par exemple, la valeur moyenne de l'ensemble des données de la fenêtre (avg_{new}) après l'arrivée d'une nouvelle donnée x peut être mise à jour de la façon suivante :

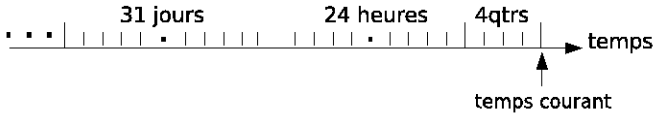
$$avg_{new} = avg_{old} \times p + x \times (1 - p); \quad 0 < p < 1$$

- Les fenêtres inclinées (tilted-time windows) : La structure des fenêtres inclinées repose sur l'intuition que l'on est souvent plus intéressé par les événements récents que par les événements plus éloignés. Il s'agit donc de retenir les données récentes avec une granularité fine et les données plus anciennes avec une granularité plus large. Il existe plusieurs variantes pour maintenir des fenêtres inclinées (Aggarwal, 2007). La figure 1.3 présente un exemple d'une structure de fenêtres inclinées dites naturelles basée sur l'échelle usuelle de temps : les quatre quarts d'heure les plus récents, les dernières 24 heures et enfin les derniers 31 jours. À partir de ce modèle, un archivage périodique des données dans des clichés (ou "snapshots", cf. Section 2.1.3) est effectué pour représenter les données apparues pendant la dernière heure avec une précision d'un quart d'heure, le dernier jour avec une précision d'une heure, etc. Dans l'exemple de Figure 1.3, le modèle stocke 4+24+31=59 clichés au lieu de $31 \times 24 \times 4 = 2976$ données avec un niveau acceptable de précision. Une variante qui fait encore diminuer le nombre de clichés à stocker est basée sur l'échelle logarithmique (voir Figure 1.3). Supposons que le cliché le plus récent représente les données du dernier quart d'heure, selon l'échelle logarithmique $\log_2(31 \times 24 \times 4) + 1 = 11.5$ clichés sont à stocker au lieu de $31 \times 24 \times 4 = 2976$ données. Le problème avec l'échelle logarithmique est que l'on risque de sauts importants entre les intervalles successifs de temps. Les structures pyramidales du temps offrent une solution intermédiaire où l'on conserve un grand nombre de clichés représentant le flux dans le passé proche et un peu de clichés pour le passé plus lointain, avec plusieurs niveaux de granularité entre les deux. Pour cela, les données sont stockées dans des clichés d'ordres variant de 1 à $log(T)$, où T est le temps écoulé depuis le début du flux pris en considération (c'est à dire la longueur du flux). Les clichés du i-ème ordre se produisent à des intervalles de temps réguliers de α^i, où α est un entier tel que $\alpha \geq 1$. À tout instant, seulement les m derniers clichés d'ordre i sont sauvegardés (m=$\alpha^l + 1$, $l \geq 1$). La figure 1.3 donne un exemple du nombre de clichés sauvegardés au temps 55, où les données arrivent selon un pas de temps de 1. Soulignons, à partir de cet exemple, qu'un grand nombre de clichés seraient communs aux différents ordres. Ces redondances peuvent être éliminées pendant le processus de la sauvegarde en stockant des clichés d'ordre i uniquement aux pas de temps divisible par 2^i mais pas par 2^{i+1}. Ainsi, les clichés d'ordre 0 sont uniquement stockés aux pas de temps impairs. Ces dernières structures sont désignées sous le nom de structures géométriques (Aggarwal, 2007).

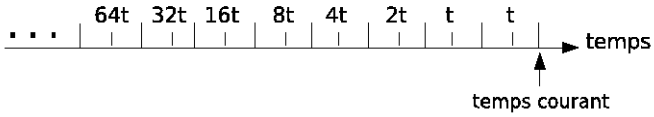
1.3 Modèles de flux de données : cas spéciaux

Un flux de données $a_1, a_2, \ldots$ arrive séquentiellement, et décrit un signal sous-jacent A, correspond à une fonction $A : [1..N] \rightarrow R$. Cette fonction est supposée unidimensionnelle, cependant, les données peuvent aussi être composées des flux unidimensionnels multiples ou des flux multidimensionnels. Les modèles de flux diffèrent selon la façon dont les a_i décrivent A ; on en distingue principalement trois (Muthukrishnan, 2005) :

- Modèle des séries temporelles : Chaque a_i est égal à A[i] et les a_i apparaissent dans un ordre croissant de i. Il y a donc une parfaite adéquation entre le flux de données et le signal



(a) une fenêtre basée sur l'échelle naturel



(b) une fenêtre basée sur l'échelle logarithmique

Ordre des clichés	Pas de temps
0	55 54 53 52 51
1	54 52 50 48 46
2	52 48 44 40 36
3	48 40 32 24 16
4	48 32 16
5	32

(c) Une structure pyramidale du temps pour $\alpha = 2$ et $l = 2$.

FIG. 1.3 – Trois exemples de fenêtres inclinées. Adapté de (Aggarwal, 2007).

qu'il décrit. Le modèle des séries temporelles se manifeste dans de nombreuses applications où les données arrivent dans un ordre croissant de temps, telles que la surveillance de volume de trafic IP sur un nœud réseau toutes les 5 minutes et la surveillance régulière de la température mesurée par un capteur thermique.

- Modèle dit du tiroir-caisse (Cash register) : Les données du flux ne correspondent pas à des éléments d'un signal sous-jacent mais à des variations d'un signal. Les a_i sont sous la forme de (j, I_i), $I_i \geq 0$. Ils sont des incréments de $A[j]$, $A_i[j] = A_{i-1}[j] + I_i$, où A_i est le signal après la i-ème donnée du flux. Le modèle du tiroir caisse est à utiliser lorsque les données se répètent fréquemment et ont donc juste à être insérées dans un multi-ensemble. Un exemple est la surveillance des adresses IP accédant à un site web ou à un serveur (N, le nombre de variables dans la fonction caractérisant le signal, représente donc ici le nombre d'adresses IP à surveiller).
- Modèle dit du tourniquet (Turnstile) : Similaire au modèle précédent mais sans faire des restrictions sur le signe des variations. Les a_i sont alors sous la forme (j, U_i), $U_i \geq 0, \leq 0$. Ils mettent à jour les valeurs du signal comme suit : $A_i[j] = A_{i-1}[j] + U_i$, où A_i est le signal après la i-ème donnée du flux. Le modèle du tourniquet est donc à utiliser lorsque les données peuvent être aussi bien insérées que supprimées, comme c'est le cas d'un ensemble de capteurs de température dans un environnement contrôlé qui ne fournissent que les variations de la température mesurée (N représente ici le nombre de capteurs).

Afin de ne créer aucune confusion dans la discussion qui suit, il est important de préciser que le terme de "séries temporelles" désigne principalement les flux de données unidimensionnelles. Plus formellement, une série temporelle est une suite d'observations correspondant à la même variable. Il s'agit donc d'analyser l'évolution d'une variable au cours du temps, de prédire une valeur future, ou bien d'extraire des tendances à partir des séries temporelles multiples. Il est aussi à noter que les séries temporelles sont souvent traitées de manière statique hors ligne, tandis que l'essentiel dans le cadre des flux de données se concentre sur les problèmes d'adaptation dynamique et d'exploitation en ligne des données. Nous nous concentrerons donc dans le reste de ce document sur le modèle des séries temporelles, principalement, dans le cadre des flux de données multidimensionnelles.

Avant d'aborder les méthodes qui sont actuellement élaborées dans le cadre de l'analyse des flux de données (ce qui fera l'objet du chapitre suivant), il nous paraît important de rappeler brièvement le principe générale de l'analyse de données et les principales méthodes associées qui sont souvent à la base des méthodes de flux de données.

1.4 Généralités sur l'analyse de données multidimensionnelles

L'analyse de données désigne l'ensemble des méthodes permettant de collecter, d'organiser, et de présenter des données de manière à pouvoir en extraire des informations valides, nouvelles, potentiellement utiles, et compréhensibles dans un but exploratoire ou décisionnel. L'analyse exploratoire consiste à synthétiser, résumer et structurer l'information contenue dans les données en mettant en évidence des propriétés des données et en suggérant des hypothèses. L'analyse décisionnelle, quant à elle, consiste à étendre les propriétés constatées sur un ensemble restreint de données à la population toute entière et à valider des hypothèses a priori ou formulées après une phase d'analyse exploratoire.

L'analyse de données est largement pluridisciplinaire, faisant appel principalement à deux vastes disciplines : 1) les mathématiques appliquées, avec l'analyse statistique des données, et 2) l'informatique et l'intelligence artificielle, avec l'apprentissage numérique et symbolique. C'est cette pluralité qui traduit la diversité de nombreuses discussions sur les différentes phases de l'analyse de données et les outils qui doivent être employés pour chacune d'entre elles selon le type d'informations que l'on entend acquérir sur les données et ce que l'on a prévu de faire de celles-ci (voir notamment (Saporta, 1990)). Ici nous insistons sur quelques caractéristiques fondamentales de ce type de processus. Une première partie est consacrée à l'analyse élémentaire des données, une deuxième partie fait découvrir des méthodes de base pour l'analyse des données multidimensionnelles. Cette section sera également l'occasion d'introduire quelques unes des notations utilisées dans la suite de ce document.

1.4.1 Les données et types de variables

Les données soumises à une technique d'analyse statistique se présentent le plus souvent sous la forme d'un tableau rectangulaire avec en ligne les individus (objets, instances, etc) et en colonne les variables (attributs, propriétés, etc). Le type de données/variables traitées conditionnent fortement les techniques utilisées. On en distingue deux types principaux :

- Les variables numériques peuvent être continues ou discrètes. Les variables continues sont les variables dont les valeurs appartiennent à un sous-ensemble infini de l'ensemble $\mathbb{R}$ des réels. Le salaire, l'âge, et la température sont des exemples des variables continues. Les variables discrètes, quant à elles, sont celles dont les valeurs appartiennent à un sous-ensemble fini de l'ensemble $\mathbb{N}$ des entiers naturels (exemples : nombre de produits achetés). Donc, ce qui distingue les données numériques des autres est qu'il s'agit de quantités sur lesquelles des calculs, tels que la moyenne, peuvent être effectués.
- Les variables catégorielles sont des variables dont l'ensemble des valeurs est fini. Ces valeurs sont numériques ou alphanumériques, mais quand elles sont numériques, ce ne sont que des codes et non des quantités (exemple : numéro de département). Les variables catégorielles sont dites nominales dans le cas où les différentes valeurs ou modalités ne contiennent pas de notion d'ordre (exemple : couleur des yeux). Si les différentes valeurs peuvent être classées, on parle alors de variables catégorielles ordinales (exemple : intensité d'une douleur "faible, moyenne, forte"). Les variables ordinales peuvent être rangées dans la famille des variables

discrètes et traitées comme telles. Un cas particulier des variables ordinales, mais aussi des variables discrètes, sont les variables binaires qui peuvent prendre uniquement deux valeurs (exemple : 0 et 1).

Toutes les méthodes ne gèrent pas tous les types de données ; par exemple, les données utilisées dans un réseau de neurones doivent être numériques. Néanmoins certaines opérations permettent de passer d'un type à un autre. À titre d'exemple, la discrétisation consiste à effectuer un découpage de l'ensemble des valeurs continues en des tranches afin d'obtenir un nombre fini d'états possibles. Les tranches elles-mêmes sont traitées comme des valeurs discrètes et ordonnées. Inversement, une analyse des correspondances multiples permet le passage du discret au continu. Aussi, les données catégorielles peuvent être transformées en données binaires tout simplement en remplaçant chaque variable catégorielle par autant de variables binaires que de modalités qu'elle présente. La valeur 1 ou 0 de chaque variable binaire signifie donc que la variable catégorielle a ou non la catégorie correspondante. De cette façon, les variables catégorielles peuvent aussi être l'objet de calculs statistiques. De même, le passage d'un type de données à un autre intervient très souvent dans le contexte de données mixtes, contenant à la fois des variables numériques et catégorielles, afin d'éviter la perte des relations possibles entre les variables de divers types. Plus de détails sont disponibles dans (Saporta, 1990).

Dans le domaine des flux de données, beaucoup de travaux ont déjà été consacrés au dévellopement des méthodes adaptées au traitement des données binaires, catégorielles et mixtes. Dans la suite de ce document, nous ferons à l'occasion référence à certains de ces travaux (cf. Chapitre 2). Pour l'essentiel, nous nous concentrerons sur les travaux dédiés au traitement des données numériques multidimensionnelles. Il faut donc tenir compte du caractère multidimensionnel lors de l'analyse de ce type de données. Il est courant de faire une réduction du nombre de variables et/ou des pondérations afin de donner plus ou moins d'importance à certaines de ces variables. Nous révélerons davantage de détails sur ces aspects dans les chapitres suivants, notamment au chapitre 3 portant sur les données documentaires.

1.4.2 Analyse statistique de données

La statistique offre toute une gamme de méthodes dont le choix dépendra de plusieurs facteurs : Le type de variables ; le nombre de variables (d'où les méthodes statistiques univariées, bivariées, et multivariées) ; et le but recherché de l'analyse des données : exploratoire (analyse descriptive) ou prise de décision (analyse inférentielle). L'*Analyse statistique de données* recouvre un ensemble des techniques statistiques descriptives pour l'analyse des données multivariées (ou multidimensionnelles).

Les méthodes d'analyse multivariée visent à transformer un ensemble de données à un nouveau système de coordonnées de manière à mettre en évidence les différentes relations pouvant exister entre les variables et/ou les données et par conséquence, réduire la dimensionnalité de celles-ci afin de simplifier leur représentation. Les méthodes statistiques sont souvent présentées en distinguant les méthodes verticales, consistant à réduire le nombre de données, des méthodes horizontales, consistant à réduire le nombre de variables. L'échantillonnage représente l'approche la plus communément utilisée pour la réduction des données (cf. Section 2.1.1). Une autre approche peut consister à utiliser des indicateurs statistiques sur les données, comme la moyenne, l'écart-type et la variance des données.

Les méthodes horizontales peuvent aussi être considérées comme des méthodes de réduction

de données, mais de manière moins directe, du fait qu'elles réduisent la complexité de ces dernières en ramenant le nombre de leurs variables à un petit nombre de composantes synthétiques, obtenues par des combinaisons souvent linéaires des variables initiales (les données peuvent être issues d'une procédure d'échantillonnage ou bien de l'observation d'une population toute entière). Ces méthodes permettent, entre autres, de rendre possible une représentation graphique très simplifiée des données et de détecter l'existence éventuelle de groupes de données et de groupes de variables. Elles sont aussi dites factorielles car elles tentent de déterminer les facteurs ou causes, qui permettent de distinguer les données entre elles. Il existe une panoplie de méthodes factorielles permettant de traiter différentes structures de données, telles que l'analyse en composantes principales (données numériques), l'analyse des correspondance (données catégorielles), et l'analyse des correspondances multiples (données mixtes).

L'analyse en composantes principales (ACP), introduite en 1901 par Pearson et intégrée à la statistique mathématique par Hotelling (1933), est souvent considérée comme la méthode de base de l'analyse factorielle des données multivaries. Il s'agit d'une transformation linéaire d'un grand nombre de variables intercorrélées de manière à obtenir un nombre relativement petit de composantes décorrélées de variance maximale et d'importance décroissante, appelées composantes principales. L'ACP cherche une solution où les composantes sont orthogonales entre elles, ce qui implique que ces nouvelles variables sont mutuellement décorrélées. Et pourtant, il a été démontré que la recherche simple des variables décorrélées ne suffit pas à assurer l'indépendance des variables. Il est alors apparu nécessaire de considérer des méthodes d'analyse en composantes indépendantes (ICA). Ces méthodes sont des extensions de l'ACP pour traiter des données multivariées afin d'en extraire des composantes linéaires aussi indépendantes que possible. Elles nécessitent ainsi l'utilisation d'une mesure de dépendance qui doit être choisie de manière adéquate (Hyvarinen et al., 2001). Nous renvoyons la discussion en détail de certaines des méthodes multivariées dans la section 4.7.1 du chapitre 4. Nous verrons particulièrement leur utilisation comme des méthodes de détection de changements dans le contexte de la classification à partir d'une seule classe, ceci notamment par l'exploitation des composantes retenues sur la base de certains critères. À cet égard, il est à préciser que ces méthodes ne sont pas adaptées au problème de détection de changements en ligne qui nous préoccupe. Nous les considérons juste pour des fins de comparaison de la performance de notre modèle (ILoNDF) par rapport aux méthodes existantes.

1.4.3 Apprentissage

L'apprentissage artificiel joue un rôle fondamental dans tout processus d'analyse fine et intelligente des données. Il englobe toute méthode permettant de construire un modèle de l'information présente dans les données. Plus précisément, un algorithme d'apprentissage reçoit un ensemble d'exemples d'apprentissage et doit produire des règles générales qui représentent les informations obtenues à partir de ces exemples. À ce stade, il est intéressant de faire référence aux différents contextes d'apprentissage, à savoir, l'apprentissage supervisé, l'apprentissage non-supervisé, et l'apprentissage semi-supervisé.

- Dans le cadre de l'apprentissage supervisé, l'on dispose d'un ensemble des données préalablement étiquetées sous la forme d'entrées/sorties (les exemples d'apprentissage) et l'on cherche à trouver ou approximer la fonction qui permet d'effectuer automatiquement l'étiquetage le plus vraisemblable sur d'autres données (dites les exemples de test). Ce mode d'apprentissage peut servir à des fins d'analyse, de prise de décision et de prévision. Toutes les méthodes de classification sont donc des algorithmes d'apprentissage supervisé (cf. Sec-

tion 1.4.5).

- L'apprentissage non-supervisé utilise des données sans étiquettes. Dans de telles conditions, l'apprenant ne reçoit aucune information indiquant quelles devraient être ses sorties ou même si celles-ci sont correctes. Il doit donc découvrir par lui-même la structure des données à partir des corrélations existantes entre les exemples d'apprentissage qu'il observe. Ce mode d'apprentissage concerne plutôt des tâches d'analyse exploratoire des données. Toutes les méthodes de clustering sont des algorithmes d'apprentissage non supervisé (cf. Section 1.4.4).
- Dans le cadre d'apprentissage semi-supervisé, aussi appelé apprentissage par renforcement, l'apprenant ne dispose que d'indications imprécises sur la justesse de sa sortie (par exemple, échec/succès). Il s'agit donc de produire de plus en plus de sorties correctes en recourant à un processus d'essais et d'erreurs.

Les algorithmes d'apprentissage requièrent typiquement un ensemble d'exemples à partir duquel un modèle est construit, on parle alors d'un ensemble d'apprentissage. Dans un cadre supervisé, il est aussi indispensable d'avoir à disposition un autre ensemble d'exemples pour évaluer la validité de la solution trouvée par ce modèle, on parle alors d'un ensemble de validation ou de généralisation. Ces ensembles sont nécessairement disjoints et souvent établis à partir d'un ensemble unique d'exemples en le divisant en n parties égales et en construisant n modèles, en écartant à chaque fois une des n parties qui servira pour le test et en utilisant les $n-1$ autres parties pour l'apprentissage. Cette procédure est connue sous le nom de *validation croisée* (en anglais "cross-validation"). Elle permet notamment d'augmenter la capacité de généralisation d'un modèle d'apprentissage et d'arrêter le processus d'apprentissage avant de trop se spécialiser sur l'ensemble d'apprentissage, ce phénomène est connu sous le nom de "sur-apprentissage", ou "apprentissage par cœur" (ou encore "overfitting" en anglais). Dans la pratique, ce sont les capacités de généralisation d'un modèle qui vont établir les possibilités de l'appliquer à d'autres exemples que ceux vus au cours de la phase d'apprentissage.

Par défaut, la plupart des algorithmes requièrent de multiples itérations sur l'ensemble des exemples d'apprentissage, c'est donc le mode d'apprentissage "*hors ligne*" ou "*batch*", mais il existe bien des algorithmes capables d'effectuer l'apprentissage en ligne d'une manière incrémentale, c'est à dire, de pouvoir accepter les exemples les uns après les autres au fur et à mesure de leur disponibilité et d'être capable d'affiner le modèle après la présentation de chacun des exemples. Ce type d'algorithmes est dit "*en ligne*", nous leur accordons une grande importance dans le cadre de notre travail du fait qu'elles sont soumises à la contrainte du traitement en ligne de flux de données. Néanmoins, l'aspect dynamique lié à l'apprentissage en ligne pose des problèmes difficiles loin d'être complètement résolus aujourd'hui. Il s'agit principalement du dilemme entre la stabilité et la plasticité d'un modèle d'apprentissage (Grossberg, 1976). Ce dilemme peut s'exprimer par les interrogations suivantes : Comment un système peut-il être adaptatif à l'égard d'informations pertinentes, et stable à l'égard d'informations non pertinentes ? Comment apprendre constamment de nouvelles informations non familières (plasticité) sans oublier celles antérieurement acquises (stabilité) ? Comment oublier des informations "abîmées" dans un souci permanent d'efficacité et de fiabilité du processus d'apprentissage ? Pour contourner ce problème, il existe trois sortes de solutions : 1) sélectionner les exemples pertinents ou qui peuvent le devenir en s'inspirant des méthodes d'apprentissage actif (Bordes et al., 2005), 2) interrompre l'apprentissage avant la phase d'instabilité, et 3) protéger les informations acquises en rendant l'apprentissage conditionnel. Cette dernière solution est mise en œuvre dans les réseaux neuronaux de type ART que nous présenterons dans la section 2.2.4 du chapitre 2. Au-delà de ce

problème important, pointe un autre problème, auquel la plupart des algorithmes en ligne sont sensibles. Il s'agit de la dépendance du modèle construit à l'ordre de présentation des données d'apprentissage. Ce problème ne se pose que dans le cadre d'un apprentissage en ligne du fait qu'on apprend en une seule passe sur les données.

Enfin, il convient de préciser que les approches d'apprentissage peuvent aussi être réparties en différentes familles, comme celles de l'apprentissage numérique et symbolique, selon le type des données manipulées. Les méthodes par apprentissage symbolique ne sont pas aujourd'hui, à notre avis, véritablement opérationnelles sur de grandes quantités de données. Nous nous plaçons donc dans le cadre des méthodes numériques qui possèdent de nombreux avantages par rapport aux méthodes symboliques. Citons, entre autres, la capacité à traiter des données sous forme binaire ou non binaire, les facultés considérables de synthèse d'informations qui assurent souvent plus d'efficacité en termes de temps de calcul, la robustesse vis-à-vis du bruit et l'avantage de faciliter la visualisation des données et des résultats obtenus à l'issue du traitement des données. Parmi les méthodes d'apprentissage numérique les plus fréquemment utilisées figurent les réseaux de neurones et les machines à vecteurs supports. Nous présentons ci-dessous les principes d'apprentissage dans les réseaux de neurones. Les méthodes des machines à vecteurs supports seront aussi présentées après un rapide rappel des méthodes de classification supervisée et non supervisée.

L'apprentissage de réseaux connexionnistes

Les réseaux de neurones artificiels (RNA) sont des modèles fonctionnels qui s'inspirent du fonctionnement du cerveau humain plus ou moins librement. Ils peuvent être considérés comme une organisation cohérente d'unités de traitement élémentaires, nombreuses et interconnectées. L'organisation des unités élémentaires en réseau induit l'émergence de propriétés nouvelles, analogues à celles que l'on attribue habituellement à l'intelligence humaine. Un réseau de neurones se caractérise par son architecture, les types de ses neurones constituants, sa dynamique, et sa stratégie d'apprentissage.

De façon générale, un neurone est défini par une activité A, une fonction de transfert f et un vecteur de poids des connexions d'entrée du neurone w, appelé "vecteur référent" ou "prototype", cf. Figure 1.4. Le neurone reçoit des données en entrée (ou bien des activités des neurones afférents) sous la forme d'un vecteur $x = (x_1, x_2, \ldots, x_n)$ et répond en sortie unique par une activité A_i. Cette sortie peut se décrire en fonction de l'état d'activation du neurone σ_i :

$$A_i = f(\sigma_i) \tag{1.1}$$

L'état d'activation d'un neurone peut principalement être calculé de deux manières différentes, d'où l'on peut définir deux types de neurones distincts : Le neurone "sommateur" et le neurone "distance". Le neurone "sommateur" réalise une somme des entrées x_j pondérées par les poids w_{ij} de ses connexions. Cette somme est ensuite transformée par une fonction de transfert f qui produit la sortie A_i du neurone. La fonction f peut être du type fonction linéaire, fonction à seuil, fonction sigmoïde, fonction gaussienne, etc. Pour un neurone i on a donc :

$$A_i = f\left(\sum_{j=1}^{n} w_{ij} \times x_j\right) \tag{1.2}$$

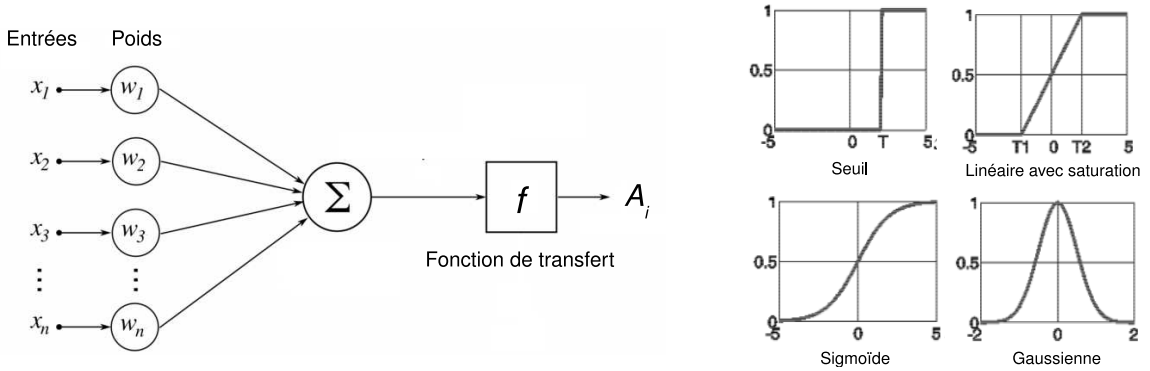


FIG. 1.4 – (a) Schéma fonctionnel d'un neurone artificiel. (b) Exemples de fonctions de transfert.

Le neurone "distance", quant à lui, effectue un calcul de distance — le plus souvent de type distance Euclidienne — entre le vecteur d'entrée x et le vecteur référent w d'un neurone i :

$$\sigma_i = \|w - x\|^2 = \sum_{j=1}^{n} (w_{ij} - x_j)^2 \tag{1.3}$$

Cette distance peut aussi être transformée par une fonction de transfert. La sortie d'un neurone i est ainsi donnée par :

$$A_i = f\left(\sum_{j=1}^{n} (w_{ij} - x_j)^2\right) \tag{1.4}$$

Les réseaux de neurones recouvrent une grande diversité d'architectures standards ayant chacune leurs spécificités en termes d'organisation des connexions entre neurones. On distingue généralement deux types d'architectures : Les réseaux unidirectionnels ou non bouclés (ou encore "feedforward") et les réseaux récurrents ou bouclés. Les réseaux unidirectionnels ont leurs neurones organisés sous la forme d'une ou de plusieurs couches successives. La propagation des informations se fait de la couche d'entrée vers la couche de sortie au travers d'éventuelles couches intermédiaires mais sans retour en arrière. C'est à dire que la sortie du réseau ne peut influencer son entrée. Ce sont donc des systèmes statiques sans mémoire. Le perceptron multicouche (MLP) adopte ce type d'architecture. Les réseaux récurrents possèdent une structure similaire à celle des réseaux unidirectionnels mais complétée par des connexions entre neurones de la même couche ou vers des couches afférentes. Le fonctionnement de ces réseaux est séquentiel et adopte un comportement dynamique qui prend en compte l'aspect temporel dans les calculs. Il existe bien d'autres types d'architectures des réseaux à connexions complexes qui ne trouvent pas leur place dans les deux types précédemment cités, comme par exemple, les gaz neuronaux simples (Neural Gas) ou incrémentaux (Growing Neural Gas) que nous verrons dans la suite de ce document. L'architecture du réseau étant fixée, le but de l'apprentissage est d'estimer les poids des connexions entre neurones pour remplir au mieux la tâche à laquelle le réseau est destiné.

L'apprentissage connexionniste peut être perçu comme un processus de mise à jour des poids des connexions au sein d'un réseau de neurones, de manière à lui permettre d'établir des associations entrées-sorties afin d'apporter des bonnes solutions au problème qui lui a été posé.

Les règles d'apprentissage dérivent généralement de la loi de Hebb (Hebb, 1949). Cette loi, d'inspiration biologique, renforce les connexions entre deux neurones activés simultanément en proportion de leur niveau d'activité. Pour bien illustrer cette règle, considérons deux neurones i et j ayant, à un pas d'apprentissage t, les activités A_i et A_j (comprises entre 0 et 1), reliés par une connexion de poids w_{ij}. La loi de Hebb modifie la valeur de w_{ij} en y ajoutant la quantité $\Delta w_{ij} = \eta \times A_i(t) \times A_j(t)$, où η est une constante positive qui contrôle la vitesse d'apprentissage, et appelée *taux d'apprentissage* : Ainsi, la mise à jour prend la forme suivante :

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \eta \times A_i^{(t)} \times A_j^{(t)} \tag{1.5}$$

Dans la pratique, un problème immédiat avec cette règle est qu'elle mène à une croissance non bornée de poids w_{ij}. Donc, cette règle est souvent complétée en y ajoutant un autre terme permettant la diminution des poids des connexions (par exemple, dans (Oja, 1982), ce problème a été efficacement contourné par l'introduction d'un facteur d'oubli ou un terme de normalisation des poids). Une autre règle d'apprentissage, que nous verrons par la suite, couramment appelée "anti-Hebb", a pour effet d'atténuer les connexions entre les neurones simultanément actifs, ce qui se traduit par une modification des poids contraire au principe de Hebb.

L'apprentissage est la caractéristique principale des RNA et il peut se faire de différentes manières et selon différentes règles. Principalement, les réseaux de neurones se divisent en deux grandes familles selon le type d'apprentissage, supervisé ou non supervisé :

- Pour les réseaux de neurones à apprentissage supervisé (Perceptron multicouche (Rosenblatt, 1958), Hopfield (Hopfield, 1982), LVQ (Learning Vector Quantization) (Kohonen, 2001), etc.), on présente au réseau des entrées en parallèle avec les sorties attendues pour ces entrées. Le réseau doit alors se configurer, c'est-à-dire adapter les poids des connexions du réseau, afin que les sorties effectives qu'il fournit correspondent bien aux sorties attendues. La différence entre la sortie du réseau et la sortie attendue est alors utilisée par adapter les poids des connexions du réseau de façon à corriger son comportement. Ce processus est répété de façon itérative jusqu'à obtenir le meilleur comportement. La règle générale d'apprentissage qui est appliquée dans le cas de l'apprentissage supervisé peut donc être considérée comme une règle de minimisation d'une fonction de coût (Widrow et Hoff, 1960; Rosenblatt, 1958), ou encore d'une fonction d'énergie (Hopfield, 1982).
- Pour les réseaux de neurones à apprentissage non supervisé (les cartes auto-organisatrices de Kohonen (SOM) (Kohonen, 2001), les réseaux de type ART (Adaptive Resonance Theory) (Carpenter et Grossberg, 1987a), etc.), on présente différentes entrées au réseau et on le laisse évoluer librement jusqu'à ce qu'il se stabilise. Les poids des connexions entre les neurones du réseau sont alors modifiés de manière à ce que les exemples possédant les mêmes caractéristiques produisent les mêmes sorties. Ce type d'apprentissage est également appelé "apprentissage compétitif" du fait que les neurones sont en compétition pour être actifs. Pour tout exemple présenté à l'entrée du réseau, seul un neurone, dit *neurone gagnant*, est sélectionné comme étant le plus proche au sens d'une distance au vecteur d'exemple en entrée. À ce stade, deux types d'apprentissage compétitif sont à différencier selon la manière de modifier des poids d'un ou plusieurs neurones à chaque itération. 1) L'apprentissage compétitif de type "winner-take-all" adapte uniquement le vecteur référent du neurone gagnant de façon à rapprocher ce vecteur du vecteur de l'exemple présenté en entrée. L'algorithme LBG (Linde et al., 1980) est un exemple d'un tel type de réseaux. L'apprentissage de type "winner-take-all" a l'inconvénient d'être souvent très lent et des précautions particulières sont à prendre lors de l'initialisation pour faire face

au problème des neurones morts, c'est à dire des neurones qui ne gagnent jamais et qui ne sont donc pas adaptés. Ce problème provient souvent d'une mauvaise initialisation des vecteurs référents des neurones[4]. C'est pourquoi, on préfère plutôt utiliser l'apprentissage de type "winner-take-most". 2) L'apprentissage compétitif de type "winner-take-most", qui à chaque itération adapte le vecteur référent du neurone gagnant pour le rapprocher du vecteur d'entrée, ainsi que les vecteurs référents des autres neurones mais avec un facteur moins important, dépend principalement de la proximité entre les vecteurs référents et le vecteur d'entrée. Les cartes auto-organisatrices de Kohonen (SOM, pour Self Organizing Maps) sont des exemples d'un tel type de réseaux.

Les RNA offrent une diversité de techniques appropriées pour la résolution de problèmes très variés : la classification, le clustering, la modélisation, la prédiction, la compression des données et le contrôle des systèmes complexes, etc. Ils connaissent aussi un intérêt important pour le traitement des données spatio-temporelles, étant parmi les premiers modèles à intégrer la dimension temporelle des informations dans leur mode de fonctionnement (les tous premiers modèles sont caractérisés par l'utilisation de fenêtres temporelles et de délais temporels, comme les TDNN — Time Delay Neural Networks (Waibel et al., 1989)). Toutefois, il faut mentionner que l'on leur reproche souvent d'être des "boîtes (presque) noires" : il est difficile de savoir comment les résultats sont produits, ce qui rend parfois délicate l'interprétation des résultats obtenus même s'ils sont bons. Par ailleurs, le temps d'apprentissage des RNA peut être extrêmement long, c'est probablement pourquoi ils n'ont pas encore suscité l'intérêt qu'ils méritent dans le domaine des flux de données. Mais c'est aussi parce que la gestion de la composante temporelle des données n'est pas encore vraiment considérée dans ce nouveau domaine. Pour cela, nous pensons que les méthodes connexionnistes finiront avec le temps par s'imposer dans le domaine des flux de données. Nous reviendrons donc sur ces méthodes dans le chapitre 2.

Nous passons maintenant brièvement en revue les principes des méthodes classiques de classification supervisée et non supervisée. Leur adaptation à la classification des flux de données sera présentée plus loin dans les sections 2.2 et 2.3 du chapitre 2.

1.4.4 Clustering

La classification non-supervisée ou non-dirigée — clustering — est une étape importante du processus de l'analyse de données. Elle vise à découvrir la structure intrinsèque d'un ensemble de données en formant des regroupements homogènes — clusters — auxquels appartiennent les données qui partagent des caractéristiques similaires, sans aucune connaissance a priori sur ces données. Ceci s'opère en tenant compte principalement de deux contraintes interdépendantes que constituent une forte similarité intra-classe (deux données appartenant à une même classe doivent être aussi similaires que possible) et une forte dissimilarité inter-classes (deux données appartenant à des classes différentes doivent être aussi dissimilaires que possible). Le clustering trouve son intérêt dans des domaines très divers où les facteurs humains rendent les données nombreuses et difficiles à apprendre. Il s'agit donc de simplifier une réalité complexe pour laquelle aucune classification a priori n'est préalablement établie, en révélant des structures cachées et en isolant des données aberrantes (outliers) si elles existent.

[4]Ce problème est en tout point similaire à celui qui se pose lors de l'initialisation de la méthode K-means, qui pourrait être assimilée à une méthode neuronale de type "winner-take-all" (cf. Section 2.2.1).

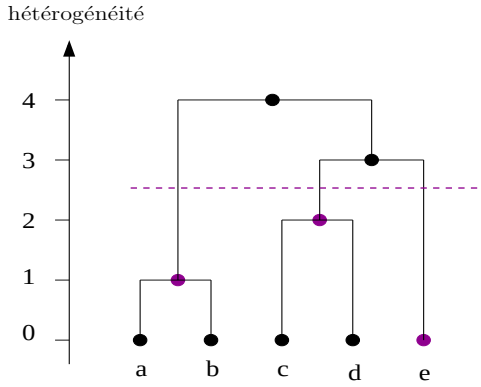


FIG. 1.5 – Exemple de dendrogramme

Les méthodes de clustering se répartissent en trois grandes familles : les méthodes de partitionnement, les méthodes hiérarchiques et les méthodes hybrides.

- Les méthodes de partitionnement fournissent directement une partition unique des données en un certain nombre de clusters. Une donnée est alors affectée au cluster dont elle est la plus proche au sens d'une distance ou d'un indice de similarité. Des algorithmes de réaffectation sont souvent appliqués afin d'améliorer continûment la qualité des clusters construits. La plupart du temps le nombre de clusters de la partition à construire doit être défini à l'avance. La méthode de k-means, ainsi que ses dérivées (cf. Section 2.2.1) représentent des exemples typiques de ces méthodes. L'avantage de certaines de ces méthodes est que leur complexité est linéaire, c.-à-d. leur temps d'exécution est proportionnel au nombre d'individus, ce qui les rend applicables à de grands volumes de données. Cependant, ces méthodes sont sensibles aux conditions initiales, on n'a donc pas un optimal global, mais seulement la partition la plus optimale possible à partir de celle de départ. Un autre inconvénient de ces méthodes est qu'elles favorisent certaines formes de cluster en fonction du critère de similarité choisi. Par exemple, les méthodes basées sur les critères de "lien moyen" ne détectent bien que les formes sphériques des clusters.
- Les méthodes hiérarchiques génèrent une séquence de partitions emboîtées, de la plus fine à la plus grossière, organisées sous forme de dendrogramme (arbre hiérarchique), dont la figure 1.5 présente un exemple. La hiérarchie peut être formée en utilisant soit une approche ascendante (agglomérative) ou bien une approche descendante (divisive). Dans une approche ascendante, on commence avec autant de clusters qu'il y a de données : ainsi, chaque données à classifier constitue initialement son propre cluster. Les clusters sont ensuite fusionnés par étapes successives en utilisant des mesures de similarité, et ceci jusqu'à ce que tous les clusters soient fusionnés en un seul cluster (le niveau le plus élevé de la hiérarchie), ou encore, jusqu'à atteindre un critère d'arrêt. Inversement, l'approche descendante commence par un cluster qui contient l'ensemble complet des données. Ensuite, à chaque étape, un cluster est désigné pour être divisé en deux clusters "descendants" dans le but d'optimiser un critère d'évaluation. La division s'opère de façon à ce que la distance entre les deux descendants soit la plus grande possible, de façon à créer deux clusters bien séparés. Le processus se répète jusqu'à obtenir une partition formée de singletons, ou tant qu'une condition d'arrêt n'a pas été atteinte. Ces méthodes ne présentent pas les deux inconvénients majeurs des méthodes de partitionnement mentionnés ci-dessus, cependant, la complexité algorithmique de ces méthodes est relativement élevée.
- Le principe des méthodes hybrides est de combiner les points forts des méthodes de parti-

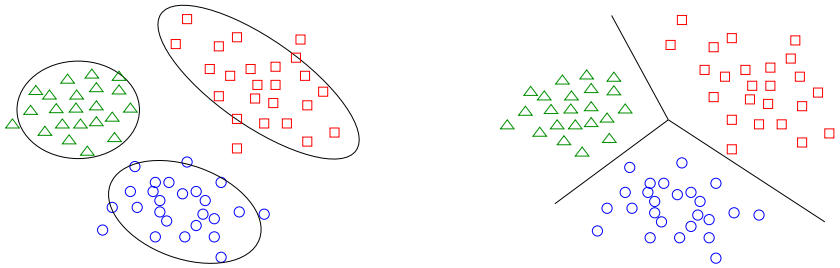
tionnement et des méthodes hiérarchiques, à savoir la rapidité des premières et la précision et l'absence d'a priori des secondes. Par exemple, on peut effectuer un premier clustering en utilisant une méthode de partitionnement en fixant un nombre de clusters suffisamment grand pour limiter le risque de fusionner des classes naturelles. Puis on effectue une classification ascendante hiérarchique sur les centres de ces clusters et non sur les données initiales. Il existe de nombreux exemples de méthodes hybrides, tels que BIRCH (Zhang et al., 1996), CHAMELEON (Karypis et al., 1999), CURE (Guha et al., 1998) et ROCK (Guha et al., 2000). Certaines de ces méthodes seront détaillées ultérieurement dans le chapitre qui suit.

Outre les méthodes qui viennent d'être exposées ci-dessus, trois autres méthodes sont souvent considérées parmi les plus aptes à détecter la structure de clusters un peu complexes. Il s'agit de : méthodes de clustering par estimation de densité (Ester et al., 1996), méthodes de clustering par la théorie des graphes (Günter et Bunke, 2002), et méthodes de clustering par utilisation de grilles (Schikuta et Erhart, 1997). Il s'avère également intéressant de diviser les méthodes de clustering en deux sous-types : le clustering strict (Hard clustering) et le clustering recouvrant (Soft clustering). Dans le cas du clustering strict, les clusters sont disjoints, i.e. chaque objet est affecté à un seul cluster, au contraire du cas du clustering recouvrant, où les clusters peuvent avoir un certain degré de recouvrement et ainsi un objet peut être affecté à plusieurs clusters. Pour plus de détails sur les techniques de clustering, nous renvoyons le lecteur au travail de synthèse présenté dans (Jain et al., 1999).

1.4.5 Classification

La classification supervisée — catégorisation — s'oppose à la classification non supervisée en ce qui concerne l'étiquetage des données d'apprentissage : Les données d'apprentissage sont regroupées a priori en divers concepts sémantiques, appelés classes, ce qui nécessite une étape d'étiquetage préalable à l'apprentissage. Les données étant étiquetées à l'avance sous la forme <donnée, classe>, il s'agit alors d'assigner automatiquement une ou plusieurs des classes prédéfinies à des données inconnues — dont on ne connaît pas à l'avance les classes d'appartenance —. Une méthode de classification supervisée tente donc de trouver un modèle — une fonction de décision — qui explique le lien entre des données, leur variables et leurs classes d'appartenance et qui induise le minimum d'erreur de classement.

Il existe de nombreuses familles d'approches pour réaliser une classification supervisée, on en distingue deux principalement : les approches génératives et les approches discriminatives (cf. Figure 1.6). Le premier type d'approches cherche à trouver une description explicite des classes indépendamment les unes des autres en en apprenant des modèles correspondant à leurs propriétés intrinsèques. La décision de classification est prise en utilisant une mesure de similarité entre une donnée à classer et les modèles des classes. Les classifieurs de Bayes Naïfs (NB) sont des exemples typiques de telles approches (Lewis, 1998). De leur côté, les approches discriminatives cherchent à trouver une description implicite des classes par la définition des frontières de décision permettant de les séparer le mieux possible. La classification se fait simplement selon les positions dans lesquelles se trouvent les données à classer par rapport aux frontières de décision. Ces approches sont particulièrement intéressantes lorsque les données ont une distribution inconnue et difficilement modélisable. Elles recouvrent, entre autres, les réseaux de neurones de type MLP, la méthode de k plus proches voisins, les arbres de décision, et les machines à vecteurs supports (SVM). En règle générale, les approches discriminatives sont plus performantes que les approches



(a) (b)

FIG. 1.6 – Exemples de deux approches de classification supervisée (a) Approche générative (b) Approche discriminative.

génératives. Or, il a été montré que dans certains cas, voir par exemple (Japkowicz, 1999), les approches génératives semblent plus avantageuses. De tels constats ont amené à développer des solutions combinant les deux approches afin de bénéficier de leur éventuelle complémentarité (Jebara, 2003).

Parmi les approches supervisées auxquelles une attention particulière a été accordée les techniques dites à base d'ensemble de classifieurs reposent sur le principe qu'un ensemble de classifieurs aboutirait à une solution meilleure que celle d'un seul classifieur. Ce constat est d'autant plus vrai que les classifieurs mis en jeu sont indépendants. Il existe de nombreuses méthodes pour rendre indépendants les classifieurs, telles que le "bagging" (Breiman, 1996) et le "boosting" (Schapire, 1990). L'apprentissage consiste donc à déterminer les solutions partielles des classifieurs et à trouver une manière efficace pour les combiner afin d'obtenir la solution générale.

Les méthodes de classification sont très nombreuses et diversifiées, nous nous limitons ici à présenter plus en détail les méthodes à vecteurs supports (SVM). Nous verrons d'autres méthodes dans les chapitres qui suivent. Pour le reste, nous invitons le lecteur intéressé à se tourner vers les ouvrages consacrés et parmi lesquels on peut notamment citer (Bishop, 2006), et (Sebastiani, 2002).

Les machines à vecteurs supports

Initiées par les travaux de V. Vapnik en théorie de l'apprentissage statistique (Vapnik, 1995), les machines à vecteurs supports, ou SVM en abrégé, ont été reconnues très efficaces pour le traitement de données complexes et de grandes dimensions. En version standard, les SVM sont introduites comme des techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination binaire. Il s'agit de déterminer un hyperplan optimal permettant une séparation linéaire des données en deux classes distinctes (que constituent les exemples positifs et négatifs d'apprentissage). La détermination de l'hyperplan s'appuie sur deux étapes clés : 1) Une transformation non linéaire ($\Phi : \mathcal{X} \rightarrow \mathcal{F}$) des données de l'espace d'origine $\mathcal{X}$ vers un espace dit de redescription $\mathcal{F}$ de dimension supérieure (éventuellement infinie), muni d'un produit scalaire, dans lequel il est probable qu'il existe un séparateur linéaire. Cette transformation est réalisée au moyen d'une fonction symétrique dite "noyau" sans nécessiter une représentation explicite des données dans ce nouvel espace. Pour être admissible, une fonction noyau

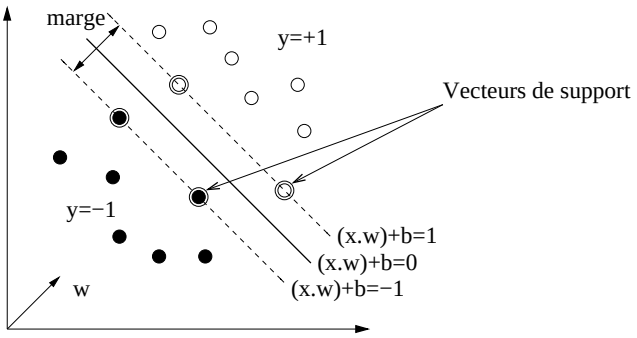


FIG. 1.7 – Hyperplan séparateur et vecteurs supports dans un espace à deux dimensions

k doit vérifier les conditions de Mercer (1909) qui se résument à vérifier que cette fonction corresponde bien à un produit scalaire dans l'espace $\mathcal{F}$, ce qui peut s'écrire sous la forme : $k(x_1, x_2) =< \Phi(x_1), \Phi(x_2) >$ pour tout $(x_1, x_2) \in \mathcal{X}^2$ où $< . , . >$ dénote le produit scalaire. 2) Dans l'espace $\mathcal{F}$, un séparateur linéaire est choisi comme celui qui maximise la marge entre l'hyperplan qu'il définit et les données. Ceci est fait en formulant le problème comme un problème d'optimisation quadratique, pour lequel il existe des algorithmes connus.

Supposons dans un premier temps que l'on dispose d'un ensemble de l exemples d'apprentissage $\{x_1, \ldots, x_l\}$, $x_i \in \Re^n$, associés à des étiquettes correspondant à leurs classes respectives $\{y_1, \ldots, y_l\}$, $y_i \in \{+1, -1\}$, et que les exemples sont linéairement séparables. Il existe alors une fonction de décision $f(x) = sign(< w.x > +b)$ tel que :

$$\begin{cases} < w.x_i > +b \geq 1 & \text{si} \quad y_i = +1 \\ < w.x_i > +b \leq 1 & \text{si} \quad y_i = -1 \end{cases} \tag{1.6}$$

L'hyperplan séparateur ($< w.x > +b = 0$) est paramétré par w, le vecteur normal à l'hyperplan, et b, la distance minimale de l'hyperplan à l'origine. On appelle "*hyperplan optimal*" l'hyperplan qui est situé à la distance maximale des exemples les plus proches parmi l'ensemble des exemples d'apprentissage ; autrement dit, c'est l'hyperplan qui maximise la marge de séparation (cf. Figure 1.7). La valeur de la marge étant tout simplement $2/\|w\|$, maximiser la marge revient à minimiser $\frac{1}{2} < w.w >$ sous les contraintes de Eq. 1.6. Les exemples particuliers qui satisfont ces contraintes sont appelés "*vecteurs supports*". La solution de ce problème d'optimisation est obtenue en écrivant le lagrangien, qui conduit à la formulation duale du problème sous la forme :

$$\max_{\alpha} \ \mathcal{W}(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j < x_i.x_j > \quad \text{avec} \quad \alpha_i \geq 0, \sum_i \alpha_i y_i = 0 \tag{1.7}$$

où les α_i sont les multiplicateurs de Lagrange associés à chacun des exemples. Les exemples x_i qui interviennent dans la solution ($\alpha_i \neq 0$) sont les vecteurs supports. La résolution de ce problème d'optimisation quadratique permettra de connaître la fonction de séparation :

$$f(x) = sign(\sum_i \alpha_i y_i < x_i.x > +b) \tag{1.8}$$

Plaçons nous maintenant dans le cas général où les exemples d'apprentissage ne sont pas

linéairement séparables. Il est possible de reformuler les contraintes de Eq. 1.6 sous la forme :

$$\begin{cases} < w.x_i > +b \geq +1 - \xi_i & \text{si} \quad y_i = +1 \\ < w.x_i > +b \leq -1 + \xi_i & \text{si} \quad y_i = -1 \\ \xi_i \geq 0 \quad \forall i \in \{1 \dots l\} \end{cases} \tag{1.9}$$

où ξ_i sont des variables de relâchement permettant à certains exemples de se situer du mauvais coté de l'hyperplan. SVM attribuera une classe fausse à un exemple si le ξ_i correspondant est supérieur à 1. La somme de tous les ξ_i représente donc une borne d'erreurs. Il faut alors minimiser la fonction objectif ($\frac{1}{2}w.w + C\sum_{i=1}^{l}\xi_i$) sous les contraintes de Eq. 1.9. Le paramètre C est une constante positive permettant de borner le nombre d'erreurs toléré : plus C est grand plus la solution est proche de la solution de la SVM à marge dure ($C = \infty$).

Le problème se résout alors de manière similaire au cas linéairement séparable en replaçant les produit scalaires des vecteurs des données d'apprentissage $< x_i.x_j >$ dans les formules énoncés ci-dessus par leurs correspondants dans le nouvel espace $< \Phi(x_i).\Phi(x_j) >$, ou tout simplement par la fonction noyau associée à l'espace original des données. La nouvelle fonction de séparation est donc :

$$f(x) = sign(\sum_i \alpha_i y_i k(x_i.x) + b); \tag{1.10}$$

et les α_i sont bornés par C ($0 \leq \alpha_i \leq C, \ \ \forall i$).

Les SVM peuvent utiliser différents types de fonctions de noyau dont les plus connues sont : les fonction polynomiales de degré d, les fonctions à base radiale (RBF) et les fonctions sigmoïdales. Cela suggère qu'une meilleure solution avec les méthodes de type SVM est directement liée au bon choix de noyau, et de ses paramètres, en fonction du contexte de l'application. La littérature propose plusieurs extensions de la version standard des SVM. Parmi ceux-ci, notons, principalement, les méthodes SVM en apprentissage non supervisé (Xu et al., 2004), les SVM multiclasses (Hsu et Lin, 2002), et les méthodes SVM à une classe (Schölkopf et al., 2001). Dans le chapitre 2 (Section 2.3.2), nous verrons des adaptations possibles des méthodes SVM au mode de fonctionnement en ligne permettant leur application à la classification des flux de données. Nous reviendrons aussi sur leur application à la classification à partir d'une seule classe dans le chapitre 4.

1.4.6 Synthèse

Toutes les problématiques classiques d'analyse de données que nous avons décrites interviennent également dans le contexte de l'analyse de flux de données. Cependant, il ne faut pas s'attendre à ce que les méthodes classiques qui existaient déjà et qui rentrent dans le cadre de l'analyse des données statiques fonctionnent dans ce nouveau contexte. Bien que nombreuses et très diversifiées, ces méthodes ne sont pas conçues pour gérer des données en grandes quantités qui varient très fréquemment. Au contraire, elles sont généralement établies hors ligne à partir des données statiques sur lesquelles plusieurs passages sont possibles. De plus, ces méthodes sont fondées sur certaines hypothèses, comme celle qui suppose que les données sont indépendantes et identiquement distribuées, qui ne peuvent plus se soutenir dans le cadre des flux de données évolutifs. En bref, les caractéristiques spécifiques des flux de données sont forcément plus contraignantes et nécessitent un important travail d'adaptation et de réflexion par rapport aux méthodes classiques. Par ailleurs, la prise en compte de la dimension temporelle des données

est un des enjeux primaires pour l'analyse de flux de données qui implique de pouvoir aussi, d'une part, extraire des régularités susceptibles d'interprétations sémantiques et, d'autre part, prendre en compte les changements qui surviennent au cours de temps. Cela entraîne bien sûr de nouveaux problèmes qui ne se posaient pas dans le cadre de l'analyse des données statiques.

1.5 Modalités et critères d'évaluation

L'évaluation est une tâche difficile, mais pourtant indispensable, pour mesurer les performances des méthodes d'analyse qui ont été élaborées et pouvoir les comparer entre elles. Heureusement, les critères d'évaluation qui sont conçus dans le cadre des données statiques s'appliquent également dans le cadre des méthodes destinées au traitement de flux de données. Le choix d'un critère d'évaluation dépend principalement du type de supervision, des connaissances dont on dispose a priori sur les données, et aussi de l'objectif final de l'analyse. Trois critères sont en général utilisés :

- Les critères externes se fondent sur la connaissance d'une partition de référence sur les données. L'évaluation consiste donc à mesurer la différence entre la partition de référence et la partition obtenue par un système de classification ;
- Les critères internes s'intéressent à tester si le modèle de classification est intrinsèquement cohérente avec les données. Ils sont souvent à relier au sens du critère "optimisé" pendant le processus de classification ;
- Les critères relatifs permettent de comparer deux ou plusieurs classifications pour en choisir "la meilleure".

À cet égard, diverses suggestions pour chacun de ces critères sont mentionnées dans la littérature. Le but de cette section n'est pas de présenter toutes les mesures existantes mais de donner quelques exemples parmi celles qui sont les plus fréquemment utilisées.

1.5.1 Évaluation supervisée

Les critères d'évaluation externes sont adaptés aux méthodes d'apprentissage supervisé pour lesquelles l'ensemble des exemples sont a priori étiquetés par leurs classes d'appartenance, mais aussi aux méthodes d'apprentissage non supervisé si on dispose de données étiquetées pour l'évaluation. Tout d'abord, nous considérons un problème simple de classification où les exemples peuvent être divisés en deux classes (C_i ou $\overline{C}_i$), et nous voulons évaluer un système qui nous indique à laquelle de ces deux classes un exemple appartient ; c'est donc une décision binaire. Nous verrons ensuite comment fusionner les mesures de performance pour une évaluation multiclasse.

Pour mieux illustrer les différentes mesures qui vont suivre, nous donnons un exemple d'un tableau de contingence (Tab.1.1) qui met en relation la partition de référence des exemples et celle issue d'un système de classification binaire. Chaque entrée représente le nombre de décisions avec le résultat spécifié. Par exemple, VP_i (les vrais-positifs à l'égard de la classe C_i) désigne le nombre de données appartenant à la classe C_i et qui ont été assignées correctement par le système.

Précision, Rappel et critères associés

Les performances en termes de décisions binaires sont généralement évaluées par deux mesures issues de la recherche d'information : la *précision* et le *rappel*. La *précision* est définie comme étant le pourcentage d'assignations correctes parmi la totalité de celles faites par le système.

		Standard de référence	
		C_i	$\overline{C}_i$
Système	C_i	VP_i (vrais-positifs)	FP_i (faux-positifs)
	$\overline{C}_i$	FN_i (faux-négatifs)	VN_i (vrais-négatifs)

TAB. 1.1 – Tableau de contingence à la base des mesures d'évaluation supervisée

Cette mesure reflète la capacité du système à rejeter toutes les données non pertinentes. Donc, une précision de 100% signifie que toutes les données assignées à la classe en question sont effectivement membres de cette classe ; ainsi il n'y a pas de bruit. De son côté, le *rappel* est le pourcentage d'assignations correctes faites par le système parmi toutes celles qui sont correctes. Cette mesure reflète donc la capacité du système à détecter toutes les données appartenant à chaque classe. Ainsi, un rappel de 100% signifie que toutes les données d'une classe y ont été assignées par le système. En utilisant les notations de Tab. 1.1, la précision (P_i) et le rappel (R_i) sont donnés par les formules suivantes :

$$P_i = \frac{VP_i}{VP_i + FP_i} \tag{1.11}$$

$$R_i = \frac{VP_i}{VP_i + FN_i} \tag{1.12}$$

Idéalement, un système de classification devrait être conçu de manière à permettre d'atteindre à la fois de hautes valeurs de précision et de rappel. Néanmoins, ces deux critères sont généralement inversement proportionnels : une très forte précision ne peut être acquise qu'au détriment d'un rappel faible et vice-versa. Dans ce cas, il est nécessaire de réaliser un bon compromis entre ces deux facteurs vis-à-vis de l'utilisation finale du système. En utilisant une moyenne harmonique pondérée, la mesure F_β de van Rijsbergen (van Rijsbergen, 1979) permet d'équilibrer la précision et le rappel. Elle est définie de la manière suivante :

$$F_\beta = \frac{1}{\alpha(\frac{1}{P_i}) + (1-\alpha)\frac{1}{R_i}} = \frac{(\beta^2+1)\ P_i\ R_i}{\beta^2 P_i + R_i} \tag{1.13}$$

Le paramètre β est un coefficient de pondération positif permettant l'ajustement du niveau d'importance accordé à la précision et au rappel. La valeur de β varie de 0 à ∞, où $\beta = 0$ ($\alpha = 1$) correspond à un système qui n'attache aucune importance au rappel ($F_0 = P$) et $\beta = \infty$ ($\alpha = 0$) correspond à un système qui n'attache aucune importance à la précision ($F_\infty = R$).

Une autre manière de tenir compte à la fois de la précision et du rappel est de calculer le "break-even point", c'est-à-dire le point — le plus élevé — où la précision et le rappel sont égaux. Cette mesure est couramment utilisée dans la littérature pour comparer la performance de plusieurs systèmes de classification (voir, par exemple, (Joachims, 1998)). Plus ce point se rapproche de 100%, plus le système est performant à la fois en précision et en rappel. Bien entendu, il existe d'autres mesures pour l'évaluation et la comparaison des systèmes de classification, notamment le taux d'erreur ($E = \frac{FP+FN}{VP+FP+FN+VN}$) ou, de façon équivalente, le taux de classifications correctes ($A = 1 - E$).

Qualité de l'ordonnancement

Les mesures présentées précédemment sont dédiées à l'évaluation des systèmes de classification "dure". Elles ne sont pas adaptées à l'évaluation des systèmes de "ranking" qui, au lieu de donner une décision "dure" sur l'appartenance ou non d'une donnée à une classe, ordonnent les données par ordre de pertinence pour la classe en question. Il reste à noter que la plupart des système de classification fournissent également une probabilité de pertinence pour en déduire une décision binaire. Par conséquent, l'évaluation de l'ordonnancement proposé par un système de classification permet de rendre compte de la performance propre du système indépendamment du choix d'un seuil de décision. De même, certaines applications — comme l'aide au diagnostic médical — exigent non seulement l'assignation des données à leur classe d'appartenance mais aussi l'ordonnancement des données assignées à chaque classe. Dans ce dernier cas l'évaluation de l'ordonnancement fait partie de l'évaluation globale du système de classification. Nous présentons ci-après deux manières permettant d'évaluer la qualité de l'ordonnancement.

Le recours à la courbe Rappel-Précision (R-P) apporte une solution au problème de l'évaluation de l'ordonnancement (Salton, 1971). Cette courbe est une représentation graphique de la relation existant entre le rappel (axe des abscisses) et la précision (axe des ordonnées) calculée pour toutes les valeurs seuils possibles, c.-à-d. à chaque niveau de rappel[5]. Comme les niveaux de rappel ne sont pas unifiés pour l'ensemble des classes, on retient généralement les 11 niveaux standards de rappel variant entre 0 et 1 avec un pas de 0.1, pour pouvoir ensuite calculer la moyenne sur toutes les classes. En pratique, il arrive que ces valeurs de rappel ne peuvent pas être atteintes, les valeurs de la précision doivent donc être interpolées[6]. L'aire sous la courbe R-P peut ainsi être vue comme une mesure globale de l'efficacité d'un système : Plus la courbe est haute, meilleur est le système.

Il existe également des mesures résumant l'information contenue dans une courbe R-P en une seule valeur pour comparer simplement plusieurs modèles de classification. La précision moyenne non-interpolée (MAP) est un exemple de telles mesures. En parcourant une liste de données tirées par ordre décroissant de pertinence par rapport à une classe, cette mesure est définie comme la somme des valeurs de précision à chaque point où une donnée pertinente apparaît dans la liste, divisée par le nombre total de données pertinentes.

Extension à l'évaluation multiclasse

Jusqu'à présent nous avons présenté le problème d'évaluation à travers des mesures concernant une seule classe. À partir de ces mesures de base, il est possible d'évaluer la performance d'un système sur plusieurs classes à l'aide des mesures de moyennes. Il y a principalement deux types de moyennes : macro-moyenne et micro-moyenne (Salton, 1971; Lewis, 1991). La macro-moyenne consiste à faire tout simplement une moyenne globale des mesures sur l'ensemble des classes ;

[5]De façon similaire, les courbes ROC (Receiver Operating Characteristics) sont couramment utilisées dans le domaine de la détection et du traitement du signal (Davis et Goadrich, 2006). Les courbes ROC présentent en abscisse le taux de faux-positifs et en ordonnée le taux de vrais-positifs. Par ailleurs, l'aire sous la courbe (AUC - Area Under the curve) peut être vue comme la mesure globale de l'efficacité d'un système.

[6]Le principe de l'interpolation est le suivant : Soient i et j deux points de rappel, et $i < j$. Si au point i, la précision est inférieure à la précision au point j, alors on impose à la précision de i d'être égale à celle du point j. Suivant ce principe une valeur de précision pour un rappel nul correspond au niveau maximal de précision obtenu pour un rappel quelconque. Concrètement, cela signifie qu'on remplit un creux de la courbe par une ligne horizontale.

	Macro-moyenne (M)	Micro-moyenne (μ)
Précision	$P^M = \frac{1}{\|C\|}\sum_{i=1}^{\|C\|} P_i$	$P^\mu = \frac{\sum_{i=1}^{\|C\|} VP_i}{\sum_{i=1}^{\|C\|}(VP_i + FP_i)}$
Rappel	$R^M = \frac{1}{\|C\|}\sum_{i=1}^{\|C\|} R_i$	$R^\mu = \frac{\sum_{i=1}^{\|C\|} VP_i}{\sum_{i=1}^{\|C\|}(VP_i + FN_i)}$

TAB. 1.2 – Les macro- et micro-moyennes de la Précision et du Rappel sur plusieurs classes

ainsi, cette moyenne donne un poids égal à chaque classe. Tandis que la micro-moyenne regroupe d'abord les données de chaque classe dans un tableau de contingence global et calcule ensuite les mesures à partir de celui-ci, elle donne donc un poids égal à chaque donnée. La différence de comportement de ces deux moyennes se manifeste essentiellement dans le cas où les différentes classes sont de tailles très variées. De manière générale, la micro-moyenne a une tendance à être dominée par la performance du système de classification sur les classes les plus peuplées (classes communes), alors que la macro-moyenne tend à être dominée par la performance du système de classification sur les classes les moins fréquentes (classes rares).

Les indices de pureté et d'entropie

Une des méthodes d'évaluation de la pertinence d'un clustering fait appel à un étiquetage a priori des données. On suppose donc connaître l'ensemble des classes des données $L = \{l_1, l_2, \ldots, l_{|L|}\}$ et on cherche à évaluer dans quelle mesure une méthode de clustering est capable de fournir des clusters $C = \{c_1, c_2, \ldots, c_{|C|}\}$ en accord avec l'étiquetage des données. Nous présentons ici deux indices usuels pour ce faire : la pureté et l'entropie (Zhao et Karypis, 2002).

La pureté d'un cluster correspond au pourcentage de données appartenant à la classe majoritaire. Ainsi, la pureté est égale à 1 si toutes les données associées au cluster sont issues de la même classe. La pureté associée à un clustering est donc définie comme suit :

$$Purity = \sum_{k=1}^{|C|} \frac{|c_k|}{n} \frac{max_{i=1}^{|L|} |c_k^i|}{|c_k|} \tag{1.14}$$

où $|c_k|$ represente le nombre total de données associées au cluster c_k, $|c_k^i|$ représente le nombre de données de la classe l_i qui sont associées au cluster c_k.

L'entropie permet d'évaluer comment les classes sont distribuées ou réparties dans les clusters. Pour un ensemble de $|C|$ clusters, l'entropie est la suivante :

$$Entropy = \sum_{k=1}^{|C|} \frac{|c_k|}{n} \left(-\frac{1}{log|L|} \sum_{i=1}^{|L|} \frac{|c_k^i|}{|c_k|} log\frac{|c_k^i|}{|c_k|} \right) \tag{1.15}$$

L'entropie est toujours positive et prend la valeur nulle dans le cas idéal. Donc, plus l'entropie est faible, meilleur est le clustering.

1.5.2 Évaluation non supervisée

Une seconde famille de critères dits "internes" est adaptée aux méthodes d'apprentissage non supervisé pour lesquelles l'on ne dispose d'aucune connaissance a priori sur les classes. Ces critères sont basés sur l'évaluation de la cohésion des classes obtenues par le système et l'écart entre les différentes classes. Les critères les plus classiques sont basés sur les mesures d'inerties intra-classe et inter-classes. Nous présentons ici ces mesures de base et deux autres variantes, à savoir, l'indice de Davies-Bouldin (DB), et l'indice de Calinski-Harabasz (CH), qui sont tout spécialement destinés à la sélection automatique du nombre de clusters dans le cadre de la classification non supervisée. Pour une vue plus complete, d'autres mesures peuvent être trouvées dans (Halkidi et al., 2001) et (Theodoridis et Koutroumbas, 1999).

Inerties intra-classe et inter-classes

L'inertie intra-classe évalue l'homogénéité des données présentes à l'intérieur des classes alors que l'inertie inter-classes évalue l'hétérogénéité entre les différentes classes. Ces mesures peuvent être définies comme suit (Lebart et al., 1982) :

$$I_{intra} = \frac{1}{|\mathcal{C}|} \sum_{c_j \in \mathcal{C}} \frac{1}{|c_j|} \sum_{x_i \in c_j} \|x_i - z_j\|^2 \tag{1.16}$$

$$I_{inter} = \frac{1}{|\mathcal{C}|^2 - |\mathcal{C}|} \sum_{c_i \in \mathcal{C}} \sum_{c_j \in \mathcal{C}, c_j \neq c_i} \|z_i - z_j\|^2 \tag{1.17}$$

avec $\mathcal{Z} = \{z_1, \ldots, z_n\}$ les centres de gravité des classes $\mathcal{C} = \{c_1, \ldots, c_n\}$, et $|.|$ dénote le cardinal d'un ensemble. Des valeurs faibles d'inertie intra-classe désignent des classes homogènes vis à vis des données qui les composent et des valeurs fortes d'inertie inter-classes désignent des classes hétérogènes entre elles. Ces grandeurs constituent donc des critères de qualité des classes, et par conséquent de la classification proprement dite.

Indice de Davies-Bouldin

L'indice de Davies-Bouldin (DB) (Davies et Bouldin, 1979) tient compte à la fois des inerties intra-classe et inter-classes de sorte que l'évaluation de la distance entre classes dépendrait de l'inertie intra-classe. En d'autres termes, les grandes classes (dont l'inertie intra-classe est forte) devraient être proportionnellement loin les unes des autres pour qu'elles soient bien séparées, alors que les petites classes (dont l'inertie intra-classe est relativement faible) peuvent être plus proches les unes des autres tout en restant encore bien séparées. Cet indice est défini par :

$$DB = \frac{1}{|C|} \sum_{k=1}^{|C|} \max_{k \neq l} \left(\frac{I_{intra}(c_k) + I_{intra}(c_l)}{DC(c_k, c_l)} \right) \tag{1.18}$$

où $|C|$ est le nombre de classes, $I_{intra}(c_i)$ est l'inertie intra-classe de c_i, et $DC(c_i, c_j)$ est la distance Euclidienne entre les centres de gravité des classes c_i et c_j. La meilleure classification est celle donnant la valeur minimum de l'indice DB.

Indice de Calinski-Harabasz

L'indice de Calinski-Harabasz (CH) (Calinski et Harabasz, 1974) considère une solution minimisant l'interie intra-classe et maximisant l'inertie inter-classes. Soit un total de n données de moyenne $\bar{x}$ regroupées en k classes, l'indice de CH est donné par la formule suivante :

$$CH(k) = (\frac{SSB}{k-1})/(\frac{SSW}{n-k}) \tag{1.19}$$

avec

$$SSB = \sum_{c_k \in C} |c_k| \; ||z_k - \bar{x}||^2; \quad SSW = \sum_{c_k \in C} \sum_{x_i \in c_k} ||x_i - z_k||^2$$

La meilleure classification est celle donnant la valeur maximum de l'indice CH.

1.6 Conclusion

Nous avons commencé ce chapitre avec le constat que de plus en plus d'applications sont confrontées au problème d'analyse de données temporelles qui arrivent de façon continue en très grandes quantités. L'avènement de telles applications a accompagné l'émergence du paradigme dit de "flux de données" et des nouveaux défis majeurs auxquels il fallait trouver des solutions. Une première partie de ce chapitre a mis en évidence le contexte et les caractéristiques spécifiques des flux de données. Une seconde partie a introduit les notions essentielles relatives au processus d'analyse de données en mettant en avant certaines méthodes numériques d'intérêt potentiel pour le traitement des données multidimensionnelles.

À l'heure actuelle, il existe très peu (ou pas) de méthodes d'analyse qui opèrent, de manière exacte, sur les flux de données. Très souvent, des solutions approchées ont dû être envisagées sous les exigences et les diverses contraintes liées au traitement des flux de données. Toutefois, la recherche sur le domaine des flux de données se développe encore de manière importante en raison des nombreuses applications potentielles qui sont concernées par ce type de problèmes. Un panorama des solutions existantes et des travaux réalisés autour de l'analyse et de la fouille des flux de données sera présenté dans les deux chapitres qui suivent.

Chapitre 2

Panorama sur les méthodes d'analyse de flux de données

En raison des nombreuses applications potentielles, la recherche dans le domaine des flux de données est devenue très intense depuis le début des années 1995. De nombreux travaux scientifiques ont été menés pour apporter des solutions aux défis évoqués par l'exploitation des flux de données. Les approches généralement suivies consistent soit à construire des résumés de l'historique d'un flux de données pour se substituer aux données originales issues du flux, de sorte à pouvoir ensuite appliquer les approches classiques d'analyse et de fouille de données, soit à étendre les approches classiques ou bien à proposer de nouvelles approches appropriées au traitement de flux de données. Le but de ce chapitre est de proposer un état de l'art sur les travaux de recherche relatifs au domaine de l'analyse des flux de données. Quatre axes principaux seront abordés successivement. Dans un premier temps, nous présentons les différentes techniques de construction et de maintenance des résumés représentant l'historique de flux de données (Section 2.1). Nous procédons ensuite à la présentation des méthodes de clustering qui peuvent s'adapter au contexte de flux de données (Section 2.2). Le troisième axe est consacré aux méthodes de classification de flux de données (Section 2.3). Pour terminer, nous abordons les méthodes de détection de changements qui existent, et leur mise en place pour la surveillance des flux de données (Section 2.4).

2.1 Structures de synthèse de flux de données

Devant le volume en constante augmentation des données à traiter, il devient difficile et coûteux de les stocker dans leur totalité antérieurement à l'application des méthodes d'analyse et de fouille de données. Ainsi, seule une certaine quantité d'informations pourrait être chargée dans la mémoire de travail. Des méthodes de synthèses ou de résumés automatiques ont été développées de manière à conserver une trace de l'historique du contenu d'un flux de données, l'objectif étant d'offrir la possibilité d'effectuer des analyses sur l'historique des données. Un tel type de méthodes doit globalement vérifier de nombreuses propriétés, à savoir :

- Les résumés doivent être construits à la volée à partir d'un ou plusieurs flux de données. Par conséquent, le temps nécessaire à la construction incrémentale de résumés doit être compatible avec le débit d'arrivée des données.
- Les résumés doivent couvrir uniformément le contenu des flux de données afin de fournir l'information nécessaire à la description appropriée de toutes les parties d'un flux de données.

- Malgré la continuité des flux de données, les résumés qu'on en tire doivent être de taille fixe ou lentement variable de manière à ne pas dépasser la capacité de stockage limitée.

De nombreuses structures de résumés ont été développées ces dernières années comme l'échantillonnage, les sketches, les ondelettes et les histogrammes. Toutes ces différentes structures sont des solutions approchées veillant à trouver un équilibre entre l'efficacité et la précision des résultats. Elles ne nécessitent souvent qu'un seul passage sur les données ce qui les rend particulièrement utiles dans le cas des flux de données. Cependant, la question de synthèse de flux de données est assez délicate et encore relativement ouverte. Par la suite, nous présentons les différentes techniques existantes pour la construction et la maintenance de résumés de flux de données en précisant les avantages et les inconvénients qui en dérivent.

2.1.1 Échantillonnage

L'échantillonnage est une technique bien établie dans des domaines aussi nombreux que variés. Il s'agit de se limiter à un sous-ensemble des données plutôt que leur intégralité, sans perdre trop d'information. Ce sous-ensemble est appelé échantillon. L'avantage de l'échantillonnage, par rapport aux autres techniques de synthèse telles que les histogrammes, les ondelettes, et les sketches, est qu'il conserve la représentation originale des données. Ce qui le rend très facile à utiliser avec tous les types des données et tout spécialement les données multidimensionnelles. En pratique, l'échantillonnage n'est cependant envisageable qu'à condition, d'une part, de pouvoir contrôler la représentativité de l'échantillon, et d'autre part, de ne pas s'intéresser aux informations les plus rares. L'échantillonnage des flux de données s'appuie principalement sur les méthodes d'échantillonnage traditionnelles. Toutefois, des mesures particulières ont été prises, en réponse aux problèmes découlant de l'indisponibilité de la totalité des données et du fait que l'on ne connaît pas à l'avance la taille du flux.

L'échantillonnage par réservoir est l'une des méthodes traditionnelles qui est bien adaptée au cas des flux de données (Vitter, 1985). Il permet de maintenir en ligne un échantillon uniforme aléatoire de taille fixe, k, et ne nécessite pas de connaître a priori la taille des flux. L'idée est simple : les premières k données sont stockées dans un réservoir de taille k lors de l'étape d'initialisation. Lorsque la (t+1)-ème donnée arrive, elle remplace aléatoirement l'une des données dans le réservoir avec une probabilité de $k/t+1$. Ainsi, la probabilité d'inclusion se réduit avec le temps. Bien évidement, ceci est en contradiction avec le raisonnement usuel dans le contexte des flux de données consistant à accorder plus d'importance aux données récentes. Pour pallier ce problème, deux solutions ont été envisagées : la première repose sur l'emploi des techniques de fenêtrage (Babcock et al., 2002) et la second solution consiste à appliquer des fonctions de biais exponentielles afin de réguler l'échantillon (Aggarwal, 2006).

Les techniques de fenêtrage permettent d'insérer régulièrement de nouvelles données dans le réservoir mais trouvent leurs limites à l'expiration des données dans une fenêtre glissante. En effet, les données qui ne font plus partie de la fenêtre deviennent invalides, et si elles appartiennent à l'échantillon, il faut les remplacer par des données aléatoirement choisies à partir de la fenêtre courante. Toutefois, dans les modèles de flux de données il n'y a en principe pas d'accès aux données passées. Une solution simple consisterait à stocker toutes les données de la fenêtre courante mais cela exigerait $O(n)$ d'espace mémoire. Dans (Babcock et al., 2002), ce problème a été abordé avec deux approches : "échantillonnage avec réserve" et "échantillonnage en chaîne" :

- L'échantillonnage avec réserve place les données arrivées dans une réserve avec une proba-

bilité $2cklog(n)/n$; où n est la taille de la fenêtre ($n \gg k$) et $c > 0$. Babcock et al. (2002) montrent que la taille de la réserve est entre k et $4cklog(n)$. Lorsqu'une donnée expire, elle est retirée de la réserve. Un échantillonnage aléatoire est ensuite effectué à partir des données de la réserve afin de générer un échantillon de taille k.

- L'échantillonnage en chaîne optimise l'espace occupé par l'approche précédente en construisant k chaînes dont les têtes forment l'échantillon. Son principe est d'ajouter chaque nouvelle donnée i à l'échantillon avec une probabilité $1/Min(i, n)$. À chaque fois qu'une donnée i est ajoutée, un indice correspondant à une autre donnée qui remplacera la donnée i dès qu'elle expire est choisi aléatoirement dans l'intervalle d'indices $[i + 1, i + n]$. La donnée choisie est stockée à son arrivée, et la donnée qui la replacera est également choisie, et ainsi de suite.

Avec le même objectif consistant à contrôler l'expiration des données dans un échantillon réservoir, (Aggarwal, 2006) propose l'utilisation des fonctions de biais temporelles afin de moduler l'échantillon réservoir de manière à se focaliser sur des données plus ou moins récentes en fonction des contraintes de l'application. La fonction de biais est liée à la probabilité $p(r, t)$ qu'une donnée introduite dans le réservoir à l'instant r soit encore présente à l'instant t, avec $r \leq t$. Plus spécifiquement, la probabilité $p(r, t)$ est proportionnelle à la fonction de biais $f(r, t)$, donnée par $f(r, t) = e^{-\lambda(t-r)}$ où $\lambda \in [0, 1]$ représente le taux de biais. En outre, l'introduction de cette classe spéciale des fonctions exponentielles rend possible, d'une part, la définition d'une borne supérieure sur la taille du réservoir qui est indépendante de la taille du flux ($R(t) \leq \frac{1}{\lambda}$), et d'autre part, l'utilisation d'algorithmes de remplacement simples.

L'efficacité des méthodes de l'échantillonnage par réservoir pourrait encore être améliorée en utilisant d'autres techniques comme l'échantillonnage concis et l'échantillonnage compteur (Gibbons et Matias, 1998). Ces techniques sont particulièrement utiles lorsque l'on s'intéresse aux fréquences d'occurrence des données dans le flux (néanmoins, elles sont moins adaptées au cas des données multidimensionnelles). L'échantillonnage concis stocke toutes les données apparaissant plus qu'une fois sous la forme des paires (valeur, compteur) formées des données et leur fréquences de répétition dans l'échantillon. De cette façon, un échantillon concis de taille m se rapportera à un échantillon de taille $m' \geq m$. Donc, l'échantillonnage concis ne serait jamais moins précis que l'échantillonnage classique. Cependant, il présente l'inconvénient de nécessiter plus de temps pour effacer une donnée de l'échantillon en cas de débordement (une période de répétition jusqu'à ce qu'au moins une (valeur, compteur) devienne un singleton ou qu'un singleton soit éliminé). L'échantillonnage compteur est une variante de l'échantillonnage concis consistant à garder en plus les fréquences exactes des données à partir du moment où l'on les a ajoutées à l'échantillon.

2.1.2 Histogrammes, Sketches, et Ondelettes

Histogrammes

L'histogramme est un modèle fréquemment utilisé pour résumer des données sous la forme d'une représentation graphique qui montre la répartition des données. Il divise les données en paquets contigus (intervalles). Pour chaque paquet, un rectangle est construit dont la largeur représente la gamme de valeurs recouverte par le paquet qu'il représente, alors que la hauteur du rectangle indique le nombre de données appartenant à ce paquet. Selon la règle de division utilisée, la largeur et la hauteur des paquets peuvent varier. Une forme particulière très utilisée

des histogrammes est dite "équi-réparti", dans laquelle les paquets sont de largeur (ou hauteur) égale. Bien que simple, ce type d'histogrammes ne saurait représenter la répartition des données de manière bien précise. Des meilleurs modèles sont les histogrammes optimaux (Jagadish et al., 1998). L'idée est de choisir la largeur des paquets de façon à minimiser la variance des fréquences dans chacun des paquets, ce qui implique la minimisation de l'erreur de répartition.

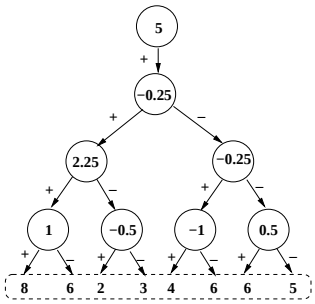
Les histogrammes classiques étant des modèles statiques ne pouvant pas être mis à jour incrémentalement, des techniques alternatives ont été élaborées pour les maintenir à jour. L'approche de Gibbons et al. (2002) préserve un échantillon réservoir des données à partir duquel des histogrammes approximatifs sont recalculés si nécessaire. Deux types d'histogrammes sont considérés : histogrammes de hauteur égale (un nombre égal de données dans chacun des paquets) et histogrammes comprimés (les N données les plus fréquentes sont stockées dans des paquets singletons et un histogramme équi-hauteur est utilisé pour traiter les autres données). Les deux types d'histogrammes maintiennent la plus grande valeur et la fréquence des données pour chaque paquet. Des techniques de division/fusion sont utilisées pour réduire l'accès au réservoir. Par exemple, dans le cas des histogrammes équi-hauteurs, une borne supérieure T_S et une borne inférieure T_I sur la hauteur des paquets sont fixées. Tant que la limite T_S n'est pas atteinte, l'insertion de nouvelles données fait seulement augmenter la fréquence des données dans le paquet concerné, mais au moment où la limite est dépassée, au lieu de recalculer l'histogramme, le paquet saturé est divisé en deux alors que deux paquets dont la fréquence totale est inférieure à T_S sont fusionnés afin de maintenir le nombre de paquets fixe. L'histogramme est recalculé à chaque fois qu'une fusion n'est pas possible. De façon analogue, la suppression d'une donnée fait diminuer la fréquence des données dans le paquet concerné tant qu'elle est supérieure à la borne T_I. Dans le cas contraire, deux paquets sont fusionnés et un paquet dont la fréquence est la plus grande est divisé en deux.

Un algorithme intéressant utilisant la programmation dynamique pour construire les histogrammes V-optimaux (H_{opt}) a été discuté dans (Jagadish et al., 1998). Cet algorithme a cependant l'inconvénient d'être très coûteux en temps et en espace mémoire. Afin de surmonter cette complexité, une méthode plus efficace a été envisagée dans (Guha et al., 2006) permettant la construction d'histogrammes optimaux avec un rapport d'approximation $(1+\epsilon)$, c.-à-d., la méthode approxime un flux de données S par un histogramme H tel que $\|S-H\|^2 \leq (1+\epsilon)\|S-H_{opt}\|^2$. D'autres algorithmes pour la construction d'histogrammes peuvent être trouvés dans (Gilbert et al., 2002) et (Matias et al., 2000).

Ondelettes

Les techniques à base d'ondelettes (Wavelets) peuvent être vues comme une transformation mathématique réversible qui décompose hiérarchiquement un ensemble de données en composants multi-résolutions appelés "coefficients d'ondelettes" capturant toutes les caractéristiques des données. La compression des données est généralement réalisée en omettant certains des coefficients sur la base d'un mécanisme de seuillage et/ou d'un critère d'erreur à minimiser. Nous nous limitons ici à la présentation des ondelettes de Haar, les plus simples et les premières à avoir été utilisées dans une large variété de problèmes y compris les flux de données. Pour simplifier, nous discutons la version basique des ondelettes de Haar appliquée aux données à une dimension. Les ondelettes unidimensionnelles peuvent souvent servir de base à l'élaboration d'ondelettes en dimension plus élevée (Stollnitz et al., 1996).

Résolution (niveau de détails)	Moyenne (μ values)	Coefficients de détail (ψ values)
k=4	(8, 6, 2, 3, 4, 6, 6, 5)	—
k=3	(7, 2, 5, 5, 5, 5)	(1, -0.5, -1, 0.5)
k=2	(4.75, 5.25)	(2.25, -0.25)
k=1	(5)	(-0.25)



TAB. 2.1 – Un exemple de calcul des coefficients d'ondelettes de Haar et l'arbre d'erreur correspondant. données originales : S=8, 6, 2, 3, 4, 6, 6, 5, données transformées : W_S=5, -0.25, 2.25, -0.25, 1, -0.5, -1, 0.5. D'après (Aggarwal, 2007).

Soit un flux de n données, une décomposition sur la base d'ondelettes de Haar définit 2^{K-1} coefficients à un niveau de détail k ; chacun de ces coefficients correspond à un segment contigu du flux d'une longueur $n/2^{k-1}$. Par exemple, le i-ème de ces 2^{K-1} coefficients correspond au segment de $(i-1)n/2^{k-1}+1$ à $in/2^{k-1}$. Notons ce coefficient par ψ_i^k et notons aussi la moyenne des données du segment correspondant par μ_i^k. La transformée de Haar utilise deux fonctions, μ et ψ, reliés par les formules suivantes :

$$\psi_i^k = \frac{\mu_{2i-1}^{k+1} - \mu_{2i}^{k+1}}{2} \tag{2.1}$$

Ainsi, le passage d'un niveau à l'autre s'effectue récursivement en remplaçant une paire de données par le résultat des fonctions μ et ψ jusqu'à atteindre le niveau 1 correspondant à la moyenne globale des données μ_1^1. La transformée de Haar est formée de la moyenne globale des données suivie par les coefficients des niveaux 1 à $log_2(n)$[7]. La décomposition en ondelettes peut aussi être représentée sous la forme d'un arbre binaire, appelé l'arbre d'erreur (cf. Tableau 2.1). L'avantage de la transformée de Haar par rapport aux données originales est que la plupart des coefficients tendent à avoir des valeurs absolues très faibles spécialement lorsque les données originales ont des valeurs semblables. Ainsi, l'exclusion de ces coefficients n'engendrait que de petites erreurs lors de la reconstruction des données originales, ce qui rend cette transformée très efficace pour la compression de données.

Comme il est possible de le constater, ce mode de fonctionnement ne s'applique pas directement au cas des flux de données où le calcul des coefficients d'ondelettes ainsi que la sélection des coefficients à retenir doivent se faire de manière incrémentale. Quelques travaux ont donc été menés pour répondre à ces exigences fonctionnelles. Ainsi, (Karras et Mamoulis, 2005) propose de construire un arbre à un nombre limité B de coefficients. À l'arrivée de chaque paire de données, une paire de coefficients est tirée de l'arbre. La paire à tirer est choisie de manière à ce que l'erreur engendrée soit aussi petite que possible. Pendant ce processus, une structure auxiliaire consistant en un nœud pour chaque niveau de l'arbre d'erreur est utilisée afin de tenir

[7]Notez que, intuitivement, les coefficients d'ondelettes n'ont pas le même poids d'importance lors de la reconstruction des valeurs des données originales. Par exemple, la moyenne globale est évidemment plus importante que les autres coefficients puisqu'elle affecte la reconstruction de toutes les valeurs. Afin d'égaliser l'importance des coefficients d'ondelettes, ils sont au préalable normalisés en les divisant par $\sqrt{2^k}$. Ainsi, on attribue une priorité selon le niveau de résolution : les coefficients correspondant aux hautes résolutions auront une importance plus faible que ceux des basses résolutions.

compte des nœuds suspendus dans l'arbre de coefficients (de tels nœuds apparaissent lorsque le nombre de données arrivant n'est pas égal à une puissance de deux). À chaque élimination d'un coefficient de l'arbre d'erreur, les relations entre les nœuds sont mises à jour ce qui implique qu'un nœud peut avoir plusieurs descendants directs à différents niveaux de l'arbre. Notons, cependant, que les heuristiques proposées pour la propagation d'erreur suite à chaque élimination dans l'arbre sont limitées au cas des données unidimensionnelles. L'extension au cas des données multidimensionnelles semble être difficile à réaliser du point de vue de la complexité et des coûts.

Sketches

Les sketches sont des résumés souvent destinés à une tâche précise, comme en particulier l'estimation de normes L_p, $\|a\|_p = (\sum_i |a_i|^p)^{\frac{1}{p}}$, $(p \geq 1)$. L'idée des sketches est essentiellement une extension de la technique de projection aléatoire au domaine des séries temporelles, l'objectif étant de déterminer des tendances représentatives dans les séries pouvant ensuite être utilisées pour une analyse rapide et approximative de celles-ci (Johnson et Lindenstrauss, 1984; Indyk et al., 2000).

La projection aléatoire consiste simplement à projeter un ensemble des vecteurs de grande dimension dans un espace aléatoire de dimension réduite en préservant approximativement les distances euclidiennes (L_2) entre vecteurs. Plus formellement, soit un vecteur $\overrightarrow{t} = t[1..l]$, qui peut être assimilé à une série temporelle de taille l, son vecteur sketch $\overrightarrow{s(t)}$ de taille k peut être formé en constituant le produit scalaire de $\overrightarrow{t}$ et des k vecteurs aléatoires $\overrightarrow{v}_1, \ldots, \overrightarrow{v}_k$ de norme 1 (les composantes des vecteurs $\overrightarrow{v}_i$ sont tirées aléatoirement suivant une loi normale de moyenne 0 et de variance 1). Ainsi, la composante i de vecteur sketch $\overrightarrow{s(t)}$ est donné par :

$$s(t)[i] = \overrightarrow{t}.\overrightarrow{v}_i = \sum_{j=1}^{l} t[j].v_i[j] \tag{2.2}$$

Comme il a été démontré dans (Johnson et Lindenstrauss, 1984), des bornes de l'erreur sur l'approximation de la distance L_2 sont préservées[8]. Des méthodes efficaces pour le calcul des sketches sur des fenêtres de taille fixe et de taille variable ont été développées dans (Indyk et al., 2000).

2.1.3 Maintenance en ligne des micro-clusters

Une autre variante des techniques de synthèse de flux de données consiste à appliquer un algorithme de clustering mettant en jeu un grand nombre de clusters. Des statistiques simples sur les clusters — qui peuvent être appelés dans ce cas des micro-clusters — sont ensuite calculées et mémorisées afin de constituer des représentations approximatives des données associées aux clusters. Par rapport aux autres techniques, celle-ci présente l'intérêt de pouvoir s'appliquer au flux de données multidimensionnelles. Nous présentons ci-dessous deux des principaux algorithmes élaborés pour la maintenance en ligne des micro-clusters, à savoir les algorithmes BIRCH et CluStream. Nous mentionnons également quelques extensions de ces algorithmes.

[8]Soit L un ensemble des vecteurs de taille l, pour $\epsilon < 1/2$, if $k = \frac{9log|L|}{\epsilon}$, quels que soient les vecteurs $\overrightarrow{u}$, $\overrightarrow{w}$ de L, si l'on note leurs vecteurs sketches par $\overrightarrow{S}(u)$ et $\overrightarrow{S}(w)$, respectivement, on a toujours :

$$(1-\epsilon)\|\overrightarrow{u} - \overrightarrow{w}\|^2 \leq \|\overrightarrow{S}(u) - \overrightarrow{S}(w)\|^2 \leq (1+\epsilon)\|\overrightarrow{u} - \overrightarrow{w}\|^2$$

avec une probabilité de 1/2.

Algorithme BIRCH

BIRCH (acronyme de "Balanced Iterative Reducing and Clustering using Hierarchies" en anglais) est un algorithme de clustering hiérarchique dû à (Zhang et al., 1996, 1997). Une des particularités de BRICH est de pouvoir s'adapter à de très grandes bases de données, ceci en deux étapes successives : Tout d'abord, il produit des résumés compacts des données originales et utilise une structure arborescente (CF-Tree) pour les gérer. Les feuilles de l'arbre correspondent à de petits paquets de données, interprétés comme des micro-clusters. Chacun de ces micro-clusters est représenté par un CF-vecteur pouvant être mis à jour incrémentalement. En principe, cette étape ne nécessite qu'un seul balayage de la base de données. Cependant, des passages supplémentaires sur les données permettent d'améliorer considérablement la performance de l'algorithme. À l'issue de l'étape de création des CF-vecteurs, un algorithme de clustering hiérarchique est appliqué à l'ensemble des micro-clusters au lieu des données originales pour obtenir une partition définitive des données avec un petit nombre de clusters. Nous détaillerons ci-après le fonctionnement de l'algorithme.

L'algorithme BIRCH introduit le concept de CF-vecteur "Cluster Feature Vecteur" pour représenter les clusters. Soit un ensemble de N données associées à un cluster $\overline{X} = \{\overline{X}_1, \ldots, \overline{X}_N\} \in \mathbb{R}^n$, un CF-vecteur est défini par un triplet $(N, \overline{LS}, SS)$ dans lequel N est le nombre de données dans le micro-cluster ; $\overline{LS}$ est la somme linéaire des N données ($\overline{LS} = \sum_{i=1}^{N} \overline{X}_i$) ; et SS est la somme des carrés des N données ($SS = \sum_{i=1}^{N} \overline{X}_i^2$). À partir de cette définition de CF-vecteurs, les mesures nécessaires pour assurer les opérations de clustering dans BRICH sont aisément calculées. À titre d'exemple, le centre de gravité (vecteur centroïde) et le rayon (déviation standard) d'un cluster sont respectivement donnés par les expressions suivantes :

$$\overline{M}_c = \frac{1}{N} \sum_{i=1}^{N} \overline{X}_i = \frac{\overline{LS}}{N}$$

$$R_c = \left(\sum_{i=1}^{N} \frac{(\overline{X}_i - \overline{M}_c)^2}{N} \right)^{\frac{1}{2}} = \left(\frac{\sum_{i=1}^{N} \overline{X}_i^2}{N} - \frac{2\overline{M}_c \sum_{i=1}^{N} \overline{X}_i}{N} + \frac{\sum_{i=1}^{N} \overline{M}_c^2}{N} \right)^{\frac{1}{2}} = \left(\frac{SS}{N} - \left(\frac{\overline{LS}}{N} \right)^2 \right)^{\frac{1}{2}}$$

En outre, les CF-vecteurs possèdent une propriété d'additivité qui permet de déduire facilement le CF-vecteur résultant de la fusion de deux micro-clusters. Autrement dit, si deux micro-clusters ayant des CF-vecteurs CF_1 et CF_2 fusionnent, le CF-vecteur du micro-cluster résultant est :

$$CF_1 + CF_2 = (N_1 + N_2, \overline{LS}_1 + \overline{LS}_2, SS_1 + SS_2)$$

Un arbre hiérarchique de CF-vecteurs est utilisé pour gérer et ajuster incrémentalement la qualité des micro-clusters. Les nœuds de l'arbre ont une capacité de stockage limitée et fixée a priori (typiquement elle correspond à la taille d'une page du disque). Chaque nœud contient un ensemble des entrées sous la forme $[CF_i, fils_i]$, avec i= 1, 2, ..., B, et $fils_i$ un pointeur vers son i-ème nœud fils. La construction de l'arbre nécessite de spécifier auparavant 3 paramètres : B le nombre maximal des entrées dans un nœud interne ; L le nombre maximal des entrées dans un nœud feuille ; et T le seuil d'absorption. L'arbre est construit en partant d'un arbre initialement vide et en insérant successivement les données d'une manière similaire à la construction d'un arbre B+. L'insertion d'une donnée dans l'arbre peut être décrite de la manière suivante :

1. Recherche d'un nœud feuille approprié en partant de la racine et en choisissant le nœud le plus proche en termes d'un critère de similarité.

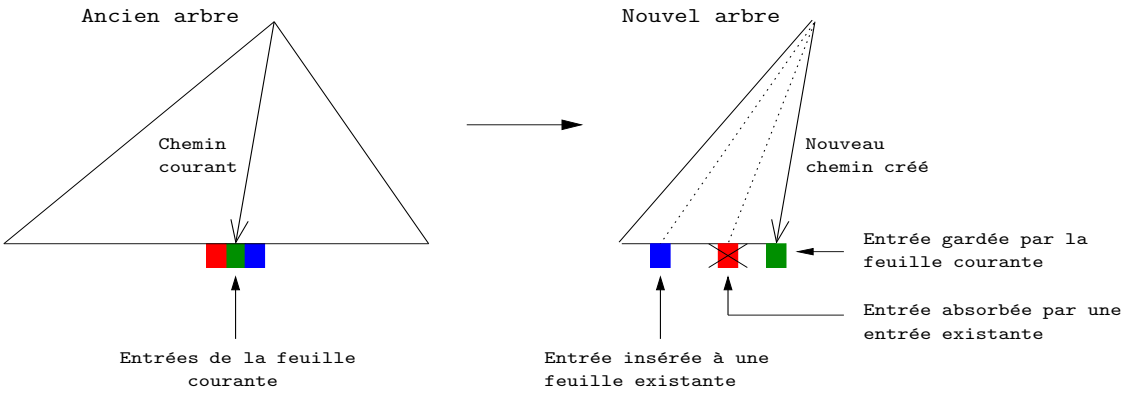


FIG. 2.1 – Illustraction de la reconstruction de l'arbre des CF-vecteurs, adapté de (Zhang et al., 1997)

2. La donnée est soit absorbée par l'entrée la plus proche dans ce nœud feuille si la condition d'absorption est satisfaite (le rayon du cluster résultant de la fusion de cette entrée et la donnée en question ne dépasse pas le seuil T) ou bien entraîne la création d'une nouvelle entrée dans ce même nœud feuille.
3. Lorsqu'un nœud est saturé, il est partitionné en deux en choisissant les deux entrées les plus éloignées comme centres et en regroupant les autres entrées (y compris la nouvelle donnée) autour de ces centres en utilisant un critère de similarité. Une nouvelle entrée est aussi créée dans le nœud parent pour jouer le rôle de parent de la nouvelle feuille que l'on vient de créer. Le dépassement de la taille limitée des nœuds parents entraîne aussi des fragmentations qui peuvent être propagées jusqu'à la racine de l'arbre. Si le fragmentation atteint la racine, la hauteur de l'arbre augmente de 1.

Bien évidemment, l'intégration continue de nouvelles données dans l'arbre ne peut que conduire à une saturation de l'espace disque dédié au stockage de l'arbre. Dans un tel cas, il s'agira alors de reconstruire l'arbre en augmentant le seuil d'absorption T. La reconstruction s'effectue en copiant l'ancien arbre chemin par chemin (à partir de la racine vers une feuille) vers un nouvel arbre (initialement vide). Ainsi, le processus s'applique récursivement en commençant par le chemin le plus à droite et en essayant soit de faire absorber les entrées des feuilles copiées par des entrées des feuilles déjà créées dans l'arbre reconstruit en utilisant la nouvelle valeur du seuil T, ou bien de les ajouter comme des nouvelles entrées dans les feuilles existantes. Dans le cas nécessaire, un nouveau chemin est créé dans le nouvel arbre pour garder les entrées/feuilles restantes. La figure 2.1 illustre l'idée de base de la reconstruction de l'arbre.

Il est clair qu'une grande valeur de T entraîne la création d'un faible nombre de entrées/feuilles et ainsi d'un arbre de petite taille. Cependant, la qualité de la représentation des données risque de ne pas être suffisante. Par conséquent, il est indispensable de trouver une valeur initiale appropriée de T pour réduire le nombre de reconstruction à effectuer, et de savoir ensuite augmenter adéquatement cette valeur pour chaque reconstruction, l'objectif étant de réduire la taille globale de l'arbre. Les auteurs de l'algorithme (Zhang et al., 1996) proposent d'utiliser une valeur initiale nulle en l'absence de connaissances a priori sur la distribution des données. Ils proposent également une heuristique acceptable pour l'augmentation dynamique de T. Il s'agit de rechercher dans les feuilles les deux entrées les plus proches et d'adapter le seuil de telle façon que ces deux entrées puissent fusionner lors de la reconstruction suivante. Ceci permet de libérer approxima-

tivement la moitié de l'espace mémoire pour accueillir les données suivantes.

La taille des micro-clusters étant limitée par les paramètres spécifiques de l'algorithme, ces micro-clusters ne peuvent qu'être considérés comme des résumés des données initiales sous la forme des CF-vecteurs. Ainsi, un algorithme global de clustering est appliqué dans une phase ultérieure à l'ensemble des micro-clusters pour trouver les clusters finaux. BIRCH adopte un algorithme hiérarchique ascendant mais d'autres algorithmes peuvent aussi être utilisés.

Comme l'on peut constater, cet algorithme est destiné à traiter les données numériques ; cependant, il existe une extension de cet algorithme pour le traitement des données catégorielles (Chiu et al., 2001). Il reste à noter que BIRCH est l'un des meilleurs algorithmes pour la classification de grosses bases de données. Néanmoins, il est assez sensible au choix de paramètres et à l'ordre de présentation des données. De plus, il a une mauvaise performance avec des clusters non-sphériques et des clusters ayant des tailles très différentes (Guha et al., 1998).

Algorithme CluStream

CluStream de Aggarwal et al. (2003) est relativement proche de l'algorithme BRICH. Néanmoins, il s'en distingue principalement par la prise en compte de fenêtres temporelles. CluStream est basé sur l'utilisation des micro-clusters faisant intervenir des informations temporelles supplémentaires (le timestamp des données), dont les CF-vecteurs sont définis comme suit : Soit un ensemble de N données $\{\overline{X}_1, \ldots, \overline{X}_N\} \in \mathbb{R}^n$ associées à un micro-cluster et dont les timestamps sont $T_1, \ldots, T_N$, le CF-vecteur du cluster en question est défini par un $(2n + 3)$ uplet $(\overline{CF2^x}, \overline{CF1^x}, CF2^t, CF1^t, N)$ dans lequel $\overline{CF2^x}$ est la somme des carrés de toutes les variables des données ; $\overline{CF1^x}$ est la somme linéaire des données ; $CF2^t$ stocke la somme des carrés des timestamps ; $CF1^t$ stocke la somme des timestamps ; et N est le nombre de données.

CluStream maintient un nombre stable de micro-clusters q, déterminé selon la taille de mémoire centrale disponible pour le stockage des micro-clusters. Chaque micro-cluster M_i reçoit un identificateur unique ID lors de sa première création. Si deux micro-clusters sont fusionnés, une liste des identificateurs IDs est établie pour identifier les micro-clusters constitués. La création des q micro-clusters initiaux est réalisée hors ligne en utilisant un algorithme de type k-means opéré sur les N premiers points des données arrivant au début du processus de calcul sur le flux de données. Ensuite, le processus de maintenance des micro-clusters est lancé en ligne : Les micro-clusters sont mis à jour dès l'arrivée d'une nouvelle donnée $\overline{X_k}$ de façon à s'adapter aux changements survenus dans le flux au fil du temps. Cela se fait en identifiant le micro-cluster M_p le plus proche de $\overline{X_k}$. La donnée $\overline{X_k}$ est absorbée par M_p si elle ne dépasse pas la frontière maximale du micro-cluster ; la frontière maximale d'un micro-cluster est définie en termes de déviation moyenne des données attachées au cluster. Si la donnée $\overline{X_k}$ se trouve au-delà de la frontière maximale de M_p, un nouveau micro-cluster est créé. Cependant, avant de créer le nouveau micro-cluster, il faut réduire de 1 le nombre de micro-clusters existants. Pour cela, un des micro-clusters est détruit s'il est possible de le reconnaître comme aberrant (c.-à-d. un micro-cluster trop vieux en termes d'un critère basé sur la date d'arrivée de ses derniers éléments), sinon deux des plus proches micro-clusters doivent être fusionnés en un seul. Bien évidemment, la liste d'ID des micro-clusters est mise à jour lors de chaque création/fusion.

Dans ce système, un archivage périodique des micro-clusters dans des clichés "snapshots" est effectué aux intervalles fixés via une structure pyramidale du temps, afin d'atteindre l'objectif

de pouvoir analyser l'évolution au cours du temps. Chaque cliché contient les CF-vecteurs des micro-clusters, leur liste d'ID et le temps du stockage. Du fait que l'ensemble des micro-clusters sauvegardés à chaque étape de l'algorithme est basé sur l'historique entière du flux traité depuis le début du processus de calcul, quand on s'intéresse à un horizon temporel $[t_c\text{-}h,\ t_c]$ de taille h sur le flux il faut en premier lieu identifier les micro-clusters spécifiques à cet horizon de temps. Ceci est fait en utilisant la propriété de soustractivité des CF-vecteurs pour supprimer les micro-clusters dont les identificateurs apparaissent simultanément dans les listes d'ID des micro-clusters des moments t_c et t_c-h. L'analyse de l'évolution est alors faite à l'aide des listes d'identification des micro-clusters. Par exemple, les nouveaux clusters au moment t_c par rapport au moment t_c-h sont les micro-clusters dont les identificateurs n'apparaissent pas dans la liste d'ID des micro-clusters du moment t_c-h. À présent, les algorithmes classiques peuvent aussi être appliqués aux micro-clusters sans être contraints par l'exigence d'un seul passage. Spécialement, les micro-clusters sont souvent classés (en utilisant une variante de la méthode des k-means) dans un niveau supérieur pour créer un nombre plus petit de clusters, appelés macro-clusters.

De nombreuses variantes de CluStream ont été proposées par la suite. En particulier dans (Aggarwal et al., 2004), HPStream est une extension de CluStream pour le traitement des données de très haute dimension. Elle a pour but de découvrir des micro-clusters dans des différents sous-espaces de dimensions inférieures à la dimension de l'espace original des données. Cela atténue, dans une certaine mesure, les problèmes liés aux espaces de haute dimension, comme la dispersion des données. Par ailleurs dans (Yang et Zhou, 2006), une extension de CluStream à des données hétérogènes — numériques et catégorielles — a été développée, donnant lieu à une modification de la structure des CF-vecteurs, des mesures de distance et de l'algorithme du clustering k-means.

2.1.4 Synthèse

La construction de résumés de flux de données est souvent une étape clé dans le développement des systèmes orientés flux de données. Elle permet de conserver une trace de l'historique d'un flux de données après son passage de façon à pouvoir réaliser a posteriori des traitements sur le flux proprement dit. Bien que la construction de résumés reste de loin le sujet le plus étudié dans le domaine des flux de données, les techniques disponibles sont souvent proposées dans le but de réaliser une tâche précise sur un flux de données particulier. Par ailleurs, la plupart de ces techniques ne sont que des extensions d'approches qui existaient déjà dans le cadre du traitement des données statiques. Ainsi que nous l'avons examiné tout au long de cette section, on peut différencier deux grands types d'approches pour la construction de résumés. Le premier ne fait pas usage de la dimension temporelle des flux de données. Il vise à conserver en mémoire une partie des informations contenues dans le flux sur une période donnée en tenant compte des contraintes liées au traitement en ligne des flux de données. Tandis que le second type d'approche vise à construire des résumés couvrant l'intégralité du flux et non pas une simple fenêtre. Cela se fait surtout par un traitement efficace de la dimension temporelle à l'aide des techniques de clichés et de fenêtres inclinées. Il est cependant à noter que, dans les deux cas, la plupart des techniques existantes ne répondent que partiellement aux contraintes propres aux flux de données. De fait, si toutes les techniques que nous avons exposées ici sont soumises à la contrainte de fonctionnement en ligne, les contraintes de ressources (espace mémoire, temps de calcul) sont moins bien prises en charge. Par exemple, l'algorithme CluStream, bien que potentiellement intéressant, n'est pas adapté au cas des flux à très haut débit du fait que l'ajout de chaque nouvelle donnée peut entraîner des calculs de distance avec tous les micro-clusters, voir si deux micro-clusters doivent être fusionnés, etc. À tout ceci s'ajoute encore le problème de la gestion relativement lourde des

clichés pour l'intégration de la composante temporelle des données.

De l'ensemble des techniques passées en revue, il apparaît que les méthodes d'échantillonnage (cf. Section 2.1.1) et de clustering (cf. Section 2.1.3) sont les plus adaptées au cas des flux de données multidimensionnelles. Toutefois, les méthodes d'échantillonnage peuvent entraîner des pertes d'informations plus importantes que les méthodes de clustering, notamment en ce qui concerne les informations relativement rares dans les flux. De ce fait, nous considérons comme fondamental de procéder à une étape de clustering de flux de données au préalable de toute autre méthode dans le cas des flux multidimensionnels. Seulement, ce domaine étant encore peu investi par manque des méthodes de clustering adaptées aux particularités des flux de données multidimensionnelles, nous nous attachons dans la section suivante, en plus des méthodes de clustering de flux de données, aux méthodes de clustering classiques où certaines sont exploitables dans le cadre des flux de données, soit directement soit à l'aide des techniques de fenêtrage.

2.2 Clustering de flux de données

Le clustering est une tâche entièrement automatique consistant à déterminer les classes qui peuvent au mieux décrire un ensemble de données, sans aucune connaissance a priori. La complexité de cette tâche s'accroît fortement lorsqu'il s'agit de clustering de flux de données. Les défis sont liés aux nombreux aspects et contraintes opérationnelles qu'il faut considérer en parallèle. Il s'agit principalement de la complexité des calculs en termes de temps et d'espace, d'où le besoin de traiter efficacement les données dès qu'elles se présentent. Les méthodes de clustering avec passages multiples sur les données ne sont pas désirables du fait qu'elles pourraient entraîner des retards intolérables. De plus, les méthodes doivent être adaptatives de manière à pouvoir suivre l'évolution des données avec le temps. En général, ces défis sont relevés par deux types de méthodes : les méthodes scalables et les méthodes adaptatives.

Les méthodes scalables répondent au problème du passage à l'échelle lors du clustering de très grosses bases de données. En principe, ces méthodes peuvent être directement appliquées au clustering de flux de données, l'objectif étant de capturer les caractéristiques inhérentes aux données sur une longue période de temps. Toutefois, la plupart de ces méthodes ne sont pas adaptatives dans le sens où le but est de suivre les changements dans les caractéristiques des données survenant au fil du temps. Cela a très souvent conduit à une modification des méthodes scalables existantes, pour les rendre adaptatives.

Les méthodes orientées flux de données et les méthodes de clustering en-ligne ou incrémentales sont aussi compatibles dans la mesure où elles requièrent de prendre des décisions avant que toutes les données ne soient disponibles. Néanmoins, les algorithmes ne sont pas identiques dans le sens où une méthode en ligne peut avoir accès aux i premiers points de données (et de ses i décisions antérieures) lorsqu'elle traite le $(i+1)$ème point, alors que la quantité de mémoire disponible à une méthode de flux de données est limitée. En outre, contrairement à une méthode en ligne, une méthode de flux de données n'a pas l'obligation de prendre une décision après l'arrivée de chaque point de données ; mais elle peut être autorisée à prendre des décisions après qu'un ensemble de données sont arrivées. C'est pour cette raison que les méthodes orientées flux de données sont très souvent connues sous le nom de "méthodes à une seule passe". Précisions à nouveau que la sortie de ce type de méthodes est restreinte à une description simple du contenu des clusters (par exemple les centres des clusters) ; les données qui y appartiennent ne peuvent

être sauvegardées.

Dans la suite de cette section, nous présentons les principales méthodes de clustering de flux de données. Nous mettons parallèlement en évidence certains modes de fonctionnement permettant d'utiliser des méthodes de clustering classiques. Par ailleurs, nous donnons beaucoup d'importance aux méthodes connexionnistes, qui connaissent un intérêt important pour le traitement des données spatio-temporelles. De fait, bien que très rares, certaines méthodes connexionnistes peuvent apprendre en ligne de manière incrémentale. Elles présentent des avantages spécifiques liés à leur capacité de détecter des dépendances temporelles entre données, de s'adapter avec souplesse à l'évolution et de conserver la topologie des données.

2.2.1 Méthodes de type k-means

La méthode des k-means, encore appelée méthode des centres mobiles, consiste à construire une partition en k clusters à partir d'un ensemble de n données de telle sorte que l'erreur quadratique $E = \sum_{r=1}^{k} \sum_{x_i \in C_r} d^2(x_i, \mu_r)$ soit minimum (Hartigan et Wong, 1979). Dans l'expression de E, C_r sont les clusters, x_i est un point qui représente une donnée d'un cluster, μ_r est le centre de gravité (vecteur moyen) du cluster C_r, et d(.,.) est une fonction de distance (habituellement la distance euclidienne). Le paramètre d'entrée de cet algorithme est le nombre k de clusters. L'algorithme général de k-means est le suivant :

1. Choisir arbitrairement K données initiales : ils servent de centres initiaux des clusters.
2. Assigner chaque donnée x_i au cluster le plus proche.
3. Recalculer les centres de chacun des clusters.
4. Répéter les étapes 2 et 3 jusqu'à stabilité des centres (les centres ne se déplacent plus, ce qui revient également à dire que les données ne changent plus de clusters).

Il existe de nombreuses méthodes de type k-means. Elles différent suivant la sélection des k données initiales, les calculs des similarités entre données et les stratégies de calcul des centres des clusters. Par example, la méthode k-médoïdes (Kaufman et Rousseeuw, 1990) diffère de k-means par le fait qu'elle utilise les points de données les plus centraux (médoïdes) pour représenter les clusters, au lieu de prendre les centres des clusters. Grâce à cette différence, la méthode K-medioïd est moins sensible au bruit et aux données marginales que la méthode des k-means (Han, 2001).

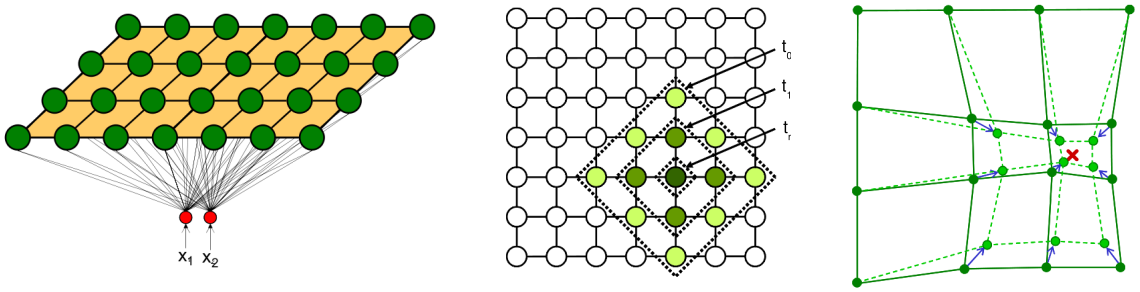
Les extensions des méthodes de type k-means au traitement des flux de données sont fondées sur l'hypothèse que les données n'arrivent pas sous la forme d'un flux continu mais plutôt d'une séquence des fenêtres temporelles définies sur celui-ci, appelées "paquets". Chaque paquet est donc un ensemble de données consécutives générées dans le flux (formé simplement en attendant l'arrivée d'un nombre suffisant de données). La détermination de la taille des paquets dépend des ressources intrinsèques du système, comme la quantité de mémoire disponible, et les exigences des applications. Cette démarche a souvent été justifiée par l'hypothèse que la prise en charge directe de l'arrivée de nouvelles données n'entraîne pas de changements majeurs dans la distribution des clusters. Plusieurs approches ont été abordées, nous en présentons les principales dans ce qui suit.

O'Callaghan et al. (2002) ont proposé l'algorithme STREAM pour trouver les clusters inhérents à un flux de données. L'idée est simple : Chaque fois qu'un nouveau paquet contenant un ensemble de données nouvellement créées est formé, un processus local de clustering "LSearch" est lancé pour choisir k points de ces données comme les centres des clusters locaux du paquet.

Chaque centre est ensuite pondéré par le nombre de données affectées au cluster correspondant. Dès que l'espace mémoire devienne insuffisant pour contenir tous les centres (ik) des clusters locaux des paquets présents, LSearch se lance à nouveau sur les centres (ik) des clusters locaux pour en conserver seulement k centres. L'algorithme LSearch est une version modifiée de k-means : au lieu d'optimiser globalement les placements de k centres en tenant un nombre fixé de clusters à chaque itération, ceux-ci sont placés selon le principe de localisation de centres "Facility Location" en autorisant un nombre important de clusters ($>k$) dans les étapes intermédiaires et en finissant par atteindre exactement le nombre k de clusters à l'issue de la dernière étape. LSearch peut aboutir à une solution bien meilleure que k-means mais nécessite un temps d'exécution plus important. Notons ici que les anciens points de données sont aussi importants que les nouveaux points, ce qui ne permet pas de détecter les changements éventuels au cours du temps.

Streaming-OSKM (Zhong, 2005b) est une combinaison d'un algorithme en ligne de type k-means (OSKM) avec un algorithme de clustering à large échelle (Farnstrom, 2000). D'après l'algorithme OSKM, les centres des clusters sont adaptés en ligne (après la présentation de chacune des données) de façon similaire à celle de l'apprentissage compétitif de type "winner-take-most". Cet algorithme s'avère être aussi efficace que k-means en termes de temps de calcul mais bien plus précis en termes de qualité de clustering (Zhong, 2005a). L'algorithme de Farnstrom (2000) a précédemment été mis au point pour faire face à de très grandes bases de données. L'idée est la suivante : Pour chaque paquet, un algorithme de clustering (k-means) est appliqué pour former k clusters à partir des données du paquet et des k vecteurs représentant des résumés de l'historique des paquets précédents. Ces k clusters servent de résumés de l'historique lors du clustering du paquet suivant. Pour rendre cet algorithme adapté au clustering de flux de données, un facteur d'oubli — qui réduit exponentiellement la contribution des historiques des anciennes données — a été introduit dans Streaming-OSKM. Il a été montré qu'un petit facteur d'oubli (ce qui veut dire oublier vite l'historique des anciennes données) conduit à de résultats meilleurs que ceux obtenus avec un facteur de 1 (c'est-à-dire lorsque tous les résumés de l'historique sont maintenus).

Dans (Domingos et Hulten, 2001), VFKM (Very Fast K-Means) adopte une approche différente de celles qui ont été décrites ci-dessus. L'objectif de cette méthode est de produire un modèle de clustering à partir d'un ensemble fini de données qui peut être considéré comme équivalent à un modèle construit à partir d'un ensemble infini de données. VFKM s'appuie sur l'établissement d'une limite supérieure de l'erreur en fonction du nombre de données considérées à chaque étape de l'algorithme. Le nombre de données est alors réduit au minimum tout en préservant toujours la limite supérieure de l'erreur. Plus précisément, la méthode construit une limite supérieure sur la distance entre les k centres de clusters générés à deux étapes séparées à partir de deux paquets de différentes tailles et en utilisant la méthode de k-means. La taille des paquets à chaque étape est choisie de manière adaptative. VFKM commence par un paquet dont la taille est déterminée selon deux paramètres spécifiés par l'utilisateur, à savoir l'erreur maximale tolérée $\varepsilon^\star$ avec une probabilité de $1-\delta^\star$. Ensuite, un nouveau paquet est formé dont la taille est déterminée en utilisant des statistiques calculées à partir du résultat de l'étape précédente (cette taille doit être au moins deux fois la taille du paquet précédent) et k-means se lance à nouveau. Les centres de clusters obtenus à l'étape courante i sont comparés à ceux de l'étape d'avant i-1 et donc une erreur ε_i avec une probabilité δ_i peut être calculée. Le processus se répète jusqu'à ce que la limite tolérée d'erreur soit satisfaite : $\varepsilon_i < \varepsilon^\star$ et $\delta_i < \delta^\star$. Notons que le processus de clustering est répété avec un nombre croissant de données : VFKM peut être coûteux en espace mémoire. Dans leur conclusion, les auteurs déclarent que l'utilisation de VFKM est



(a) (b) (c)

FIG. 2.2 – a) L'architecture d'une carte de Kohonen SOM. Tous les neurones sont connectés au vecteur d'entrée $x = (x_1, x_2)$. b) La décroissance de la taille de voisinage avec le temps. c) La modification de la topologie initiale dans la carte SOM suite à la présentation d'une donnée (×)

raisonnable d'un point de vue opérationnel quand le nombre de données se situe dans les millions.

Shah et al. (2005) ont proposé une modification simple de VFKM tenant en compte de contraintes de disponibilité de mémoire libre. Cette modification consiste à augmenter les valeurs de $\varepsilon^\star$ et $\delta^\star$ par un certain facteur, à chaque fois la mémoire libre disponible atteint un certain niveau d'épuisement afin d'accélérer la convergence de VFKM.

D'autres adaptations de k-means au traitement des flux de données, que nous ne détaillons pas ici, peuvent être trouvées dans (Bradley et al., 1998) et (Hore et al., 2007). Certaines variantes ont spécialement été introduites pour le traitement des données binaires et catégorielles ; voir, à titre d'exemple, (Ordonez, 2003; Huang, 1998; Ralambondrainy, 1995).

2.2.2 Les cartes auto-organisatrices temporelles

Les cartes auto-organisatrices de Kohonen (SOM) sont typiquement destinées à la classification topographique non supervisée des données statiques (Kohonen, 2001). L'apprentissage du réseau est de type compétitif non supervisé où les classes s'auto-organisent selon une structure de neurones sur laquelle les relations de voisinage sont prédéfinies[9] (cf. figure 2.2). Selon ce processus, lors de la présentation d'une des données d'apprentissage x_k — représentables dans le cas général sous la forme des vecteurs de dimension n — en entrée du réseau, l'on détermine le neurone gagnant $i^\star$ dont le vecteur référent $w_{i^\star}$ est le plus proche de la donnée présentée :

$$\forall\ i, \quad \|x_k - w_{i^\star}\| \leq \|x_k - w_i\|$$

Un fois que le neurone gagnant est sélectionné, les vecteurs référents associés à tous les neurones sont ajustés selon une fonction de voisinage associée au neurone gagnant ($h_{i^\star}(i,t)$, plus un neurone est proche du neurone gagnant et plus il sera influencé) :

$$\forall\ i, \quad w_i(t+1) = w_i(t) + \alpha(t)\ h_{i^\star}(i,t)\ (x_k(t) - w_i(t)) \tag{2.3}$$

[9]Dans les modèles SOM, la structure de voisinage entre les neurones peut obéir à de nombreuses variantes différentes : structure en ficelle, bidimensionnelle rectangulaire ou hexagonale ou même tridimensionnelle). La structure bidimensionnelle étant la plus utilisée, on parle le plus souvent des cartes SOM.

où $\alpha(t)$ est un terme qui contrôle la vitesse d'apprentissage, décroissant en fonction du temps et convergeant vers 0, et $h_{i^\star}(i,t)$ est souvent de type *gaussienne* ou *chapeau mexicain*. Ce type d'apprentissage nécessite de présenter de manière aléatoire plusieurs centaines de fois l'ensemble des données pour que les neurones puissent s'auto-organiser convenablement dans l'espace d'entrée. À l'issue de l'apprentissage, les données voisines dans l'espace d'entrée appartiennent à la même classe ou à des classes voisines sur la carte. Cette propriété permet de conserver au mieux la topologie de la distribution des entrées (un espace de n dimensions) lors de sa projection sur la carte (un espace bidimensionnel) et de visualiser de manière simplifiée des relations entre les différentes régions de cet espace. Les cartes SOM sont ainsi des outils très puissants pour l'analyse de données multidimensionnelles.

Plusieurs efforts ont été investis pour introduire la dimension temporelle des données dans la carte de Kohonen. Pour y arriver, certaines études se sont intéressées à utiliser de façon hiérarchiques plusieurs cartes SOM, d'autres se sont focalisées sur l'extension de cartes SOM pour l'intégration de l'information séquentielle en favorisant l'apprentissage des relations d'ordre entre les éléments d'un flux de données. Nous exposons, par la suite, quelques unes des méthodes hiérarchiques et non hiérarchiques qui traitent l'information temporelle.

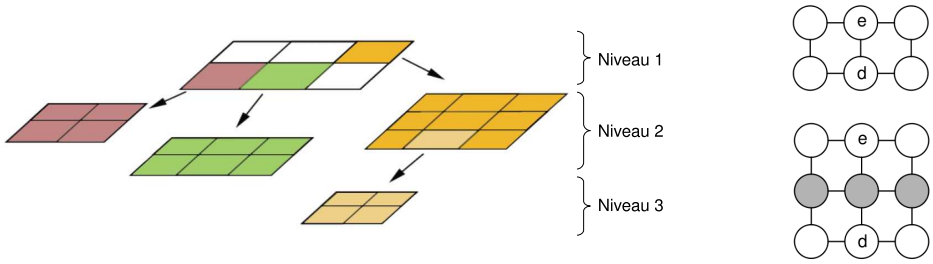
Méthodes hiérarchiques

Les méthodes hiérarchiques sont souvent employées dans des domaines d'application où il convient de décomposer une tâche complexe en sous-tâches plus simples à des niveaux multiples. Ici, chaque niveau comprend une ou plusieurs cartes SOM, généralement construites à différentes échelles de temps. La différence principale entre les méthodes existantes se situe dans le type de codification des résultats des cartes SOM d'un niveau à un autre. Elles diffèrent également par le nombre de niveaux utilisés, le nombre de cartes SOM à chaque niveau, et les interconnexions entre les différents niveaux. Nous exposons, par la suite, deux des nombreuses méthodes hiérarchiques — GHSOM (Growing Hierarchical SOM) et H^2SOM (Hierarchically Growing Hyperbolic SOM) — qui nous semblent importantes pour expliciter le principe sous-jacent à toutes ces méthodes.

GHSOM (Growing Hierarchical Self-Organizing Map)

GHSOM (Rauber et al., 2002) présente une méthode permettant de s'adapter à la distribution des données de façon automatique et itérative. Dans une certaine mesure, GHSOM fournit une solution à deux limites de SOM tenant à sa structure statique et à ses capacités limitées à représenter des relations hiérarchiques entre les données. L'idée de GHSOM est d'établir une structuration hiérarchisée des cartes SOM, où chaque niveau comporte un certain nombre de cartes indépendantes. La phase initiale débute par la création d'une carte SOM singleton (composée d'un seul neurone, son vecteur référent w_0 représente le profil moyen de toutes les données d'entrée) au niveau 0 de GHSOM, et une carte SOM de 2×2 neurones (dont les vecteurs référents sont initialisés aléatoirement) au niveau 1. L'apprentissage de GHSOM repose sur deux stratégies : une stratégie d'élargissement et une stratégie d'expansion hiérarchique, menées respectivement selon deux paramètres τ_1 et τ_2 ($1 > \tau_1 \gg \tau_2 > 0$).

La stratégie d'élargissement consiste à insérer des nouvelles lignes ou colonnes dans des régions où la qualité de représentation des vecteurs d'entrée n'est pas suffisamment bonne. Après un nombre fixé d'itérations, l'erreur MMQE de la carte est calculée par la somme de l'erreur



(a) (b)

FIG. 2.3 – a) Un exemple de la structure hiérarchisée de GHSOM b) Un exemple d'élargissement d'une carte SOM. Une nouvelle ligne est insérée entre deux neurones e et d.

quadratique moyenne de ses neurones ($\text{MQE}_i = \frac{1}{|U_i|} \sum_{k \in U_i} \|x_k - w_i\|$, U_i est le sous ensemble des données associées au neurone i). Une ligne, ou une colonne, est insérée dans la carte si

$$\text{MMQE} > \tau_1 \ \text{MQE}_0 \tag{2.4}$$

où MQE_0 représente l'erreur quadratique d'un neurone parent correspondant à toutes les données d'entrée projetées sur la carte. L'insertion est guidée par le neurone e possédant la plus grande erreur et son voisin p le plus dissimilaire (cf. Figure 2.3).

Une fois que le critère d'élargissement (Eq. 2.4) n'est plus valable, une phase d'expansion peut commencer. Tous les neurones de la carte sont examinés et ceux vérifiant :

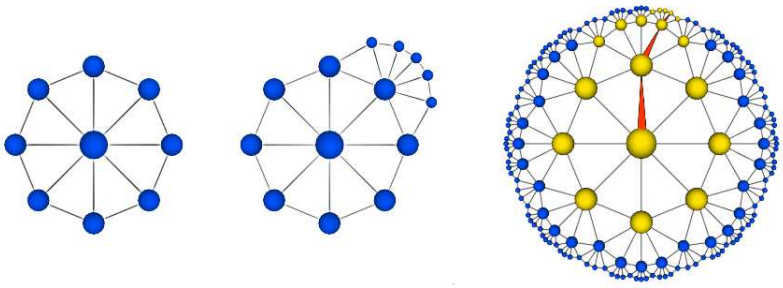
$$\text{MQE}_i > \tau_2 \ \text{MQE}_0 \tag{2.5}$$

sont étendus au prochain niveau hiérarchique en créant pour chacun une nouvelle carte SOM de 2×2 neurones. L'apprentissage de la nouvelle carte se fait en utilisant seulement le sous-ensemble des données qui sont déjà projetées sur le neurone parent étendu. La méthode GHSOM a été appliquée avec succès aux tâches de clustering de documents électroniques (Rauber et al., 2002), la détection des anomalies (Faour et al., 2007) et à la prédiction de séries temporelles (Liu et al., 2006).

H²SOM (Hierarchically Growing Hyperbolic SOM)

Motivé par la virtuosité de l'espace hyperbolique pour gérer de grandes structures hiérarchiques[10], (Ritter, 1999) a introduit la méthode HSOM (Hyperbolic SOM) dans laquelle les neurones sont organisés sur une grille approximativement circulaire consistant en une projection d'un espace hyperbolique sur un espace plat bidimensionnel. En d'autres mots, les neurones sont organisés en anneaux de n triangles équilatéraux de telle façon que chaque neurone de la grille

[10]L'espace hyperbolique $H2$ représente un élément important de la géométrie non-euclidienne. Ses propriétés géométriques spécifiques le rendent un candidat idéal pour gérer de grandes structures hiérarchiques (Lamping et Rao, 1994). En effet, cet espace peut être projeté sur un plan bidimensionnel de l'espace euclidien sous la forme d'un disque unité, également appelé disque de Poincaré (Henle, 2001) possédant un certain nombre de propriétés commodes pour la visualisation. Très particulièrement, il préserve la forme originale de distribution de $H2$ et permet de visualiser des relations complexes entre les données par des mécanismes de focus et de contexte. Ceci permet de se concentrer sur les parties intéressantes d'une distribution originale dans $H2$ tout en gardant toujours une vue générale de son contexte.



(a) (b) (c)

FIG. 2.4 – Topologie de H^{2}SOM a) Les neurones aux sommets — dans ce cas $n = 8$ — des triangles équilatéraux forment le premier niveau dans la hiérarchie de H^{2}SOM. b) Quand un neurone vérifie un critère d'expansion, il est étendu en créant un ensemble de n-3 neurones fils. c) Un chemin virtuel correspondant à l'ensemble des neurones visités selon la stratégie approximative qui pourrait être utilisée pour accélérer la recherche du neurone gagnant est accentué. D'après (Ontrup et Ritter, 2005).

soit entouré par un nombre n de voisins fixé[11]. La figure 2.4(a) montre un exemple pour un nombre minimum de n. L'apprentissage se déroule de manière analogue à celui de SOM. Un des problèmes de cette méthode est que la topologie de la carte est toujours spécifiée a priori. Par ailleurs, le nombre de neurones se développe très rapidement (asymptotiquement exponentiellement) avec des rayons R différents.

La méthode H^{2}SOM a été introduite pour s'affranchir les limites liées à la méthode HSOM (Ontrup et Ritter, 2006). Elle est basée sur le même type de grille que la méthode HSOM. Initialement, un anneau de n triangles hyperboliques équilatéraux, centrés à l'origine de l'espace $H2$ est construit avec n+1 neurones qui forment les triangles de premier niveau hiérarchique. Ces neurones sont initialisés à partir du profil moyen des données, et de petites variations de celui-ci. Chaque neurone dans la périphérie de cette grille peut l'étendre en s'entourant par des neurones issus de n-2 triangles équilatéraux, ce qui revient finalement à former n-gones autour du neurone en question, cf. Figure 2.4. La phase d'apprentissage de ce modèle fonctionne d'une façon similaire à celle de SOM et de HSOM. Pour décider l'extension des neurones, un critère lié à l'erreur quadratique moyenne des neurones est évalué à l'issue d'un certain nombre d'itérations. Une stratégie approximative basée sur l'organisation arborescente des neurones peut également être utilisée pour accélérer la recherche du neurone gagnant de manière significative à chaque étape de l'apprentissage (cf. Figure 2.4(c)).

Pour visualiser les variations de comportement des données avec le temps, l'activité des neurones de la grille est gérée explicitement en y attachant un potentiel dépendant du temps d'activation. Ainsi, l'activité interne d'un neurone i à un instant donné t varie suivant :

$$a_i(t) = \beta\, a_i(t-1) + S_i(t) \tag{2.6}$$

$$\text{avec} \quad S_i(t) = \begin{cases} 1 & \text{si } i \text{ est le neurone gagnant au moment } t \\ 0 & \text{sinon} \end{cases} \tag{2.7}$$

[11] En dehors des neurones du bord de la grille, pour lesquels le nombre de voisins est inférieur, comme dans le cas des cartes SOM.

où β est un facteur définissant le degré d'affaiblissement de l'activité au cours du temps. Cette méthode a été appliquée pour la surveillance des variations thématiques dans un flux de dépêches de l'agence Reuters (Ontrup et Ritter, 2006). L'arrivée de nouvelles dépêches fait augmenter les activités des neurones correspondants dans la hiérarchie de H^2SOM, permettant de détecter des points de convergence de thématiques sur la grille. En cas de divergence, les activités des neurones diminuent.

La méthode H^2SOM représente une des méthodes de visualisation les plus efficaces qui combine les vertus du modèle *SOM* avec un apprentissage plus rapide et des erreurs de quantification et de classification moindres. Cependant, sa capacité d'adaptation aux distributions complexes reste moindre que celle de méthodes à grilles dynamiques ou à topologies libres (cf. Section 2.2.3).

Méthodes non hiérarchiques

Les méthodes non hiérarchiques de SOM peuvent être classées en différentes catégories. Certaines traitent l'information temporelle à l'extérieur de la carte ; elles sont dites cartes à mémoire externe. D'autres traitent l'information temporelle de façon interne au niveau des neurones ou des connexions ; elles sont dites cartes à mémoire interne.

Les cartes à mémoire externe font appel à un codage spatial de la dimension temporelle des données d'entrée avant l'utilisation de la carte, en s'appuyant sur des délais exponentiels pondérés ou des fenêtres temporelles (Kangas, 1991). Ce type de codage présente certains désavantages importants. Il nécessite de retarder ou de retenir les données jusqu'au moment de leur présentation à l'entrée de la carte. En plus de la difficulté à déterminer a priori la taille de la fenêtre temporelle d'entrée ou du nombre de délais. Les cartes à mémoire interne, pour leur part, modifient l'algorithme SOM soit en effectuant une propagation de l'activité temporelle dans la carte, soit en ajoutant un contexte des données passées aux poids des connexions, sous une forme explicite ou implicite. De nombreuses variantes ou extensions des cartes SOM ont ainsi été développées (Guimarães et al., 2003). Nous en présentons ci-après quelques-unes.

Les cartes temporelles de Kohonen (Chappell et Taylor, 1993) sont l'une des premières approches à prendre en compte des dépendances temporelles entre données. Pour cela, elles utilisent des neurones dits "intégrateurs à fuite" (leaky integrators) dont l'activité se dégrade exponentiellement avec le temps. Chaque neurone de la carte est ainsi muni à sa sortie d'un intégrateur à fuite. De cette façon, l'activité temporelle d'un neurone i au temps t est modélisée en fonction de l'entrée courante et l'activité du neurone à l'étape précédente ; elle peut s'écrire :

$$V_i(t) = \alpha V_i(t-1) - \frac{1}{2}\|x(t) - w_i\|^2 \tag{2.8}$$

où $\alpha \in [0, 1[$ est une constante dont dépend l'étendue de la mémoire (Pour α=0 la carte TKM se résume à une carte SOM classique). Le neurone gagnant est celui dont l'activité $V_i(t)$ est maximum, ce qui permet de tenir compte de l'historique des vecteurs d'entrée considérés précédemment lors de la sélection du neurone gagnant. Les poids des neurones sont adaptés pour les rapprocher de l'entrée courante de manière similaire à celle de SOM, et le passé n'est pas pris en compte.

Les cartes récurrentes de Varsta et al. (1997) intègrent l'historique des vecteurs d'entrée considérés précédemment lors de la sélection du neurone gagnant mais également lors de l'adaptation

des positions des neurones. Les intégrateurs à fuite sont localisés au niveau du calcul du vecteur d'erreur au lieu de sa norme. Le vecteur d'erreur d'un neurone i à l'étape t est défini comme suit :

$$y_i(t) = (1-\alpha)y_i(t-1) + \alpha(x(t) - w_i) \tag{2.9}$$

avec $0 < \alpha < 1$. Le neurone gagnant $i^\star$ est celui qui minimise la norme de ce vecteur ($\|y_i(t)\|$). Ce vecteur est aussi utilisé lors de l'adaptation des poids des neurones pour tenir compte du passé. La règle d'adaptation est :

$$\forall\ i, \quad w_i(t+1) = w_i(t) + \alpha(t)\ h_{i^\star}(i,t)\ y_i(t) \tag{2.10}$$

Notons que, pour α=1, Eq. 2.10 se réduit de manière triviale à l'adaptation standard des poids dans une carte SOM. Une comparaison analytique et expérimentale des deux approches TKM et RSOM peut être trouvée dans (Varsta et al., 2001). Une des limites de ces approches est que le calcul de l'activité de chaque neurone ne dépend que de son activité précédente. Des approches alternatives surmontent ce problème en intégrant le contexte temporel de manière plus explicite.

Les cartes récursives de Voegtlin (2002) utilisent une représentation explicite du contexte et incluent les activités précédentes de tous les neurones dans le calcul de l'activité d'un neurone. En plus du vecteur référent standard w_i^x, chaque neurone possède un vecteur de contexte w_i^y, qui doit être comparé aux activités de tous les neurones lors de l'étape précédente, $V(t-1)$. L'activité d'un neurone i est donnée par :

$$V_i(t) = exp(-\alpha\|x(t) - w_i^x\|^2 - \beta\|V(t-1) - w_i^y\|^2) \tag{2.11}$$

où α et β sont des constantes contrôlant l'équilibre entre la distance du neurone à l'entrée et celle du neurone au contexte. Le neurone gagnant $i^\star$ est celui qui maximise $V_i(t)$ et les règles d'adaptation des poids des vecteurs du neurone gagnant et de ses voisins sont :

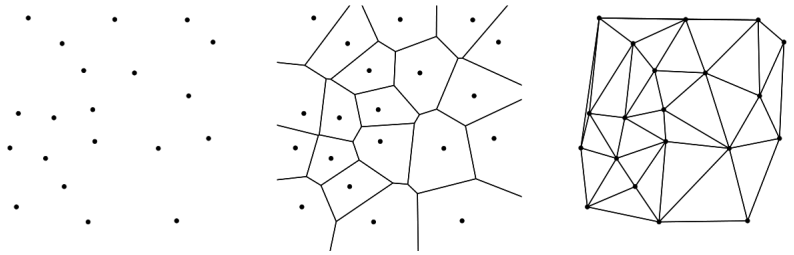
$$\forall\ i, \quad w_i^x(t+1) = w_i^x(t) + \alpha(t)\ h_{i^\star}(i,t)\ (x(t) - w_i^x(t)) \tag{2.12}$$

$$\forall\ i, \quad w_i^y(t+1) = w_i^y(t) + \alpha(t)\ h_{i^\star}(i,t)\ (V(t-1) - w_i^y(t)) \tag{2.13}$$

La plupart de ces approches sont destinées au traitement des données séquentielles. Un neurone représente une séquence, ce qui permet de les utiliser pour différentes tâches comme la segmentation. Cependant, par analogie avec les cartes SOM, ces approches n'opèrent qu'en mode hors ligne, puisque pour assurer la convergence, il faut pouvoir présenter plusieurs centaines de fois l'ensemble des données. Cela constitue une limite importante dans certaines applications orientées flux de données. Dans ce cas, des approches évolutives permettant de construire des modèles neuronaux de manière incrémentale peuvent être plus adaptées.

2.2.3 Les réseaux neuronaux à topologie adaptative

En raison de leur structure topologique fixée, les cartes auto-organisatrices de Kohonen ne s'adaptent pas nécessairement parfaitement à la structure topologique de l'espace des entrées. L'idée de l'auto-adaptativité est donc de ne pas fixer préalablement la topologie du réseau et de le laisser apprendre par lui-même la topologie de la distribution des entrées. Dans ce contexte, les réseaux auto-adaptatifs considèrent deux concepts géométriques importants qui peuvent être associés à l'organisation spatiale des neurones par l'intermédiaire de leurs vecteurs référents : "Tessellation de Voronoï" et "Triangulation de Delaunay".



(a) (b) (c)

FIG. 2.5 – a) Un ensemble de points dans $\mathbb{R}^2$, b) Tesselation de Voronoi, c) Triangulation de Delaunay. D'après (Fritzke, 1997b)

Soit, un ensemble de vecteurs $w_1, w_2, ..., w_N, \in \mathbb{R}^n$ où $w_i = \{w_{i_1}, w_{i_2}, ..., w_{i_n}\})$, la région de Voronoï V_i d'un vecteur w_i est définie comme l'ensemble de tous les points de $\mathbb{R}^n$ pour lesquels w_i est le plus proche vecteur :

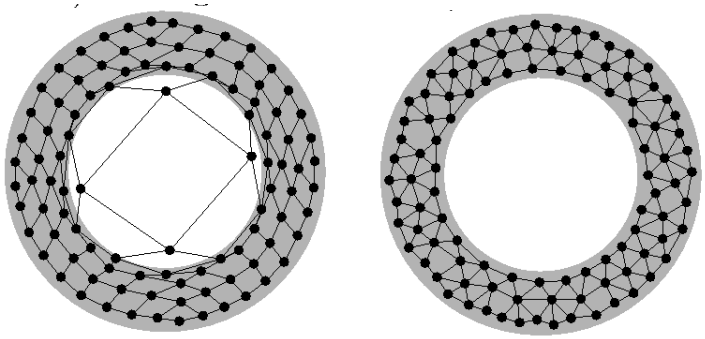
$$V_i = \{x \in \mathbb{R}^n | i = \underset{j \in \{1,...,N\}}{argmin} \|x - w_j\|\} \tag{2.14}$$

La partition de $\mathbb{R}^n$ formée par tous les polygones de Voronoï est nommée "Tesselation de Voronoï" (cf. Figure 2.5 b). La Triangulation de Delaunay est obtenue en reliant par connexions tous les points (neurones) dont les régions correspondantes dans la tesselation de Voronoï sont adjacentes (cf. Figure 2.5 (c)).

La topologie des réseaux auto-adaptatifs est formée en appliquant des principes de l'apprentissage hebbien compétitif, CHL (Competitive Hebbian Learning), proposé par Martinetz (1993). L'idée de base est simplement de créer, à chaque itération, une connexion entre les deux neurones les plus proches de la donnée présentée en entrée (si une telle connexion n'existe pas déjà). Le CHL ne déplace pas les vecteurs référents des neurones mais il les relie pour préserver la topologie de la distribution des données d'entrée. Le graphe résultant constitue un sous-graphe de la triangulation de Delaunay — appelé triangulation induite de Delaunay — restreint aux régions où se trouvent les données d'entrée.

Bien entendu, un apprentissage hebbien compétitif doit être combiné avec une méthode à apprentissage compétitif pour déplacer progressivement les vecteurs référents des neurones en suivant la distribution des données. Une telle combinaison a été initialement introduite par Martinetz et Schulten qui ont combiné un réseau de type "Neural Gas" avec CHL (Martinetz et Schulten, 1994) (cf. Figure 2.6)). Le Neural Gas (NG) (Martinetz et Schulten, 1991) adapte les vecteurs référents des neurones sans aucune contrainte de topologie préalablement établie sur la distribution des données. L'adaptation dépend des distances relatives entre les neurones dans l'espace d'entrée — au lieu des distances relatives entre les neurones dans un treillis topologiquement prédéfini comme c'était le cas des cartes SOM — d'où le nom de ce réseau "Neural Gas". À chaque étape d'apprentissage, les neurones sont ainsi classés dans l'ordre de leurs distances par rapport au vecteur en entrée x. Un rang $k_i(x, w_i) \in \{0, 1, \ldots, N\text{-}1\}$ est affecté à chacun des neurones tel que le rang vaut 0 pour le neurone le plus proche de x, et vaut N-1 pour le neurone le plus éloigné de x. La règle d'adaptation devient alors :

$$\Delta w_i = \alpha(t).h_\lambda(k_i(x, w_i))(x - w_i); \quad h_\lambda(k) = exp(-k/\lambda(t)) \tag{2.15}$$



(a) (b)

FIG. 2.6 – Exemples de (a) une carte SOM et de (b) un réseau NG-CHL entraînés sur des données issues d'une distribution en anneau, après 40000 itérations. D'après (Fritzke, 1997b)

où $\alpha(t)$ et $\lambda(t)$ représentent des fonctions décroissantes du temps (nombre d'itérations).

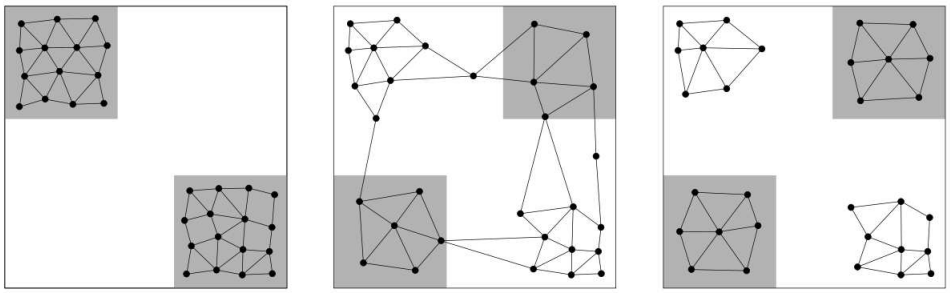
Pendant l'apprentissage, les vecteurs référents étant en adaptation permanente, un mécanisme d'élimination des connexions qui ne sont plus valides est également mis en place dans le réseau NG-CHL. Pour ce faire, un âge est associé à chaque connexion, tel qu'un âge faible d'une connexion signifie une activation intense des neurones sous-jacents, alors qu'au contraire, un grand âge signifie une activation moins fréquente des neurones. Les connexions ne peuvent pas dépasser un certain âge a_{max}, sinon elles disparaissent du graphe. La figure 2.6 illustre la différence entre la topologie d'une carte SOM et celle d'un réseau NG-CHL pour une distribution uniforme de données en anneau.

La liberté topologique inhérente à ce réseau représente un de ses avantages majeurs. Elle lui permet en effet de s'adapter avec souplesse à des distributions multidimensionnelles complexes. Cependant, ce réseau reste stationnaire du fait qu'elle impose de fixer le nombre de neurones à l'avance. La réflexion sur ce problème a ouvert la voie à des réseaux auto-adaptatifs croissants capables de s'adapter à des distributions qui varient dans le temps. Dans ce type de réseaux, des neurones/connexions peuvent y être ajoutés et supprimés lorsque le besoin s'en fait sentir au cours de l'apprentissage. Le Growing Neural Gas (GNG) (Fritzke, 1995) est un exemple typique de tels réseaux, dans lequel la structure topologique du réseau est construite et mise à jours via un processus CHL.

Dans GNG, les neurones sont ajoutés périodiquement à un réseau — initialement composé de deux neurones reliés par une connexion — au cours d'apprentissage. Ces ajouts sont prioritairement effectués dans les régions où le réseau commet le plus d'erreurs, pour y combler le manque de neurones. Dans ce but, une mesure d'erreur locale est accumulée pour chaque neurone pendant l'apprentissage. Initialement, l'erreur est nulle pour tous les neurones ($E_i = 0$). À chaque itération, seul le neurone gagnant $i^\star$ voit l'erreur qui lui est associée s'accroître, suivant :

$$\Delta E_{i^\star} = \|x - w_{i^\star}\|^2 \tag{2.16}$$

Cela aide à détecter les neurones qui recouvrent une large région de la distribution des données d'entrée. Ainsi, un neurone est ajouté périodiquement entre le neurone présentant le cumul d'erreur le plus important et son voisin ayant le cumul d'erreur le plus important. Il est aussi



(a) 20000 itérations (b) 25000 itérations (c) 40000 itérations

FIG. 2.7 – Un réseau GNG essaie (et échoue) de suivre une distribution non stationnaire des données $p(x)$. Le nombre maximum de neurones autorisé est mis à 30. a) Un état initial de $p(x)$ et un réseau GNG qui a atteint sa taille maximum. À ce moment la distribution change. b) Le réseau GNG après 5000 itération depuis le changement de la distribution. c) Après 20000 itérations, quelques neurones ont été adaptés aux nouvelles régions, mais certains sont morts et ils ne sont plus utilisés. D'après (Fritzke, 1997a)

possible, à chaque itération, de supprimer des neurones s'ils se trouvaient sans aucune connexion.

En ce qui concerne l'apprentissage d'un réseau GNG, seuls le neurone gagnant $i^\star$ et ses voisins topologiques directs $N_{i^\star}$ (c.-à-d. l'ensemble des neurones reliés directement par une connexion au neurone $i^\star$) voient leurs vecteurs référents modifiés suivant le même type de règle que celle utilisée pour la carte SOM de Kohonen :

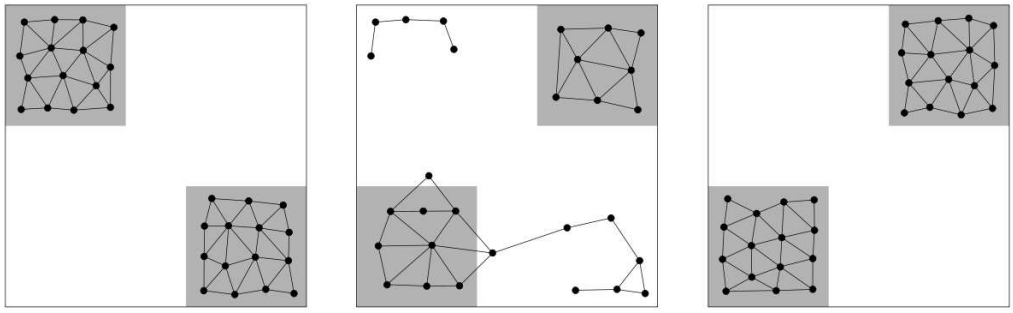
$$\Delta w_{i^\star} = \epsilon_b(x - w_{i^\star}) \tag{2.17}$$

$$\Delta w_i \ = \epsilon_n(x - w_i) \quad (\forall i \in N_{i^\star}) \tag{2.18}$$

où ϵ_b et ϵ_n sont deux taux d'apprentissage tels que $0 < \epsilon_n << \epsilon_b < 1$ (contrairement au cas des cartes SOM, ces taux restent fixes tout au long de l'apprentissage). De plus, à chaque itération, la connexion entre le gagnant et le deuxième neurone le plus proche du vecteur d'entrée est activée et son âge est remis à 0 (Dans le cas où cette connexion n'existe pas, elle est crée et son âge est mis à 0), alors que toutes les connexions adjacentes au gagnant voient leurs âges augmentés de 1. Toutes les connexions dont l'âge est supérieur à a_{max} sont retirées du réseau. Pour une description détaillée des différents algorithmes, nous renvoyons le lecteur aux références sus-citées, et plus particulièrement à (Fritzke, 1997b).

Grâce à sa nature constructive, le GNG n'exige pas de connaître a priori le nombre de neurones à employer. Pourtant, il est possible de spécifier un nombre de neurones maximum au-delà duquel le réseau doit cesser d'augmenter. Autrement, le réseau peut continuer à augmenter jusqu'à atteindre un critère d'arrêt. Les propriétés de GNG font de ce réseau un candidat très intéressant pour des problèmes dont nous ne savons rien ou peu au sujet de la distribution des données. Cependant, GNG est adapté aux distributions stationnaires ou lentement transitoires. Dans le cas de distributions fortement non-stationnaires, beaucoup de neurones pourraient "se coincer" dans d'anciennes régions de haute densité et ne contribuent plus à l'apprentissage du réseau. Ces neurones sont dits "*neurones morts*" (cf. Figure 2.7).

Pour satisfaire le besoin de suivre des distributions non stationnaires, une extension de GNG



(a) 20000 itérations (b) 25000 itérations (c) 40000 itérations

FIG. 2.8 – Un réseau GNG-U qui s'adapte pour suivre une distribution non stationnaire des données $p(x)$. a) L'état du réseau avant le changement de la distribution. b) L'état du réseau après 5000 itérations depuis le changement de la distribution. c) L'état du réseau après 20000 itérations depuis le changement de la distribution. Tous les neurones ont été adaptés et déplacés vers les régions de haute densité. D'après (Fritzke, 1997a)

— Growing Neural Gas with Utility (GNG-U) — a été établie en vue de l'introduction de notion d'utilité des neurones (Fritzke, 1997a). L'idée est de retirer les neurones qui contribuent très peu à la réduction d'erreur, pour les insérer dans des régions où ils pourraient y contribuer de manière plus efficace (cf. Figure 2.8). Pour ce faire, un facteur d'utilité est affecté à chaque neurone et, à chaque itération, seul le neurone gagnant $i^\star$ voit son utilité s'accroître, suivant :

$$\Delta U_{i^\star} = E_i - E_{i^\star} \tag{2.19}$$

avec i le deuxième neurone le plus proche de l'entrée. Cette mise à jour est en relation directe avec l'augmentation de l'erreur, pour une donnée en entrée x, si le neurone gagnant $i^\star$ n'existait pas. Le neurone ayant l'utilité la plus faible U_i est retiré si

$$\frac{E_j}{U_i} > k$$

où j est le neurone ayant la plus forte erreur, et k est une constante.

Une autre variante de GNG — Grow When Required (GWR) — permettant de suivre des distributions non stationnaires a été proposé dans (Marsland, 2002). La différence avec GNG réside dans la manière dont le réseau s'accroît. De nouveaux neurones peuvent, si nécessaire, être ajoutés l'un après l'autre à chaque itération, plutôt qu'un seul neurone après un certain nombre d'itérations. Un nouveau neurone est ajouté entre l'entrée et le gagnant courant, et ne dépend plus du cumul d'erreurs. L'ajout de nouveaux neurones se fait lorsque l'activité du neurone gagnant ($a_{i^\star} = exp(-\|x - w_{i^\star}\|)$ n'est pas suffisamment forte. Pour tenir compte du fait qu'un neurone récemment créé peut ne pas être assez entraîné pour répondre correctement, ce qui signifie que le neurone devrait être encore entraîné plutôt que créer un nouveau nœud, le réseau est muni d'un mécanisme d'habituation pour mesurer combien de fois un neurone a été activé (sélectionné comme étant un gagnant). Selon (Marsland, 2002), GNG-U nécessite un certain temps d'apprentissage pour pouvoir s'adapter au changement de la distribution des données, alors que GWR offre une solution plus simple, et quand la distribution des données change, il déplace les neurones qui ont été précédemment créés et ajoute de nouveaux neurones très rapidement. Cependant, un

bon fonctionnement de ces méthodes reste conditionné au bon choix de ses nombreux paramètres.

Toujours dans le cadre des distributions non stationnaires, il est parfois important de s'adapter au changement sans oublier les informations précédemment apprises (ce qui renvoie au fameux dilemme de stabilité-plasticité (cf. Section 1.4.3)). Une telle capacité d'adaptation offre la possibilité de reprendre l'apprentissage du réseau uniquement avec les nouvelles données et sans avoir à disposition les données précédemment apprises par le réseau. Ceci permet particulièrement de faire un apprentissage en ligne par fragments (ou paquets) dans le cas où les données arrivent en flux. Dans un tel cas, des méthodes comme GNG-U et GWR ne peuvent pas être utilisées du fait qu'elles suppriment les neurones d'anciennes régions de la distribution des données.

Dans l'objectif de conserver les informations précédemment acquises par un réseau type GNG, (Furao et Hasegawa, 2006) ont proposé une extension envisageant deux possibilités d'insertion de neurones. La première est lorsque les distances entre une entrée x et les deux neurones, i, j, les plus proches de celle-ci sont respectivement supérieures aux seuils T_i et T_j. Dans ce cas, un nouveau neurone est ajouté, dont le vecteur référent est celui de x, sans être relié à d'autres neurones. Les seuils sont initialement mis à $+\infty$ et mis à jour de manière adaptative à chaque itération pour le gagnant et le vice-gagnant :

- Si le neurone k, soit le gagnant ou le vice-gagnant, possède des voisins topologiques directs, N_k, le seuil T_k est défini comme étant la distance maximum entre le neurone k et ses voisins :

$$T_k = \max_{r \in N_k} \|w_k - w_r\|$$

- Si le neurone k n'a pas de voisins, le seuil est défini comme étant la distance minimum entre le neurone k et tous les autres neurones N :

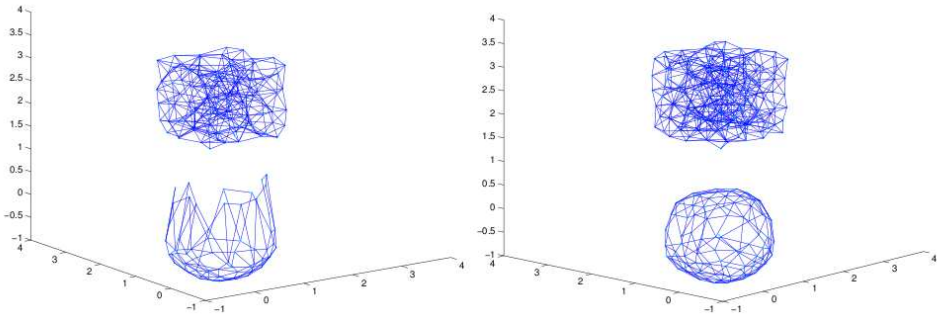
$$T_k = \min_{r \in N \setminus \{k\}} \|w_k - w_r\|$$

La seconde insertion se fait périodiquement, comme c'est le cas dans GNG, entre le neurone présentant le cumul d'erreur le plus important et son voisin ayant le cumul d'erreur le plus important. Toutefois, cette dernière insertion ne sera validée que si elle fait effectivement diminuer l'erreur locale.

De manière quasi similaire, (Prudent et Ennaji, 2005) ont proposé une autre extension de GNG, nommée IGNG (Incremental GNG). Dans IGNG, à chaque itération, un nouveau neurone est ajouté s'il n'existe pas au moins deux neurones qui verifient :

$$\|x - w_i\| \leq \sigma \tag{2.20}$$

où σ est un seuil fixé a priori. Également, dans le cas où seul le neurone gagnant $i^\star$ vérifie Eq. 2.20, un nouveau neurone est ajouté au réseau et relié par une connexion à $i^\star$. Quand un neurone r est inséré, c'est un neurone de type "embryon" avec $w_r = x$ et un âge nul. Dans le dernier cas correspondant à l'existence d'au moins deux neurones vérifiant Eq. 2.20, aucun neurone n'est ajouté au réseau. L'adaptation des vecteurs référents et de la topologie du réseau se font comme pour le GNG. En plus, l'âge de tous les neurones reliés au neurone gagnant est augmenté, et tous les neurones embryons avec un âge supérieur à a_{mature} deviennent des neurones matures. La figure 2.9 illustre le comportement du GNG et du IGNG dans le contexte de distributions non stationnaires.



(a) (b)

FIG. 2.9 – Le comportement du a) GNG et du b) IGNG dans le contexte de distributions non stationnaires, d'après (Prudent et Ennaji, 2005). La figure illustre un problème synthétique composé de deux classes avec une distribution sphérique pour la première et cubique pour la deuxième classe. L'apprentissage est effectué en deux étapes : une première où les réseaux apprennent à partir des données associées à la première classe, puis dans une deuxième étape, ils apprennent à partir des données associées à la seconde classe. Notons que, pour pouvoir apprendre la topologie de la seconde classe, GNG a dégradé la topologie de la première classe.

2.2.4 Les réseaux de type ART

Les réseaux ART sont des modèles neuronaux à apprentissage par compétition, développés par G. A. Carpenter et S. Grossberg, dans le cadre de la théorie de la résonance adaptative (ART, Adaptive Resonance Theory (Grossberg, 1976)). Ils ont été conçus spécifiquement pour contourner le dilemme stabilité/plasticité, la stabilité spécifiant la capacité du réseau à organiser les informations acquises en classes stables, et la plasticité, sa capacité à s'adapter continûment à des données non familières en construisant de nouvelles classes. Pour y arriver, ils considèrent deux types de mémoires : une mémoire à long terme (LTM) qui mémorise les informations familières, contenue dans les connexions et modifiée par apprentissage, et une mémoire à court terme (STM) qui mémorise les nouvelles informations instables, contenue dans les neurones. De nouveaux neurones sont dynamiquement créés quand de nouvelles entrées sont trop différentes des neurones actuels. La stabilité de l'apprentissage est assurée par un mécanisme de résonance qui vérifie l'adéquation entre la sortie du réseau et l'exemple présenté en entrée. La théorie de la résonance adaptative a donné naissance à plusieurs familles de modèles qui se distinguent selon le type des données traitées (ART-1, ART-2, ART-2a, Fuzzy ART, Simplified ART, etc). Nous présentons ici les deux premiers de ces réseaux : ART-1 qui classe des données binaires et ART-2 qui classe des données numériques pondérées. La figure 2.10 illustre l'architecture de base de ce type de réseaux.

Le modèle ART-1 (Carpenter et Grossberg, 1987a) est composé de deux couches de neurones : une couche de comparaison F_1 et une couche de reconnaissance F_2, représentant des mémoires à court terme. Ces couches sont totalement connectées l'une à l'autre par des connexions ascendantes w_{ij} et descendantes z_{ji}. La couche F_1 reçoit les entrées sous forme des vecteurs représentant les exemples à classer, et les transmet à la couche F_2. Cette couche présente un comportement compétitif de type "winner-take-all" dont tous les neurones sont reliés les uns aux autres par des connexions inhibitrices de poids fixes. Lorsqu'une entrée est présentée à la couche

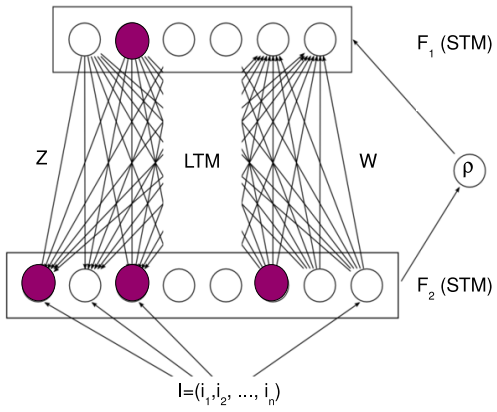


FIG. 2.10 – Architecture de base des réseaux de type ART.

F_1, elle est transmise, via les connexions ascendantes à la couche F_2. En raison de la dynamique compétitive de F_2, un neurone est activé comme candidat pour représenter la classe d'appartenance de l'entrée. Dès qu'un neurone est activé dans la couche F_2, le mécanisme de résonance entraîne la propagation du prototype de la classe choisie sur la couche F_1 via les connexions descendantes. Une comparaison est alors effectuée entre la classe choisie et l'entrée courante. Si elles sont suffisamment proches, l'entrée peut être classée dans la classe associée au neurone courant, sinon, la classe choisie étant incorrecte, le réseau désactive le neurone courant et renouvelle le processus jusqu'à trouver une classe acceptable. Si aucune classe n'est acceptable, alors un neurone supplémentaire est ajouté à la couche F_2 pour classer l'entrée. La notion de similarité est contrôlée via un paramètre de vigilance, ρ, compris strictement entre 0 et 1. Il est nécessaire que ce paramètre soit correctement maîtrisé du fait qu'il contrôle la résolution de la classification : Plus ρ est proche de 1, plus le nombre de classes est important ce qui permet une classification plus fine des entrées.

Pour sa part, le modèle ART-2 (Carpenter et Grossberg, 1987b) a été proposé dans le but de traiter des entrées numériques pondérées. Dans cette version plus complexe, la couche F_1 inclut plusieurs niveaux de pré-traitement qui font subir à l'ensemble des entrées une normalisation et un filtrage éliminant le bruit. De plus, les équations de mise à jour des poids des connexions sont modifiées mais cela ne change pas en profondeur le fonctionnement du modèle par rapport à celui de ART-1.

Malgré tous leurs avantages, les réseaux de type ART présentent encore un certain nombre de défauts. Parmi ceux-ci, il faut mentionner la sensibilité des résultats à l'ordre de présentation des données et, contrairement aux cartes auto-organisatrices de Kohonen, la non préservation des propriétés topologiques.

2.2.5 Autres méthodes

De façon similaire aux méthodes fondées sur k-means, l'algorithme GenIc (Gupta et Grossman, 2004) est également basé sur l'utilisation des paquets (fenêtres sautantes) mais emploie une approche incrémentale de clustering. Il consiste à construire une partition en k clusters à partir de m points de données considérés comme centres initiaux des clusters, $c_1, \ldots, c_m$ dont le poids $w_i = 1$. Chaque nouveau point de données p est attribué au cluster le plus proche et la

mise à jour des centres c_i est incrémentalement effectuée après la présentation de p de la manière suivante : $c_i = (w_i * c_i + p)/(w_i + 1)$; $w_i = w_i + 1$. À la fin de chaque paquet, une probabilité de survie $p_i = w_i / \sum_{i=1}^{m} w_i$ est calculée pour chacun des clusters et puis comparée à un nombre aléatoire δ tiré de façon uniforme sur $[0, 1]$. Chaque cluster dont $p_i < \delta$ doit être détruit et remplacé par un nouveau point de données choisi au hasard dans le paquet suivant. À chaque étape, les m clusters peuvent être regroupés en se basant sur la distance euclidienne afin de former les $k; (k < m)$ clusters finaux. Comparé à la méthode de O'Callaghan et al. (2002) (cf. Section 2.2.1), GenIc est plus rapide lors de l'exécution et semble moins sensible au choix des centres initiaux. Cependant, une dégradation de la performance de GenIc est observée lorsqu'un des clusters a une forte densité en raison du risque que tous les points initiaux soient choisis dans celui-ci.

Il existe certes d'autres méthodes ayant un fonctionnement en ligne, tel que l'algorithme STAR de Aslam et al. (2004) et l'algorithme des k plus proches voisins et ses variantes (Lelu, 2006). Néanmoins, ces algorithmes sont moins adaptés au contexte de flux de données du fait qu'ils requièrent de conserver toutes les données et ont une tendance à générer un nombre très important de clusters, en plus de leur sensibilité à l'ordre de présentation des données.

2.2.6 Synthèse

Après avoir montré les principales approches de clustering possibles, la constatation que l'on peut faire est que l'on dispose actuellement d'un nombre pléthorique de méthodes mais elles ne correspondent pas pour autant toujours aux particularités du traitement des flux de données. Les méthodes de type k-means restent de loin les plus populaires dans le cadre spécifique du clustering de flux de données. Ces méthodes se caractérisent par leur simplicité, leur efficacité et leur relative compatibilité avec les objectifs d'apprentissage en ligne. Cependant, les méthodes proposées jusqu'à présent dont quelques exemples ont été décrits dans la section 2.2.1 semblent être fortement dominées par l'historique du flux et donc insensibles aux évolutions récentes dans ce même flux. De plus, contrairement à l'algorithme CluStream, elles ne permettent d'effectuer des traitements à posteriori que sur l'intégralité du flux observé, la composante temporelle du flux n'étant pas prise en compte. Pour leur part, les méthodes connexionnistes sont souvent considérées comme incompatibles avec les contraintes liées au traitement de flux de données, cela se concrétise tout particulièrement dans le mode d'apprentissage itératif des réseaux de neurones. Des potentialités importantes existent cependant s'ils parviennent à effectuer le moindre calcul. Il s'agit principalement de leurs capacités multiformes en matière d'intégration de la dimension temporelle des données, mais également en matière d'interprétation et de visualization des résultats de clustering. De ce point de vue, nous pensons que les méthodes connexionnistes finiront par émerger des extensions ou des variantes adaptées au traitement des flux de données.

Dans notre travail, nous accordons une importance particulière aux méthodes connexionnistes à topologie adaptative (cf. Section 2.2.3). Ce type de méthodes permet de bien s'adapter au traitement de données multidimensionnelles complexes, à l'image des données documentaires, mais aussi de prendre bien en compte l'évolution des flux non-stationnaires et leur aspect temporel. Par ailleurs, l'exploitation des relations topologiques permet, comme nous allons le montrer dans la section 6.2 du chapitre 6, de maintenir un clustering à des niveaux multiples et avec un temps de calcul presque négligeable. Les méthodes connexionnistes ne sont pas, comme telles, imputables au clustering de flux de données, nous mettons pour l'instant au point notre approche sur la base de l'utilisation des fenêtres temporelles.

2.3 Classification de flux de données

Le problème de la classification supervisée est abondamment abordé dans des disciplines très diverses dans le but de trouver une façon de classifier une donnée dans une des classes prédéfinies avec une erreur minimum de classement. La classification s'effectue la plupart du temps à partir d'une base d'apprentissage qui est un ensemble d'exemples étiquetés (chaque exemple est attaché à une classe via une fonction d'étiquetage). Elle comprend en général deux phases : la phase d'apprentissage consiste à construire un classifieur à partir d'exemples étiquetés par leur classe, et ensuite, la phase de classement consiste à prédire la classe de nouveaux exemples avec le classifieur construit.

Il existe une grande variété de méthodes de classification dans la littérature (Sebastiani, 2002). Toutefois, la plupart de ces méthodes ont été conçues en faisant l'hypothèse que les données sont indépendantes et identiquement distribuées (bien qu'elle ne soit généralement pas réaliste), et les modèles de classification sont souvent établis hors ligne à partir de données statiques sur lesquelles plusieurs passages sont possibles. Or, dans le contexte des flux de données, les méthodes de classification doivent s'attaquer à de nombreux défis afin de mener à bien leur tâche. Ces défis se rapportent principalement à :

- La classification est typiquement effectuée en ligne ;
- La complexité des calculs en termes de temps et d'espace lors de la phase d'apprentissage d'un modèle de classification et de la phase de classement ;
- La nécessité de disposer d'un volume important de données d'apprentissage.
- Le problème dit de "dérive de concept" dû au caractère dynamique de flux de données.

Dans cette section, nous aborderons les questions primaires relatives au problème de la classification de flux de données. Nous commençons par exposer le problème de "dérive de concept", et puis nous présentons quelques méthodes majeures expressément conçues pour la classification des flux de données non-stationnaires, en en distinguant trois familles, à savoir, les méthodes à apprentissage en ligne, les méthodes à base de fenêtres temporelles et les méthodes à base d'ensemble de classifieurs.

2.3.1 La "dérive de concept"

La dynamicité de flux de données fait apparaître la notion de la "dérive de concept". Un concept désigne, selon la définition que l'on donne en apprentissage, une classe de données ayant des caractères communs. La dérive de concept signifie donc que, pour diverses raisons, les données, leurs étiquettes, et leurs relations, peuvent évoluer tout au long du temps. Les modèles de classification étant à la base de la distribution des données et leurs étiquettes, il est indispensable de les adapter régulièrement aux changements survenus afin de prévoir correctement des classes effectivement présentes dans le flux au temps de la prévision. Très souvent, la cause du changement n'est pas connue a priori, ce qui rend le problème de classification encore plus difficile. Dans la pratique, la dérive de concept peut être brusque ou progressive (cf. Fig. 2.11). Les dérives progressives peuvent aussi être divisées en dérives modérées et lentes (Stanley, 2003). Reprenons cela plus formellement : Soient A et B deux concepts et soit $\{i_1, i_2, \ldots, i_n\}$ une séquence de données à présenter à un système de classification. Supposons que le concept A soit stable avant l'arrivée d'une donnée i_t, et entre les données i_t et $i_{t+\Delta t}$ le concept change en passant de A à B. Si $\Delta t = 1$ la dérive de concept est dite brusque ou instantanée. Quand $\Delta t > 1$ le concept change

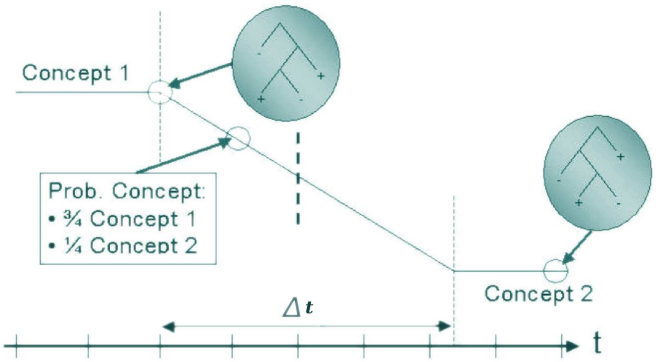


FIG. 2.11 – Exemple d'une dérive de concept progressive, passant du concept 1 au concept 2. Notons qu'avant la ligne pointillée c'est le concept 1 qui domine sur le concept 2, et vice-versa après la ligne. D'après (Scholz et Klinkenberg, 2007).

à travers un certain nombre de données et la dérive est dite progressive. Ce type de changement peut être modélisé par une fonction α représentant la dominance du concept A sur le concept B. Ainsi, avant i_t, $\alpha = 1$, et après $i_{t+\Delta t}$, $\alpha = 0$. Plus la période de la dérive, Δt, est courte, plus la vitesse de changement est rapide.

La prise en compte de l'aspect dynamique de flux de données a conduit au développement de nouvelles approches de classification, dont la plupart s'appuient sur le fait qu'*il faut oublier pour s'adapter*. Ainsi, une solution courante dans beaucoup d'approches consiste à oublier en permanence les anciennes données à un taux constant T, c.-à-d., après qu'une période de temps T soit passée depuis leur arrivée. Selon la stratégie suivie pour exclure les données anciennes, quatre méthodes peuvent être distinguées :

- Méthodes à apprentissage en ligne permettant une mise à jour continue des modèles ;
- Sélection de données récentes à base de fenêtres temporelles ;
- Pondération de données en fonction du temps ;
- Méthodes à base d'ensemble de classifieurs avec des descriptions multiples de concept.

Les trois dernières méthodes emploient, dans bien des cas, des mises à jour périodiques des modèles d'apprentissage. De même, il existe des méthodes de mise à jour différée où les modèles d'apprentissage construits sont à modifier uniquement en cas de dérive de concept assez importante (Valizadegan et Tan, 2007). Ces méthodes requièrent un mécanisme de détection de changements dans la distribution des données pour déterminer si le modèle est obsolète ou non. Nous présenterons des exemples de tels mécanismes dans la section 2.4 de ce chapitre.

2.3.2 Méthodes à apprentissage adaptatif en ligne

Dans ce type de méthodes, le modèle d'apprentissage est mis à jour à chaque fois qu'une nouvelle donnée est arrivée. De telles méthodes constituent une meilleure solution au problème de dérive de concept qu'une mise à jour périodique des modèles à partir des portions séparées du flux. Ceci est d'autant plus vrai que les méthodes arrivent à gérer au mieux l'oubli des données obsolètes dans l'historique du flux. Nous nous intéressons ici aux méthodes SVM adaptées au mode d'apprentissage en ligne.

Les méthodes des machines à vecteurs supports (SVM) ont été employées avec succès pour apprendre de grands ensembles de données multidimensionnelles. Néanmoins, le calcul d'un modèle de classification de type SVM peut être très coûteux en termes de temps et d'espace mémoire lors de traitement de grandes quantités de données. Pour dépasser ces limites, il est donc préférable de rendre son mode de fonctionnement en ligne où les exemples sont présentés les uns après les autres. Ainsi, certaines méthodes ont été mises au point en tenant compte du fait que les SVM peuvent apprendre le même modèle de classification à partir des vecteurs supports qu'à partir de tous les exemples d'apprentissage, et que le nombre de vecteurs supports ne représente qu'une partie relativement constante du nombre total des exemples d'apprentissage (Steinwart, 2003). Ces méthodes s'appuient ainsi sur une sélection active des exemples à considérer lors de l'apprentissage du modèle et donnent des résultats assez satisfaisants pour une utilisation en ligne.

L'approche de Syed et al. (1999) suppose que les données sont découpées en séquences partielles (paquets), en respectant l'espace mémoire disponible. À chaque étape, pour chaque paquet, un modèle SVM est construit à partir des données du paquet en question et des vecteurs supports issus de l'étape précédente d'apprentissage. Le problème avec cette approche est qu'elle repose sur l'hypothèse que chaque paquet est un échantillon représentatif de l'ensemble des données, tandis qu'il y a aucun moyen de s'assurer une telle hypothèse dans le cas où les exemples arrivent en ligne. Pour cette raison, et du fait que le nombre de vecteurs supports est typiquement très faible au regard du nombre des exemples considérés dans les paquets, l'influence des vecteurs supports sur la fonction de décision lors de la prochaine étape d'apprentissage peut être très faible si les nouvelles données sont différemment distribuées (un cas similaire à une présence de dérive de concept, cf. Section 2.3.1). Généralement, une propriété intrinsèque des SVM les rend robustes aux données aberrantes. Seulement dans le cas précis de cette approche, les données aberrantes sont les vecteurs supports que l'on souhaiterait prendre en considération dans la définition de la nouvelle fonction de décision.

Selon (Rüping, 2001), une solution au problème posé ci-dessus est de rendre une erreur sur l'un des anciens vecteurs supports (S) plus coûteuse qu'une erreur sur l'une des nouvelles données (I). Plus formellement, le problème d'optimisation énoncé dans la section 1.4.5 du chapitre 1, devient :

$$min \quad \frac{1}{2} w.w + C(\sum_{i \in I} \xi_i + L \sum_{i \in S} \xi_i) \tag{2.21}$$

avec L est une constante proportionnelle au nombre d'exemples dans le paquet précédent divisé par le nombre de vecteurs supports. Cette méthode s'est révélée utile dans le cas d'une présence de dérive de concept entre les paquets successifs.

L'algorithme LASVM de Bordes et al. (2005) offre une optimisation incrémentale de la fonction objectif duale $\mathcal{W}(\alpha)$ (cf. Eq 1.7). Cet algorithme est fondé sur le principe d'optimisation des SVM de type SMO (Sequential Minimal Optimization) avec des méthodes d'apprentissage actif pour la sélection des exemples les plus informatifs en fonction de différents critères (par exemple : proximité de la frontière de décision). LASVM n'optimise pas la fonction objectif de manière aussi précise que SMO mais il est plus rapide que ce dernier et nécessite beaucoup moins de mémoire. D'autres algorithmes intéressants conçus pour le fonctionnement en ligne des SVM (que nous ne présentons pas ici), peuvent être consultés dans (Cauwenberghs et Poggio, 2001), et (Fung et Mangasarian, 2002).

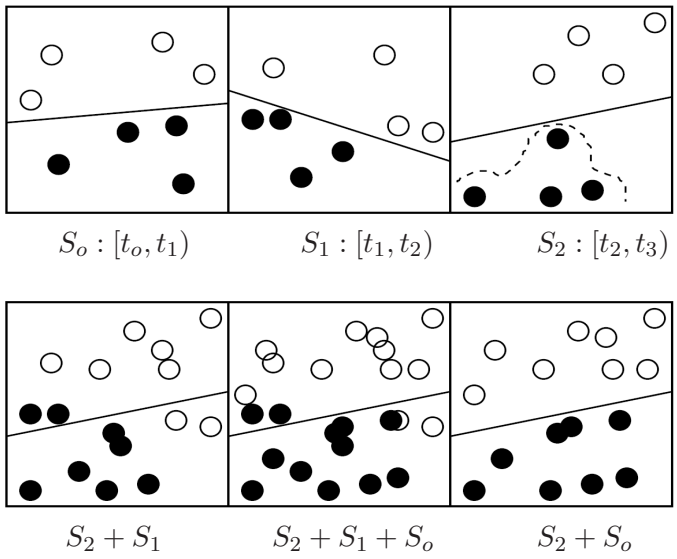


FIG. 2.12 – Illustration des problèmes liés à la dérive de concept, d'après (Wang et al., 2003). (a) la distribution des données issues des trois fenêtres (ou paquets) séquentielles sur un flux de données ; S_i représente donc les données arrivant entre t_i et t_{i+1}. Dans cet exemple, notons que si l'on fait l'apprentissage d'un modèle de classification uniquement à partir des données dans S_2, on risque de provoquer un sur-apprentissage du modèle du fait qu'il est construit à partir de très peu de données d'apprentissage. (b) Quel ensemble des données doit on utiliser pour faire l'apprentissage ? Notons que si l'on augmente le nombre de données d'apprentissage, on peut également réduire l'exactitude de classification du fait que le modèle devient moins sensible aux changements imprévisibles des données.

2.3.3 Méthodes par sélection des données

L'idée générale qui sous-tend ce genre d'approches est de limiter l'ensemble d'apprentissage des modèles de classification à une fenêtre d'un certain nombre de données récemment vues. Les données sont ajoutées à la fenêtre dès qu'elles arrivent et les données les plus anciennes sont en supprimées. Dans le cas le plus simple, la fenêtre est d'une taille fixe, et les données les plus anciennes seront exclues toutes les fois qu'une nouvelle donnée est arrivée (Widmer et Kubat, 1996). Bien que cette solution soit conceptuellement simple, elle tend à poser le dilemme suivant : une petite taille de la fenêtre limitait l'exactitude du modèle de classification — dans les phases sans dérive de concept — du fait qu'il est construit à partir de très peu de données d'apprentissage ; alors qu'une taille plus élevée rendrait le modèle moins sensible aux changements conceptuels imprévisibles des données (cf. Fig. 2.12). Autrement dit, le choix d'une "bonne" taille de la fenêtre est un équilibre entre adaptativité (petite fenêtre) et généralisation (large fenêtre). De ce fait, quelques approches ont procédé un peu différemment en utilisant des heuristiques pour ajuster dynamiquement la taille de la fenêtre en fonction de l'étendue de la dérive de concept (Widmer et Kubat, 1996). Une bonne stratégie devrait rétrécir la fenêtre (en oubliant les données anciennes) quand une dérive de concept semble se produire, et maintenir la taille de fenêtre fixe quand le concept semble être stable. Autrement, la fenêtre devrait progressivement s'élargir jusqu'à qu'une description stable de concept puisse être formée.

Un ensemble d'algorithmes de classification à base de règles simples de décision, connus sous

le nom de FLORA, ont été développés dans (Widmer et Kubat, 1996). Ils reçoivent en entrée un flux de données, où celles-ci représentent des exemples positifs et négatifs d'un concept cible qui change avec le temps. L'algorithme FLORA d'origine utilise une fenêtre glissante de taille fixe centrée sur les données récemment vues. Dans la définition d'un concept, FLORA fait intervenir des groupes de variables correspondant aux variables positives (ADES), variables négatives (NDES), et variables potentielles (PDES) qui sont apparues dans la description des exemples positifs, mais également quelques négatifs et qui sont alors susceptibles de devenir positifs avec le temps. Ces variables sont conjonctives et pondérées par le nombre d'exemples — inclus dans la fenêtre courante — dans lesquels elles sont apparues. La mise à jour des variables se fait incrémentalement lors du déplacement de la fenêtre suite à chaque ajout/oubli d'un exemple positif ou négatif.

L'algorithme FLORA a fait l'objet de nombreuses modifications afin de pouvoir aborder certains des problèmes liés. FLORA2 utilise deux indicateurs heuristiques, pour ajuster dynamiquement la taille de la fenêtre, la performance de prédiction et le nombre de variables positives. L'hypothèse de base est que la dégradation soudaine de la performance ou l'explosion du nombre de variables pourraient signaler le début d'une dérive de concept. Ainsi, si une dérive est prévue, la taille de la fenêtre est réduite de 20%, sinon la taille est réglée selon la performance de prévision : si, d'une part, la performance est de très haut niveau, la taille de la fenêtre est réduite de 1 (après l'ajout d'un nouvel exemple, les deux exemples les plus anciens sont éliminés) afin de ne pas conserver inutilement un grand nombre d'exemples. D'autre part, si la performance semble être suffisamment satisfaisante la taille de fenêtre reste inchangée. Si aucune de ces conditions n'est satisfaite, la taille de fenêtre est incrémetalement augmentée de 1 pour y inclure plus d'exemples.

Une autre extension de cet algorithme a encore été proposée de façon à lui permettre de mettre en valeur les contextes périodiques via un mécanisme de réactivation des concepts périmés (FLORA3), l'objectif étant d'éviter la répétition de l'apprentissage du même concept. Ainsi, les concepts stables périmés sont toujours maintenus et utilisés a posteriori au cas où ils se manifesteraient. Le dernier descendant de la famille de FLORA (FLORA4) est destiné à améliorer la performance en termes de robustesse au bruit. FLORA4 a traité ce problème en maintenant des solutions généralisées auxquelles des niveaux de confiance sont assignés. Plus précisément, FLORA4 n'exige pas l'uniformité des variables (par exemple, l'absence totale des variables positives dans les exemples négatifs) mais impose plutôt une notion de fiabilité estimée statistiquement sur les variables.

Dans le même cadre des travaux liés aux approches par sélection des données, Domingos et Hulten (2000) ont introduit les arbres de décision très rapides VFDT ("Very Fast Decision Tree") dédiés à la classification en ligne des flux de données stationnaires. L'idée réside dans l'utilisation de la borne de Hoeffding[12] afin de déterminer, avec une forte probabilité, le nombre minimum d'exemples, n, requis à un nœud pour choisir la meilleure variable de division. Cette variable devrait donc être identique à celle que l'on pourrait choisir en utilisant la totalité "infinie"

[12]La borne de Hoeffding permet de borner la valeur réelle d'une variable aléatoire par rapport à son espérance et un terme d'erreur : Soient $X_1, \ldots, X_n$, n observations d'une variable aléatoire X à valeurs dans [0, 1], pour tout $\delta > 0$, nous avons :

$$Prob(\mathbb{E}(X) \leq \frac{1}{n}\sum_{i=1}^{n} X_i + \epsilon) > 1 - \delta; \quad \text{avec} \quad \epsilon = \sqrt{\frac{ln(1/\delta)}{2n}}$$

où $\mathbb{E}(X)$ l'espérance mathématique de X.

des données du flux. En bref, le principe est comme suit : Soient n observations d'une variable aléatoire C qui prend ses valeurs sur la gamme R, où C est un critère de sélection de la variable de division provenant de la théorie de l'information comme, par exemple, le gain d'information et l'index de Gini (R est 1 pour une probabilité, et $log\ c$ pour le gain d'information, où c est le nombre de classes). Supposons $\overline{C}$ est la moyenne empirique de cet échantillon, la borne de Hoeffding affirme que la vraie moyenne de C est au moins $\overline{C} - \epsilon$, avec une probabilité $1 - \delta$, où δ est un paramètre à fixer par l'utilisateur et

$$\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$$

Cette borne a l'avantage d'être indépendante de la distribution de probabilité du critère de sélection $\overline{C}$. VFDT prend comme entrée un flux de données (training exemples décrits par un ensemble de variables X) et le paramètre de confiance δ. Pour un nœud donné, soit X_a la variable ayant la valeur la plus grande d'un critère de sélection $\overline{C}$, et soit X_b la variable ayant la deuxième plus grande valeur, si $\Delta\overline{C} = \overline{C}(X_a) - \overline{C}(X_b) > \epsilon$, alors X_a est la meilleure variable de division avec une probabilité $1 - \delta$, autrement le nœud doit accumuler plus d'exemples jusqu'à ce que la condition soit vraie. VFDT comprend également quelques raffinements afin d'optimiser le processus :

- Lorsque deux ou plusieurs variables ont des valeurs très proches, il faudra beaucoup d'exemples afin d'en sélectionner la variable de division avec une confiance de haut niveau, ce qui s'avère inutile. Ainsi, VFDT peut optionnellement diviser un nœud en utilisant la meilleure variable courante si $\Delta\overline{C} < \epsilon < \tau$, où τ est un seuil prédéfini.
- Le calcul du critère de sélection $\overline{C}$ n'est fait qu'après l'accumulation d'un nombre minimum d'exemples dans les nœuds feuilles.
- Lorsqu'une certaine limite mémoire est atteinte, VFDT désactive les feuilles les moins prometteuses en termes d'erreur de classification. En outre, les variables ayant des valeurs faibles de critère de sélection sont libérées de la mémoire.

Plus tard, VFDT a été étendu au cas des flux de données non-stationnaires (Hulten et al., 2001). Cette extension, qui porte le nom de CVFDT, est aussi basée sur l'utilisation des fenêtres glissantes. À l'arrivée de chaque nouvel exemple, CVFDT met à jour les statistiques de son modèle. Ceci n'aura statistiquement aucun effet si le concept à apprendre est stationnaire. Si le concept change, cependant, quelques divisions qui ont précédemment satisfait la borne de Hoeffding ne seront plus valides du fait qu'une variable alternative aurait maintenant une valeur plus élevée du critère de sélection. Dans ce cas-ci, CVFDT commence la construction d'un sous-arbre en attachant la nouvelle meilleure variable à sa racine. Quand ce sous-arbre alternatif devient plus précis que l'arbre courant, il le remplace. CVFDT a aussi la capacité de changer dynamiquement la taille de la fenêtre pour s'adapter à l'étendue de la dérive de concept. Ceci est fait en diminuant la taille de la fenêtre lorsque plusieurs nœuds de l'arbre deviennent simultanément invalides (ce qui pourrait indiquer un changement soudain de concept) et en augmentant la taille de la fenêtre lorsque les nœuds de l'arbre semblent être plutôt stables. Des extensions de VFDT au cas des données numériques ont aussi été développées (Jin et Agrawal, 2003; Gama et al., 2003).

Il existe d'autres méthodes qui ont suivi une démarche un peu près similaire à celles présentées ci-avant. Citons par exemple les systèmes de classification en ligne de type OLIN (On Line Information Network) de Last (2002), et l'approche de Klinkenberg et Thorsten (2000) qui s'appuie sur la méthode des machines à vecteurs supports (SVM).

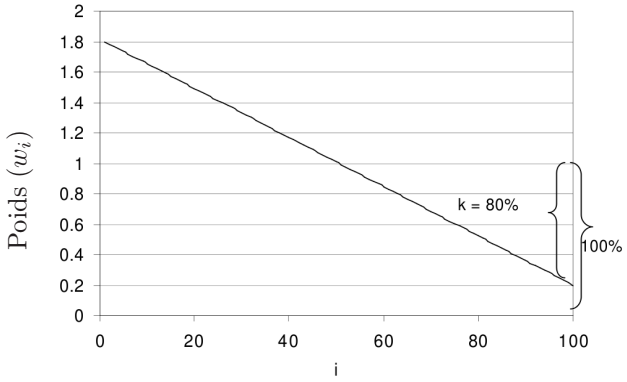


FIG. 2.13 – Un exemple d'une fonction d'oubli linéaire ($N = 100$, $k = 80\%$), d'après (Koychev, 2000)

2.3.4 Méthodes par pondération des données

Les approches par pondération présentent des stratégies pour oublier progressivement les données en se basant sur des fonctions d'oubli. Ces fonctions attribuent un poids décroissant aux données en fonction du temps, ce qui rend les données récentes plus importantes pour les algorithmes d'apprentissage que les anciennes données. Les algorithmes d'apprentissage utilisés doivent souvent être modifiés pour pouvoir traiter des données pondérées. À voir, par exemple, le travail de Kim et al. (2006) qui présente une modification de l'approche de Bayes Naïfs (NB), et le travail de Klinkenberg (2004) qui présente une version de SVM avec pondération.

Dans ce contexte, plusieurs types de fonctions d'oubli peuvent être envisagés. (Koychev, 2000) atteste qu'une famille de fonctions sous un ensemble de contraintes :

$$w_i \geq 0 \quad \text{et} \quad \frac{\sum_{i=1}^{n} w_i}{n} = 1$$

est appropriée pour la plupart des algorithmes d'apprentissage. La fonction d'oubli proposée dans (Koychev, 2000) est une fonction linéaire à appliquer aux données dans une fenêtre temporelle :

$$w_i = -\frac{2k}{n-1}(i-1) + 1 + k$$

où $i = 1..n$ est un paramètre reflétant l'ordre d'arrivée des données tel que la valeur 1 est attribuée à la donnée la plus récente ; n représente la longueur de la fenêtre (ou la séquence des données observée) ; $k \in [0, 1]$ est un paramètre correspondant au pourcentage de réduction du poids de la première donnée vue, mais aussi au pourcentage d'augmentation du poids de la dernière donnée vue (cf. Figure 2.13). En variant k la pente de la fonction peut être ajustée. Une autre famille de fonctions intéressante est les fonctions exponentielles. Dans (Klinkenberg, 2004), les données sont pondérées en appliquant une fonction exponentielle de type $w_i = e^{-\lambda t_i}$. Dès lors, cette fonction fait que plus la valeur de λ devient grande, plus l'expiration des données devient rapide. Au cours des expériences menées dans les études citées ci-avant ((Koychev, 2000) et (Klinkenberg, 2004)), les approches par pondération ont été jugées, d'une part, supérieures aux approches par sélection simples (les approches à une fenêtre de taille fixe) en matière de

prédiction et de rapidité d'adaptation en cas de dérive du concept. D'autre part, les approches par pondération se sont avérées être moins performantes que les approches par sélection à base des fenêtres adaptatives.

2.3.5 Méthodes à base d'ensemble de classifieurs

Ce type de méthodes repose sur le principe qu'un ensemble de classifieurs peut aboutir à une solution bien meilleure que celle d'un seul classifieur. L'apprentissage consiste donc à déterminer les solutions partielles des classifieurs et à trouver une manière efficace pour les combiner afin d'obtenir la solution générale.

La méthode de Wang et al. (2003) est basée sur l'observation que la construction d'un classifieur à partir des données les plus récentes n'est pas nécessairement le choix idéal, puisque des informations potentiellement utiles peuvent être gaspillées en jetant toutes les données les plus anciennes. Par exemple, la figure 2.12 montre que la combinaison des données de S_2 et S_o aide à réduire le risque de sur-apprentissage et de contradiction entre concepts. La raison est que S_2 et S_o ont des distributions semblables de classes (concepts). L'idée est donc que l'expiration des données doit se fonder sur leur distribution au lieu des critères basés seulement sur leur temps d'arrivée. Il a été mis en évidence dans (Wang et al., 2003) qu'un ensemble de classifieurs générés à partir des paquets $S_1, S_2, \ldots, S_n$ assure une classification meilleure qu'un seul classifieur généré à partir de $S_1 \cup S_2 \cup \ldots \cup S_n$. Pour y parvenir, il faut cependant que le poids de chaque classifieur C_i soit inversement proportionnel à son erreur. Ainsi, toutes les fois qu'un nouveau paquet de données S_n arrive, un nouveau classifieur est construit et les poids des classificateurs précédents sont ajustés en utilisant les données de ce paquet. Plus spécifiquement, supposons que S_n consiste en des couples (x, c) où c indique la vraie classe de x. L'erreur de prédiction d'un classifieur C_i est $1 - f_c^i(x)$, où $f_c^i(x)$ est la probabilité donnée par C_i que x appartienne à c. Donc, l'erreur quadratique moyenne de C_i peut s'écrire :

$$\mathrm{MSE}_i = \frac{1}{|S_n|} \sum_{(x,c) \in S_n} (1 - f_c^i(x))^2$$

Le poids de C_i doit alors être inversement proportionnel à MSE_i. D'autre part, l'erreur résultant d'une classification aléatoire (la probabilité que x soit dans c est égale à la probabilité de c $P(c)$) est donnée par :

$$\mathrm{MSE}_r = \sum_c p(c)(1 - p(c))^2$$

Cette erreur est retenue comme seuil afin d'exclure chaque classifieur qui n'est pas capable de classer les données mieux qu'une classification aléatoire, et le poids associé à chaque classifieur est ainsi :

$$w_i = \mathrm{MSE}_r - \mathrm{MSE}_i$$

La décision finale prise par l'ensemble de classifieurs est donc une moyenne pondérée de $f_c^i(x)$.

Comme nous venons juste de le signaler, Wang et al. (2003) ont conclu sur la base de leurs analyses théoriques et empiriques qu'un ensemble de classifieurs pondérés est plus précis qu'un seul classifieur dans le cas de la présence d'une dérive de concept. Cependant, aucune conclusion n'a été tirée quant à la précision des classifieurs entraînés sur "différents nombres de paquets" ou "différentes quantités d'anciennes données". Cet aspect a été abordé plus spécifiquement dans le travail de Fan (2004) dont le focus essentiel est sur le fait de savoir s'il est plus précis de

construire un classifieur à partir des données les plus récentes uniquement, ou s'il faut de plus considérer un certain nombre de données plus anciennes. En partant de l'idée que le modèle de classification optimal peut varier considérablement selon les différentes situations qui pourraient être évoquées :

1. Dans le cas où le nombre de données récentes est suffisant et où il n'y a pas de dérive de concept, le modèle optimal peut être celui entraîné sur ces données elles-mêmes, ou un modèle ancien peut être aussi optimal s'il a été entraîné sur un nombre suffisant de données.
2. Dans le cas où le nombre de données est insuffisant et où il n'y a pas de dérive de concept, le modèle optimal est celui entraîné sur les données récentes uniquement.
3. Si le nombre de données récentes n'est pas suffisant et si il n'y a pas de dérive de concept, le modèle optimal est un modèle ancien entraîné sur un nombre suffisant de données s'il en existe ou bien un nouveau modèle entraîné sur les données récentes plus un nombre suffisant de données plus anciennes existantes.
4. Dans le cas où le nombre de données récentes est insuffisant et où il y a une dérive de concept, un modèle doit être entraîné sur les données récentes plus un nombre suffisant de données plus anciennes ayant le même concept que les données récentes.

Bien évidemment, ces situations ne peuvent pas être prévues ou détectées aisément d'où l'intérêt de la proposition faite par Fan (2004) de comparer quelques modèles statistiquement possibles et de choisir celui ayant la plus faible erreur en se basant sur une méthode de validation croisée. L'avantage de cette méthode est qu'elle permet au modèle de classification de s'adapter plus rapidement dans le cas d'une dérive brusque de concept.

Des gammes de variations intéressantes de méthodes à base d'ensemble de classifieurs peuvent être trouvées dans (Scholz et Klinkenberg, 2007; Tsymbal et al., 2008).

2.3.6 Synthèse

En dehors des particularités partagées à travers différentes aspects de traitement de flux de données, la grand défi en matière de classification de flux de données réside dans la capacité de s'adapter à la dérive de concept qui peut potentiellement intervenir dans les flux de données non stationnaires. Nous avons abordé dans cette section quatre principales familles d'approches destinées à traiter le problème de la dérive de concept : 1) approches à apprentissage adaptatif en ligne, 2) approches par sélection des données, 3) approches par pondération des données et 4) approches à base d'ensemble de classifieurs.

L'approche la plus populaire semble être la sélection de données à l'aide de fenêtres temporelles. Cependant, une telle approche présente de nombreux problèmes quant à l'ajustement de la taille de la fenêtre. En effet, une dérive de concept peut se produire à n'importe quel point du flux, elle peut être brusque ou progressive, et sa durée peut être courte ou longue. Si les méthodes fondées sur la base de fenêtres temporelles peuvent être envisagées dans le cas d'une dérive progressive de concept, elles deviennent incertaines dans le cas d'une dérive brusque de concept. L'approche par pondération des données ne semblent non plus fonctionner dans ce même cas. Pour leur part, les approches à base d'ensemble de classifieurs parviennent généralement à traiter les deux types de dérive, brusque et progressive, du fait qu'il en résulte immédiatement une forte réduction des poids associés aux modèles de classification appris précédemment. En contrepartie, il est moins clair comment appliquer des méthodes d'ensemble directement dans un

mode de fonctionnement en ligne. Souvent des hypothèses fortes sont faites sur le nombre de données pour chaque classifieur ou sur le nombre de classifieurs à retenir. Enfin, les approches à un apprentissage adaptatif en ligne représentent la solution la plus évidente et radicale au problème de la dérive de concept. Cependant, très peu de méthodes permettent aujourd'hui d'envisager de tel type d'apprentissage dont la plupart sont développées sur la base de SVM. Le modèle ILoNDF que nous développons dans ce travail se situe également dans le cadre des méthodes à un apprentissage adaptatif en ligne. Il présente l'avantage de pouvoir gérer efficacement la présence d'une dérive de concept aussi bien brusque que progressive. Le comportement du modèle ILoNDF en présence d'une dérive de concept sera explicité dans la section 5.3 du chapitre 5.

2.4 Analyse de changements dans les flux de données

La nature dynamique inhérente aux flux de données ébranle le paradigme classique d'analyse de données. En sus des différences algorithmiques entre le traitement des données statiques et le traitement des flux de données, il existe également certains problèmes étroitement liés à la dimension temporelle intrinsèque aux flux de données. L'intégration de la dimension temporelle dans l'analyse de données implique un certain nombre d'aspects qui ont été étudiés, parfois individuellement, sous différents angles. Principalement, il s'agit de pouvoir traiter de manière simultanée l'analyse des régularités inhérentes au flux de données et celle des nouveautés, exceptions, ou changements survenant dans un flux de données au cours du temps. La quantification et la détection de changements est ainsi l'un des défis majeurs dans le traitement de flux de données qui touche aux questions fondamentales sur la nature des changements, leur importance et leur effet potentiel sur les modèles conceptuels des flux.

La détection de changements, telle que nous l'entendons ici, repose sur l'identification de nouvelles données qui diffèrent d'une manière quelconque de celles qui sont déjà apparues dans un flux de données. Ce procédé est parfois aussi désigné sous le nom de "détection de nouveauté" ou bien "détection d'anomalies". L'idée fondamentale est donc d'apprendre un modèle des données normales du flux étant surveillé jusqu'alors de façon à pouvoir ensuite contrôler et identifier automatiquement les nouvelles données dérivant du modèle appris. Ces nouvelles données pourraient définir un nouveau concept, pas encore connu, ou bien pourraient refléter des anomalies affectant ledit flux (fraudes, intrusions ou encore erreurs de mesure, de fonctionnement, etc). Par ailleurs, des changements peuvent se produire même dans les concepts normaux déjà reconnus par le système du traitement du flux. Ce type de problèmes est très souvent connu sous le nom de "dérive de concept" (cf. Section 2.3.1). La détection de changements est aussi étroitement liée à la détection de données aberrantes, abordée dans le domaine des statistiques. Une donnée aberrante est une donnée qui s'écarte exagérément des autres données et qui peut complètement changer le modèle qui pourrait caractériser les données. Généralement, les méthodes statistiques ne cherchent pas à identifier de manière explicite les données aberrantes mais plutôt à les ignorer, leur objectif étant de créer des modèles robustes qui ne sont pas (ou peu) sensibles à la présence de ce type de données.

Tout cela mène au problème de la classification à partir d'une seule classe qui peut également être perçu comme un problème de classification à deux classes, où chacune des deux classes a une sémantique bien définie, mais seulement des exemples d'apprentissage issus de la classe dite "*classe cible*" ou "*classe positive*", sont disponibles. L'autre classe qui est totalement absente, "*classe négative*", représente toute autre donnée possible n'appartenant pas à la classe cible.

L'absence des exemples négatifs rend la tâche de classification délicate et bien différente de celle de la classification à deux classes. Or, ce problème a souvent été résolu en recourant à des versions adaptées des méthodes traditionnelles de classification à deux ou multiples classes, avec comme conséquence une forte dégradation de ces méthodes. Cette dégradation semble être plus rapide pour des approches discriminatives que génératives (Japkowicz, 1999; Markou et Singh, 2003b). Dans la suite de cette section, nous présenterons les différentes approches de la détection de changements en tenant compte principalement de deux familles : les approches de classification à partir d'une seule classe et les approches de classification non supervisée. Nous verrons aussi comment ces approches peuvent être adaptées au cas de flux de données.

2.4.1 Classification à partir d'une seule classe

Pour remédier au problème d'absence d'exemples négatifs dans ce type de classification, deux directions ont principalement émergé : La première direction cherche à déduire artificiellement des exemples négatifs à partir d'un ensemble de données non étiquetées, tandis que la seconde direction apprend directement à partir des exemples positifs uniquement.

Méthodes de classification à partir d'exemples positifs et d'exemples non étiquetés

L'extraction des exemples négatifs à partir d'un ensemble de données non étiquetées a été abordée dans plusieurs travaux scientifiques (Yu et al., 2003; Liu et al., 2003; Li et Liu, 2003). Les auteurs de ces travaux arguent du fait que des exemples non étiquetés peuvent aisément être rassemblés, par exemple à partir du WWW dans le cadre d'applications Web. Leur objectif ultime est de réduire l'effort manuel d'étiquetage et maintenir simultanément une performance aussi haute que celle des méthodes de classification classiques (c.-à-d. méthodes d'apprentissage à partir des exemples positifs et négatifs étiquetés manuellement). La plupart des propositions adoptent une même stratégie d'extraction d'exemples, constituée de deux étapes :

1. Établir un modèle initial de classification à partir d'exemples positifs uniquement (Yu et al., 2003), ou bien à partir d'exemples positifs et d'exemples non étiquetés en tant qu'exemples négatifs (Li et Liu, 2003; Liu et al., 2003). Ce modèle est ensuite utilisé pour identifier des exemples fortement négatifs à partir des exemples non étiquetés.
2. Appliquer itérativement une méthode de classification binaire pour trouver à chaque itération encore des exemples négatifs à partir des exemples non étiquetés restants. Le meilleur modèle résultant est alors choisi comme une solution finale de classification.

La performance des différentes propositions varie en fonction des méthodes de classification utilisées à chacune des étapes mentionnées ci-avant (voir (Liu et al., 2003) et (Fung et al., 2006) pour plus de détails à ce sujet). Le travail de Fung et al. (2006) fournit une stratégie légèrement différente (PNLH) qui cherche, dans la seconde étape, à extraire aussi bien des exemples positifs à partir des exemples non étiquetés. Les résultats empiriques ont montré que PNLH peut généralement améliorer la performance par rapport à la stratégie précédente, tout particulièrement lorsque le nombre d'exemples positifs est extrêmement faible.

Les défauts principaux de telles stratégies peuvent se résumer en trois points : L'extraction d'exemples négatifs à partir de données non étiquetées est une approche approximative qui nécessite beaucoup de précautions notamment pour s'assurer que des exemples négatifs fiables sont choisis à partir des exemples non étiquetés. Plus spécifiquement, ce type d'approches est fondé sur l'hypothèse que les exemples non étiquetés sont des exemples négatifs bruités, c'est à dire le pourcentage des exemples positifs est très faible (cf. (Yu et al., 2003)), de sorte que le modèle

initial, construit dans la première étape, pourrait identifier des exemples fortement négatifs tout en excluant ceux qui sont positifs[13]. Un autre point est que la plupart des méthodes existantes s'avèrent échouer lorsque le nombre d'exemples positifs est petit, et les exemples non étiquetés doivent être suffisamment nombreux pour pouvoir trouver un bon modèle de classification. Enfin, les itérations successives de la seconde étape rendent la classification beaucoup plus longue qu'un procédé simple de classification.

Méthodes de classification à partir d'exemples positifs uniquement

Les défauts des approches basées sur l'extraction d'exemples négatifs à partir de données non étiquetées présentent bien évidemment des limitations sérieuses dans le cadre des flux de données. Pour cela, l'apprentissage à partir d'exemples positifs uniquement s'avère être un choix évident pour surmonter ces limitations. Ce mode d'apprentissage ne semble cependant pas avoir suscité assez d'attention. Les tous premiers travaux sur cette question ont été explorés dans les domaines de la reconnaissance des formes et la détection de nouveauté (Japkowicz et al., 1995; Japkowicz, 2001). Certains de ces travaux sont brièvement exposées ci-dessous.

Les méthodes neuronales sont historiquement les premières méthodes utilisées pour la détection de nouveauté. En 1976, le filtre détecteur de nouveauté (NDF) a été introduit par Teuvo Kohonen comme un modèle d'orthogonalisation laissant passer uniquement les propriétés (ou variables) nouvelles d'une donnée relativement à un ensemble de données déjà présentées à l'entrée du modèle (les données d'apprentissage) ; les propriétés sont dites nouvelles si elles ne sont pas (ou peu) représentées dans les données d'apprentissage (Kohonen et Oja, 1976).

Une autre approche de détection de nouveauté, aussi neuronale, est présentée dans (Japkowicz et al., 1995). Cette approche est basée sur l'utilisation d'un réseau de neurones auto-associatif de type MLP (Multi-Layer Perceptron, des réseaux multicouches à rétro-propagation de l'erreur). Le réseau auto-associatif (AANN) possède une architecture à trois couches, avec le même nombre de neurones en couches d'entrée et de sortie et peu de neurones cachés en couche centrale. Pendant la phase d'apprentissage, le réseau est entraîné à reproduire en sortie les entrées (exemples positifs) qui lui ont été présentées. Ensuite, la classification est opérée en se basant sur le fait que les données positives seront plus exactement reconstruites que les données négatives.

Plus tard, des différentes versions des méthodes de type SVM ont été spécialement adaptées à la classification à partir d'une seule classe (Schölkopf et al., 2001; Tax et Duin, 2001; Manevitz et Yousef, 2001). L'idée de base derrière l'approche 1-SVM introduite par Schölkopf et al. (2001) est de transformer les données positives originelles de l'espace d'entrée vers un esapce de dimension supérieure, grâce à une fonction noyau, et de les séparer de l'origine, la donnée négative unique, avec une marge maximum. En sus des paramètres originaux de SVM, 1-SVM nécessite de fixer a priori la valeur d'un paramètre indiquant le pourcentage des données positives autorisées à se situer en dehors de la description de la classe positive. Ceci rend 1-SVM plus tolérant au bruit dans les données positives (c.-à-d. les données aberrantes). Cependant, le choix d'une valeur adéquate pour ledit paramètre n'est pas toujours intuitif, et a une influence primordiale sur la

[13]Notons que, dans le cas où beaucoup de données positives seraient incorrectement extraites en tant qu'exemples négatifs, la qualité du modèle de classification construit dans la première étape sera très médiocre et se dégradera certainement avec les phases d'itération de la seconde étape. C'est pour cette raison que nous trouvons que l'utilisation d'une méthode de classification à une classe dans la première étape (comme 1-SVM dans (Yu et al., 2003)) est un bon choix pour éviter ce genre de problèmes.

performance de 1-SVM.

Pour sa part, l'approche de Tax et Duin (2001), SVDD, cherche à trouver une hypersphère de volume minimum qui englobe toutes ou la plupart des données positives (la contrainte de volume minimum est nécessaire pour réduire au minimum la possibilité d'intégration des données négatives dans l'hypersphère). Cette approche optimise automatiquement le paramètre de 1-SVM mentionné ci-dessus en utilisant des données non étiquetées générées artificiellement et distribuées uniformément dans une hypersphère autour des données positives. Néanmoins, cette méthode d'optimisation ne s'applique pas dans le cas d'espaces de dimensions élevées (plus de 30 dimensions).

Une version légèrement différente de 1-SVM (Outlier-SVM) est également introduite dans (Manevitz et Yousef, 2001). La proposition est de traiter non seulement l'origine en tant que donnée négative mais aussi toutes les données qui sont "assez proches" de l'origine. L'identification de ces dernières est fondée sur l'hypothèse que les données qui partagent très peu de variables de l'espace de représentation des données positives (c'est-à-dire, les vecteurs de ces données ont très peu de variables avec des valeurs différentes de 0) ne constituent pas de bons représentants de la classe positive et peuvent être traitées comme étant des données aberrantes. Globalement, les résultats obtenus avec cette approche apparaissent moins satisfaisants que ceux de 1-SVM.

Dans (Manevitz et Yousef, 2001), 1-SVM a aussi été comparé à d'autres méthodes de classification : Rocchio, k plus proches voisins (KNN), Bayes Naïfs (NB), réseaux de neurones auto-associatifs (AANN). Les résultats empiriques ont montré que 1-SVM et AANN sont meilleurs que les autres méthodes. En revanche, dans (Lee et Cho, 2006), les résultats obtenus sur six corpus de test suggèrent que 1-SVM aurait toujours une meilleure performance que celle de AANN. Nous reverrons plus en détails certaines des méthodes mentionnées ci-dessus dans le chapitre 4 consacré au problème de classification à partir d'une seule classe.

2.4.2 Clustering et méthodes neuronales

Les méthodes de clustering peuvent également apporter leurs propres solutions au problème de la détection des changements, ceci en identifiant des données qui ne peuvent être classées dans aucun des clusters construits à partir des données normales ou positives. De telles méthodes sont souvent fondées sur un calcul de distance entre les clusters et les données. Dans un tel cas, il s'agit en particulier de déterminer le bon seuil au-delà duquel une donnée sera considérée comme inhabituelle ou anormale. Les approches basées sur les cartes auto-organisatrices de Kohonen sont des exemples de ce type de méthodes, voir, à titre d'exemples, (Harris, 1993), (Ypma et Duin, 1998), et (Emamian et al., 2000). D'autres méthodes adaptatives raisonnent sur la nouveauté de données en termes de leur contribution à la création de nouveaux clusters. Les exemples types sont : Le réseau GWR de Marsland et al. (2002), et les réseaux ART (Moya et al., 1993).

Un état de l'art très complet sur les différents méthodes de détection de nouveauté a été publié en deux partie dans (Markou et Singh, 2003a) et (Markou et Singh, 2003b), la première partie étant sur les méthodes statistiques et la deuxième sur les méthodes neuronales.

2.4.3 Méthodes des flux de données

Jusqu'à présent, nous avons vu les méthodes de détection de changements telles qu'elles existent dans la littérature. Très peu de celles-ci sont adaptées à un fonctionnement continu en ligne, et par conséquent, elles ne sont pas directement applicables dans le contexte des flux de données. Un algorithme de détection de changements survenant sur un flux de données exige que :

1. Le délai entre un véritable point de changement et sa détection soit minimal ;
2. Le taux de fausses détections et celui de non détections ou omissions soient réduits au minimum ;
3. Les modèles des flux de données soient correctement et efficacement mis à jour.

Lors de la mise en place d'un système de détection de changements survenant sur un flux de données, une question supplémentaire se pose : Quels sont les données qui serviront de base pour détecter un changement ? Confrontées à cette question, la plupart des méthodes orientées flux de données recourent essentiellement aux fenêtres temporelles. Dans ce cadre, deux fenêtres sont utilisées : une "*fenêtre de référence*" comprenant les données positives et une "*fenêtre courante*" décalée de la fenêtre de référence d'un intervalle prédéterminé. La fenêtre courante glisse avec l'arrivée de nouvelles données, et la fenêtre de référence est souvent mise à jour toutes les fois qu'un changement est détecté. Une telle démarche a été adoptée par exemple dans (Kifer et al., 2004) et (Spinosa et al., 2006).

En employant des fenêtres courantes de différentes tailles, il est possible de détecter des changements à court et à long terme. Dans certaines applications, des fenêtres multiples de différentes tailles, nommées "*fenêtres élastiques*", sont simultanément utilisées pour pouvoir mesurer l'étendue des changements (Zhu et Shasha, 2003). Néanmoins, une telle méthode n'est pas conçue pour des calculs en temps réel.

2.4.4 Synthèse

Au terme de cette section, il apparaît que le problème de la détection de changements dans les flux de données est encore peu investi jusqu'à aujourd'hui par manque de méthodes efficaces et adéquates. Dans ce nouveau contexte, les particularités des flux de données dépassent en effet largement les capacités des méthodes classiques dédiées à la classification à partir d'une seule classe. La complexité du problème tient avant tout — si l'on exclut les difficultés inhérentes liées à l'absence d'exemples négatifs d'apprentissage — à la nécessité d'un apprentissage en ligne avec une capacité d'adaptation permanente et en temps quasi réel. Parmi toutes les méthodes existantes, le modèle du filtre détecteur de nouveauté (NDF) est le seul qui nous paraissait théoriquement susceptible de satisfaire à ces contraintes. En effet, le filtre détecteur de nouveauté est relativement facile à mettre en œuvre selon un mode de fonctionnement en ligne et sans répétition d'apprentissage. Cependant, la performance du modèle s'est révélée insatisfaisante et sensiblement inférieure à celle de la plupart des autres méthodes de résolution du problème de la classification à partir d'une seule classe. C'est pour cette raison, et d'autres encore, que nous avons apporté des modifications à la règle d'apprentissage du modèle NDF, qui sont reportées dans un nouveau modèle, ILoNDF. Nous reviendrons en détail sur ces deux modèles, leurs spécificités, et leur performance dans la seconde partie du manuscrit.

2.5 Conclusion

Nous avons présenté dans ce chapitre un état de l'art des méthodes et techniques d'analyse des flux de données. Le but principal de la présentation était de mettre l'accent sur les aspects fondamentaux relatifs au traitement de flux de données et de montrer, par des exemples concrets, pourquoi les méthodes qui y sont actuellement apportées sont encore loin d'être entièrement satisfaisantes. Nous avons ainsi abordé de manière assez détaillée deux grandes directions adoptées pour faire face aux défis majeurs qui se posent dans le cadre de la conception et de la mise en œuvre d'un modèle de flux de données.

La première direction s'intéresse particulièrement à la mise au point de méthodes efficaces et rapides de synthèse consistant à construire des résumés de l'historique d'un flux de données pour se substituer aux données originales issues du flux. De cette façon, les approches classiques d'analyse et de fouille de données peuvent y être appliquées. Parmi les méthodes de synthèse présentées, nous avons pu constater que les méthodes d'échantillonnage et de clustering sont les plus appropriées au cas des flux de données multidimensionnelles. Toutefois, les méthodes d'échantillonnage peuvent entraîner des pertes d'information plus importantes que les méthodes de clustering, notamment en ce qui concerne les informations relativement rares dans les flux.

La seconde direction cherche plutôt à étendre les approches classiques ou bien à proposer de nouvelles approches appropriées au traitement de flux de données. Les différentes approches proposées dépendent directement des contraintes spécifiques liées à leurs propres applications. Nous avons délibérément séparé les approches de classification supervisée et non supervisée, selon les connaissances dont on dispose a priori sur les données. Tout en présentant un panorama sur les différentes méthodes relatives à ces deux approches, nous avons plus particulièrement insisté sur le cas des flux de données non stationnaires. Dans ce dernier cas, nous avons souligné le besoin de traiter les changements survenant dans un flux de données au cours de temps.

Du fait de leurs propriétés intrinsèques, les flux de données couvrent de très nombreux domaines d'applications. Comme nous l'avons mentionné dans l'introduction, le contexte applicatif de cette thèse est celui du traitement des données fortement multidimensionnelles, et en particulier des données documentaires. Le but du chapitre suivant est de présenter les aspects spécifiques au traitement de ce type de données, les méthodes qui y sont relatives, et quelques exemples d'applications, à savoir le filtrage d'informations et la détection et le suivi de thèmes dans les flux documentaires.

Chapitre 3

Aspects spécifiques à l'analyse des flux documentaires

L'internet et les services de production et de diffusion d'informations en ligne, tels que les agences de presse et les médias électroniques, les newsgroups, la messagerie électronique, etc., donnent accès à des ressources documentaires quasi illimitées, qui peuvent être modifiées, deviennent obsolètes voire disparaissent au cours de temps. La croissance incontournable de données documentaires dans des environnements dynamiques, hétérogènes et distribués, évoque la nécessité de la mise au point de nouveaux outils d'aide au repérage et à la délivrance des informations pertinentes au sein des flux de données documentaires.

Historiquement, les premiers travaux sur les flux documentaires datent de plus d'une vingtaine d'années. En particulier menés en filtrage d'informations, leur but est d'extraire, à partir d'un flux dynamique d'informations, celles qui sont susceptibles d'intéresser un utilisateur ou un groupe d'utilisateurs ayant des besoins en information relativement stables. La pertinence des informations extraites, leur opportunité et leur adaptation aux besoins des utilisateurs constituent des facteurs clés du succès des systèmes de filtrage d'informations. Les domaines d'application du filtrage d'informations sont très nombreux et d'une grande importance. Citons, entre autres : le commerce électronique, les services d'aide personnalisée à la navigation sur le Web qui bloquent les informations non pertinentes, et les services de filtrage des mails.

Le filtrage d'informations s'inscrit dans le cadre plus général d'accès personnalisé à l'information qui constitue un enjeu capital pour répondre aux problèmes de la surcharge d'informations et pour discriminer l'information pertinente de l'information secondaire ou non pertinente en fonction des besoins spécifiques des utilisateurs. Les systèmes de personnalisation de l'information se basent sur la notion de profils utilisateurs. Ces profils modélisent les besoins et les centres d'intérêts des utilisateurs en termes de filtrage et de qualité de l'information, de modalités d'accès aux systèmes et de livraison des résultats. L'essentiel de ces connaissances est fourni par les utilisateurs, le reste est acquis par le système en recourant à des algorithmes d'apprentissage à partir des connaissances préalables. La nature de ces profils dépend donc de plusieurs facteurs dont le contexte d'utilisation, ce qui met en cause deux modes principaux de filtrage : le filtrage basé sur le contenu et le filtrage collaboratif. Dans le filtrage basé sur le contenu, encore appelé filtrage cognitif, le profil utilisateur est représenté par des mots clés (ou termes d'indexation). Par contre, dans le filtrage collaboratif, le profil est représenté par les annotations que l'utilisateur a attribuées à des documents qu'il a reçus au préalable. La sélection des documents

pertinents dans le cas du filtrage basé sur le contenu repose sur une mise en correspondance entre le profil utilisateur et le contenu des documents entrant en flux, alors qu'elle est plutôt basée sur l'évaluation des annotations attribuées aux documents par les utilisateurs pour en créer des communautés ; chaque utilisateur reçoit les documents jugés pertinents pour sa communauté.

Plus récemment, en 1996, la veille automatique des flux documentaires a été initiée par la DARPA[14] à travers le projet "*détection et suivi d'événements*" TDT[15]. Ce projet a défini trois tâches de base sur un flux documentaire (ex. des dépêches d'agences de presse ou journaux télévisés) : 1) la segmentation d'un flux documentaire, si continu, en segments homogènes (histoires) ; 2) la détection de nouveaux événements survenant sur un flux documentaire ; et 3) le suivi d'un événement déjà détecté (Wayne, 1998).

Ce chapitre aborde les problématiques relatives aux flux documentaires mentionnées ci-avant en analysant des approches existantes et en expliquant leur fonctionnement. Il est structuré en trois parties principales. La première partie est consacrée aux étapes générales, mais fondamentales, de l'analyse du contenu des documents et du choix d'un modèle de représentation adéquat des données à traiter, en l'occurrence les données textuelles. La deuxième partie est consacrée aux concepts clés de la personnalisation de l'information, en se concentrant sur les étapes nécessaires à l'aboutissement d'un filtrage basé sur le contenu, qui entre dans l'un des cadres applicatifs principaux de ce travail de thèse. La troisième partie porte sur la problématique de la détection et du suivi de thèmes dans les flux de données documentaires.

3.1 Prétraitement et représentation des documents

Dans les systèmes informatiques, un document est considéré comme étant un support véhiculant l'information correspondant au contenu sémantique du document. La phase d'indexation permet de capturer cette information et de la représenter selon un modèle : le modèle de documents. La représentation de ce contenu sémantique s'opère le plus souvent sous forme condensée d'un ensemble de termes significatifs, éventuellement pondérés, appelés variables ou termes d'indexation. Ainsi, les systèmes informatiques ne manipulent pas les documents mais leur représentation. L'indexation comprend habituellement deux opérations successives : une extraction des variables et une sélection des variables (van Rijsbergen, 1979). À l'issue de ces opérations, le nombre de variables résiduelles, formant l'espace de représentation des documents, peut malgré tout rester très grand. Dans un tel cas, certaines opérations supplémentaires peuvent alors être appliquées pour réduire la dimension de l'espace de représentation des documents.

Le but de cette section est de décrire, dans un premier temps, les phases préliminaires de l'indexation. Nous examinerons, ensuite, les différents modèles de représentation des documents permettant de spécifier la présence, l'absence ou encore la proximité de termes dans les documents. Enfin, nous évoquerons les méthodes dédiés à la réduction de la dimensionnalité de l'espace de représentation des documents.

[14]U.S. Defense Advanced Research Projects Agency, http ://www.darpa.mil
[15]Topic Detection and Tracking, http ://projects.ldc.upenn.edu/TDT

3.1.1 Indexation

L'indexation est l'opération consistant à extraire les termes les plus pertinents qui caractérisent l'information contenue dans un document ou un ensemble de documents et à les représenter sous forme d'un modèle conceptuel, concis et précis. Cette opération peut s'appuyer sur deux types d'analyse et d'interprétation du contenu des documents — une analyse superficielle et une analyse en profondeur — dont la distinction est établie en fonction du niveau de spécialisation du processus d'analyse. En principe, l'analyse superficielle ne permet d'isoler que des caractéristiques élémentaires des documents au risque que celles-ci soient relativement peu représentatives du contenu réel de ces documents. De son côté, l'analyse en profondeur, qui est en général de type syntaxico-sémantique, ne s'avère pas adaptée aux modèles de représentation usuels des documents. En effet, du point de vue théorique, plus les documents sont spécialisés dans un domaine de connaissances pointu, plus l'analyse devra être précise, c'est à dire qu'elle doit être établie en profondeur. Toutefois, les contraintes d'efficacité, et les difficultés liées à l'interprétation du contenu des documents, amènent très souvent les systèmes informatiques à utiliser un modèle conceptuel simplifié et à procéder à une analyse superficielle plutôt qu'à une analyse en profondeur des documents. Nous privilégions donc ici la description de l'analyse superficielle des documents.

L'analyse superficielle peut être basée sur des vocabulaires des langages contrôlés ou des vocabulaires des langages libres, d'où les deux types d'indexation, l'*indexation contrôlée* et l'*indexation libre* ou *plein-texte* (Salton, 1989).

Dans le cas d'une indexation contrôlée, une liste de termes d'indexation susceptibles de représenter le ou les thèmes principaux abordés dans les documents, est construite a priori. L'indexation consiste alors à rechercher ces termes d'indexation dans les documents. Les termes d'indexation d'un vocabulaire contrôlé peuvent aussi être organisés sous la forme d'un thésaurus, ce qui rend possible une indexation par des termes plus généraux que ceux trouvés dans les documents grâce à des relations de synonymie et de généricité-spécificité caractéristiques de la structure hiérarchique du thésaurus. Ce type d'indexation est cependant limité par le coût de définir des vocabulaires contrôlés et des thésaurus correspondant au contenu des documents à indexer. De plus, dans un cadre dynamique où le contenu des documents peut évoluer avec le temps, une mise à jour des vocabulaires contrôlés et des thésaurus est nécessaire pour assurer une couverture suffisante du contenu des documents.

De son côté, l'indexation libre permet de construire une description plus exhaustive du contenu des documents. Dans ce type d'indexation, les termes d'indexation sont choisis a posteriori en fonction d'une analyse du contenu des documents. La forme la plus simple d'une telle analyse est l'extraction de tous les termes simples, ou uni-termes, présents dans les documents à l'exception de certains termes, dits "mots vides" (ou "stopwords" en anglais ; ex. articles, prépositions, pronoms, etc). Des méthodes d'analyse plus sophistiquées peuvent aussi être utilisées dans le but d'extraire certains concepts significatifs des faits présents des documents. Dans ce contexte, un concept est exprimé comme une association de termes simples (par exemple, le concept "une base de données" associe les termes "base" et "données"). Dans les deux cas, aucune subjectivité n'est introduite lors de la création des termes d'indexation.

Ce type d'indexation a pour inconvénient de produire une liste de termes d'indexation de taille très importante. Pour remédier, partiellement, à ce problème, il est possible de procéder

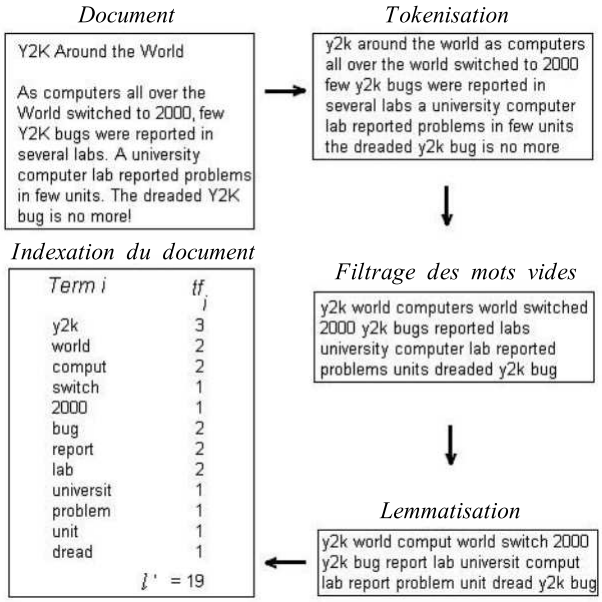


FIG. 3.1 – Les étapes principales d'une indexation libre d'un document textuel.

à une opération de lemmatisation (Stemming en anglais) pour éliminer d'éventuelles variations morphologiques des termes. Dans la version la plus simple, il s'agit de déterminer les racines lexicales des termes et de considérer uniquement ces racines plutôt que les termes entiers, par exemple, les termes "laughing" et "laughter" peuvent être tronqués en "laugh". Toutefois, même après le recours à une opération de lemmatisation, le nombre de termes reste souvent très important, ce qui donne très souvent lieu à des opérations de sélection et de réduction des termes (cf. Section 3.1.3).

Dans les expériences présentées dans cette thèse, nous avons fait le choix d'utiliser une indexation libre des documents en ne retenant que des termes simples comme termes d'indexation. Nous appliquons également une opération de lemmatisation proposée par Porter (1980) pour remplacer les termes simples par leur racine. L'intérêt d'une telle indexation réside principalement dans sa rapidité qui est tout fait adaptée à des volumes très importants mais également dans le fait qu'elle ne s'appuie pas sur une connaissance préalable approfondie, des thèmes traités dans les documents et peut, ainsi, être directement appliquée aux collections de documents couvrant divers sujets thématiques.

La qualité de l'indexation peut être évaluée à partir de différents critères, en forte interaction les uns avec les autres. On peut en citer au moins deux principaux, l'exhaustivité et la spécificité de l'indexation (van Rijsbergen, 1979).

- L'exhaustivité peut être regardée comme proportionnelle au nombre de thèmes couverts par les termes d'indexation qui sont choisis pour représenter le contenu des documents. On cherche alors à ce que la représentation des documents soit la plus complète possible. L'exhaustivité détermine ainsi le taux de rappel lors de la recherche ou du filtrage de documents : Une forte exhaustivité se traduit par de forts taux de rappel, mais aussi par de faibles taux de précision.

- La spécificité peut, de son côté, être regardée comme proportionnelle au niveau de précision et de détail du vocabulaire de l'indexation. Plus les termes d'indexation sont précis et particuliers aux documents, plus l'indexation est considérée comme spécifique. La spécificité des termes d'indexation offre une meilleure discrimination entre les documents portant sur différents thèmes, au risque de rendre cette fois les documents difficiles à retrouver. Une forte spécificité se traduit donc par de forts taux de précision, mais aussi par de faibles taux de rappel.

Une bonne indexation permet de représenter le contenu des documents par des termes d'indexation pertinents qui sont ni trop exhaustifs ni trop spécifiques.

3.1.2 Modèles de représentation des documents

Le modèle de représentation des documents joue un rôle central dans les systèmes informatiques. Il s'agit de donner une représentation interne aux termes obtenus lors de la phase d'indexation. Un modèle de représentation précise également comment sélectionner et ordonner les documents qui répondent aux besoins d'informations des utilisateurs. Les modèles de représentation des documents sont principalement de trois types, initialement connus dans le domaine de recherche d'informations :

1. Les modèles fondés sur la théorie des ensembles, dont le modèle le plus connu est le modèle booléen ;
2. Les modèles algébriques ou vectoriels qui utilisent des mesures de similarité dans un espace vectoriel engendré par les termes d'indexation ;
3. Les modèles probabilistes fondés sur la théorie des probabilités PRP (Probability Ranking Principle) (Robertson, 1977), tels que les réseaux bayésiens et les réseaux d'inférence.

Dans ce qui suit, nous présenterons brièvement le modèle booléen et le modèle booléen étendu à l'origine des premiers systèmes de recherche d'informations. Nous détaillerons ensuite le modèle vectoriel ainsi que quelques-unes des extensions et améliorations qui y sont apportées.

Le modèle booléen

Ce modèle a été introduit par Salton et McGill (1983). Un document y est représenté par un sous-ensemble de termes d'indexation associés, tous les termes étant considérés comme des propriétés vraies, au sens booléen, pour le document. Un profil descriptif d'un besoin en informations, P, est composé de termes reliés par des opérations logiques, à savoir ET et OU, et $SAUF$. Pour qu'un document D corresponde à un profil P, il faut que l'implication suivante soit valide : $D \Rightarrow P$, ce qui veut dire que le document D satisfait le besoin en informations exprimé à travers le profil P. Par exemple, soit un besoin d'informations exprimé sous la forme

$$P = (t_1\ ET\ t_2)\ OU\ (t_3\ ET\ (SAUF\ t_4))$$

La liste de documents L qui y sont pertinents peut s'écrire sous la forme ensembliste suivante :

$$L = (L_{t_1} \cap L_{t_2}) \cup (L_{t_3} \setminus L_{t_4})$$

où L_{t_i} correspond à l'ensemble de documents indexés par le terme t_i.

Bien que simple, ce modèle permet uniquement de distinguer la pertinence d'un document vis-à-vis d'un profil, sans aucune notion de pondération de celle-ci de sorte à pouvoir établir

une relation d'ordre de pertinence entre les différents documents selon leur correspondance à ce profil. Ainsi, le nombre de documents retenus n'est pas facile à contrôler.

Pour remédier à ce défaut, le modèle booléen étendu a également été introduit par G. Salton (Salton et McGill, 1983). Il s'agit d'une extension du modèle booléen tenant compte d'une pondération des termes dans les documents. Ce modèle est cependant sujet à la même limitation que le modèle booléen du fait des règles de mise en correspondance entre les documents et le profil de type "tout ou rien" des opérateurs *Min* et *Max*, utilisées au lieu des opérateurs *et* et *ou* dans le modèle booléen. Tout en y ajoutant le fait que la formulation d'un besoin d'informations sous forme booléenne n'est pas facile à maîtriser. C'est pour ces raisons que ce type de modèles n'a pas connu beaucoup d'applications pratiques.

Le modèle vectoriel

Le modèle vectoriel a également été mise en œuvre par Salton et al. (1975) dans le système de recherche documentaire SMART. Selon ce modèle, les documents et les profils descriptifs des besoins en informations sont représentés de manière unifiée par des vecteurs de termes pondérés de N dimensions, N étant le nombre de termes d'indexation des documents.

La pondération des termes a pour rôle d'attribuer une valeur de pertinence w_{td} à chaque terme d'indexation t utilisé dans la représentation d'un document d dans une collection de documents. Le calcul de cette valeur peut suivre différentes possibilités, mais, d'une manière générale, cette valeur dépend de trois facteurs, soit : 1) une mesure de la pertinence du terme pour le document, 2) une mesure de la pertinence du terme pour la collection, et 3) un facteur de normalisation.

Le schema de ponderation le plus souvent utilisé est *TFIDF* (Term Frequency - Inverse Document Frequency) (Salton et Buckley, 1988). Ce score permet d'accorder un poids de pertinence au terme en fonction de sa fréquence dans le document (TF) pondérée par la fréquence d'apparition du terme dans toute la collection (IDF), soit :

$$TFIDF_{td} = TF_{td} \times \log_2(\frac{N}{n_t}) \tag{3.1}$$

où N représente le nombre total de documents ; et n_t représente le nombre de documents indexés par le terme t. Le terme TF_{td} de Eq. 3.1 correspond généralement à la fréquence d'occurrence du terme t dans le document d (tf_{td}), mais il peut aussi prendre la valeur d'une fonction de la fréquence d'occurrence du terme t dans le document d. Les fonctions les plus couramment utilisées sont les suivantes :

- Une fonction binaire où TF_{td} vaut 1 si le terme est présent dans le document, 0 s'il ne l'est pas. Cette fonction a souvent été utilisée comme une base de comparaison par rapport aux autres types de pondération proposés ;
- Une fonction logarithmique de type :

$$1 + \log_2(tf_{td}) \tag{3.2}$$

  Cette fonction (Salton et Buckley, 1988) a été introduite dans le but d'atténuer les effets de larges différences entre les fréquences d'occurrence des termes dans le document, de sorte qu'une forte fréquence d'occurrence d'un terme ne soit pas prédominant par rapport à une faible fréquence d'occurrence de plusieurs termes.

- Une fonction à fréquence augmentée normalisée (Croft, 1983), qui peut être définie comme suit :

$$w_{td} = 0.5 + 0.5(\frac{tf_{dt}}{max\ tf_{dt}}) \tag{3.3}$$

où tf_{dt} représente la fréquence d'apparition du terme t dans le document d, et $max\ tf_{dt}$ représente le maximum de tf_{dt} des termes du document d. Cette fonction, comme la fonction logarithmique, réduit la différence entre valeurs de fréquence des différents termes du document.

Une normalisation de type cosinus des vecteurs *TFIDF* des documents est habituellement utilisée afin de diminuer l'effet de divergence des poids dû à la variation de la taille des documents[16] (Salton et Buckley, 1988). Le schéma *TF-IDF* normalisée est une représentation très utilisée aussi bien en recherche et filtrage d'informations qu'en classification (Salton, 1971; Joachims, 1998).

Dans le modèle vectoriel, la stratégie de mise en correspondance consiste à retrouver les vecteurs documents qui s'approchent le plus du vecteur profil. L'unicité de représentation entre les documents et les profils permet de définir des mesures de similarité à utiliser lors de la mise en correspondance entre documents et profils, mais également de comparer des documents et des profils entre eux. Le mesure de similarité la plus souvent utilisée est la *similarité cosinus* (Salton, 1971), soit :

$$dCos(P, D) = \frac{P \cdot D}{\|P\|\|D\|} = \frac{\sum_{i=1}^{N} p_i d_i}{\sqrt{\sum_{i=1}^{N} p_i^2 \sum_{i=1}^{N} d_i^2}} \tag{3.4}$$

L'équation 3.4 mesure le cosinus de l'angle formé par le vecteur profil P et celui du document D. Une valeur de 1 signifie que les deux vecteurs sont identiques (les deux vecteurs ont les mêmes termes, avec les mêmes poids), une valeur de 0 signifie qu'aucun terme n'est commun aux deux vecteurs.

La stratégie de mise en correspondance admet un seuil de pertinence S tel que seuls les documents D qui ont une similarité supérieure à ce seuil sont considérés comme pertinents au regard du profil, ce qui peut formellement s'écrire :

$$D = \{D_i \mid dCos(P, D_i) > S\}$$

Les documents D peuvent aussi être rangés dans l'ordre croissant ou décroissant de pertinence en fonction de la valeur de $dCos(P, D_i)$.

Traitement des relations sémantiques dans le modèle vectoriel

Le modèle vectoriel repose sur l'hypothèse forte que les vecteurs de base de l'espace de représentation sont orthogonaux, ce qui implique que les termes d'indexation soient indépendants. Une telle hypothèse peut toutefois être infirmée par la détection d'une simple relation de synonymie entre deux termes. Wong et Raghavan (1984) traduisent cette constatation par la réflexion que

[16] En général, les documents longs ont tendance à utiliser de façon répétée les mêmes termes, et à utiliser plus de termes pour décrire un sujet thématique. Par conséquent, les fréquences des termes dans ces documents seront plus élevées que ceux dans les documents plus courts, et leurs similarités avec un profil descriptif d'un besoin en informations seront également plus fortes. La normalisation est donc utilisée pour attribuer aux documents la même chance d'être sélectionnés indépendamment de leur longueur.

toute mesure de similarité devrait prendre en compte l'influence de la corrélation entre les termes d'indexation. Or, cette constatation a été ignorée du fait que la remise en question des mesures de similarité existantes ne semble pas aussi évidente. D'autres approches ont alors été proposées pour traiter les relations sémantiques entre termes. Deux types de relations ont principalement été considérés, à savoir, la synonymie (une même définition renvoie à plusieurs termes dans différents contextes) et la polysémie (plusieurs définitions sont associées au même terme) des termes. Cette ambiguïté des termes mène évidemment à de faibles taux de rappel et de précision lors de la recherche ou du filtrage d'informations.

Une des premières approches, pour traiter ce genre de problème, est de procéder à l'expansion des vecteurs profils en y ajoutant des synonymes ou des termes voisins aux termes présents dans ces profils, à l'aide d'un dictionnaire de synonyme, ou un thésaurus (Piace, 91). D'autres approches visent à étendre le modèle vectoriel de manière plus explicite en repérant des relations sémantiques entre les divers termes des documents. Nous présentons deux exemples de ces approches : la technique d'indexation sémantique latente et les réseaux neuronaux sémantiques.

Indexation Sémantique Latente (LSI)

La LSI est une extension du modèle vectoriel standard dans laquelle les dépendances sémantiques entre les termes sont explicitement prises en considération lors de la représentation des documents. Les relations de dépendance sont extraites en tirant profit du contenu des documents (Deerwester et al., 1990). Pour ce faire, LSI applique une technique de réduction de dimensions — la "*Décomposition en Valeurs Singulières*" (SVD) (Golub et Loan, 1991) — sur une matrice $(termes \times documents)$, X, d'ordre $(m \times n)$, dans laquelle chaque élément w_{ij} représente le nombre d'occurrences du terme t_j dans le document d_i. La SVD de X s'écrit comme produit de trois matrices :

$$X = T_{(m\times m)}\ S_{(m\times n)}\ D^T_{(n\times n)} \tag{3.5}$$

Les matrices T et D sont orthogonales, contenant respectivement les vecteurs propres de XX^T et de X^TX ; et la matrice S est une matrice diagonale des valeurs singulières, qui sont elles-mêmes les racines des valeurs propres de XX^T, rangées par ordre décroissant. Seules les k plus grandes valeurs singulières non nulles sont employées pour construire une version rapprochée de la matrice X, $\hat{X}$:

$$\hat{X} = T_{(m\times k)}\ S_{(k\times k)}\ D^T_{(k\times n)} \tag{3.6}$$

où $T_{(m\times k)}$ est la matrice des vecteurs de termes représentés dans l'espace des k plus grandes valeurs singulières, permettant de représenter les termes dans ce nouvel espace, et $D^T_{(k\times n)}$ est la matrice qui permet de représenter les documents dans le nouvel espace des k plus grandes valeurs singulières.

Dans le cadre du filtrage d'information, (Foltz, 1990) a démontré que LSI peut améliorer la performance du processus de filtrage par rapport à celle obtenue avec un modèle vectoriel standard. Cependant, il faut préciser que le choix de la valeur de k conditionne largement son efficacité.

Les réseaux neuronaux sémantiques

Les réseaux neuronaux sémantiques sont étroitement associés à l'analyse et à la représentation de documents. Ils peuvent être considérés comme un type spécial de réseaux neuronaux dans

lequel les neurones représentent des concepts sémantiques, i.e. des classes des termes possédant des propriétés communes, et les connexions représentent des liens sémantiques entre ces concepts.

Un exemple de tels réseaux est les cartes sémantiques auto-organisatrices (SSOM) (Ritter et Kohonen, 1989). Selon cette approche, chaque terme de l'espace de représentation est codé par un vecteur x_i de n valeurs aléatoires. Le contexte du terme est ensuite formé en calculant son vecteur moyen de contexte $X(i) \in R^{3n}$ donné par :

$$X(i) = \begin{pmatrix} \mathbb{E}(x_{i-1}/x_i) \\ \epsilon\ x_i \\ \mathbb{E}(x_{i+1}/x_i) \end{pmatrix} \tag{3.7}$$

où $\mathbb{E}$ est l'espérance mathématique et ϵ est un petit nombre. Deux termes apparaissant souvent dans les mêmes contextes auront donc des vecteurs similaires de contexte et seront associés à la même classe. Les documents peuvent ainsi être indexés en utilisant les concepts sémantiques associés aux classes de la carte au lieu des termes simples d'indexation.

Une autre approche est basée sur un réseau neuronal de type Hopefield dans le but d'indexer sémantiquement un ensemble de documents en réduisant l'impact des termes bruités (redondants, ou non pertinents). Elle identifie l'ensemble de termes (sortie du réseau) qui sont le plus sémantiquement appropriées à un ensemble de termes d'indexation (entrée du réseau) produits par un mécanisme préalable d'indexation automatique (Chen et al., 98).

3.1.3 Réduction de dimensionnalité

Dans les grandes collections de documents, le vocabulaire d'indexation peut s'avérer corrélativement très grand. Chaque mot de vocabulaire, ou terme d'indexation, exige une dimension singulière de l'espace de représentation des documents. Le coût de manipulation des vecteurs de grandes dimensions peut s'avérer rédhibitoire en terme de temps de calcul et d'espace mémoire. Par ailleurs, la performance des algorithmes d'apprentissage, contrairement à ce que l'intuition pourrait faire croire, n'augmente pas indéfiniment avec la dimension des vecteurs, mais se met à décroître fortement au-delà d'un certain nombre de termes, ce qui renvoie au problème de la "malédiction de la dimensionnalité" (curse of dimensionality) (Bellman, 1961).

Il est donc question de pouvoir réduire efficacement le nombre de termes extraits par un mécanisme d'indexation libre, en en sélectionnant ceux qui seront finalement utilisés dans la représentation des documents. Pour cela, il existe des méthodes connues de sélection de termes qui peuvent être appliquées ; on en distingue deux grandes familles : les méthodes supervisées et non supervisées. Les méthodes non supervisées sont utilisées si aucune connaissance a priori n'est disponible sur les classes auxquelles appartiennent les documents. C'est particulièrement le cas, entre autres, des systèmes de détection d'événements précédemment cités. Les méthodes supervisées sont plus adaptées lorsque les classes sont connues a priori, comme c'est souvent le cas des systèmes de filtrage d'information, où les documents sont séparés en deux classes contenant les documents pertinents et non pertinents au regard du besoin d'informations de l'utilisateur.

Sélection des termes d'indexation : cas non supervisé

L'analyse statistique des documents textuels a montré que la distribution de fréquence des termes dans les documents n'est pas uniforme : certains apparaissent fréquemment, d'autres

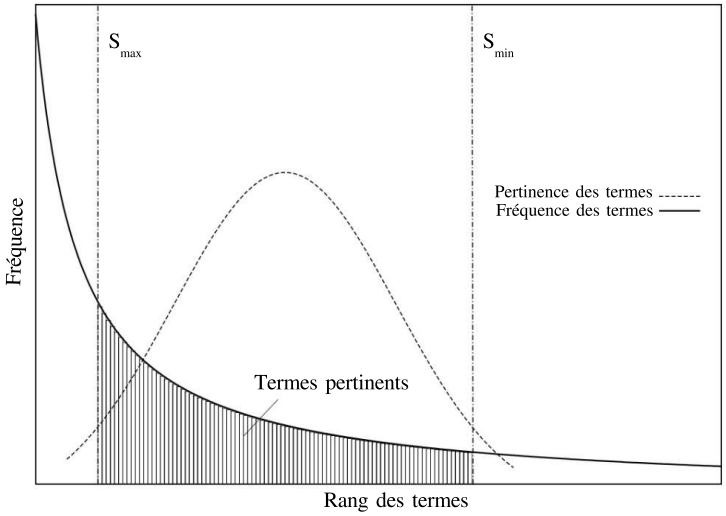


FIG. 3.2 – La loi de Zipf et la conjecture de Luhn, adapté de (Schultz et Luhn, 1968), page 120.

rarement. Cette distribution a été étudiée en premier par Zipf (1949), les résultats de cette étude ont abouti à la découverte d'une loi empirique connue sous le nom de "*loi de Zipf*". Cette loi énonce que la fréquence d'un terme dans un texte est inversement proportionnelle à son rang par rapport à sa fréquence d'apparition, ce que traduit la formule :

$$frquence \times rang = constante$$

En s'appuyant sur cette loi, Luhn (1958) émet de son côté l'hypothèse que les termes les plus fréquents et à l'inverse ceux les moins fréquents, rares, sont des termes non pertinents (non informatifs et/ou non discriminants). En fait, les termes qui apparaissent très fréquemment sont, comme nous l'avons précédemment souligné, les mots vides, tels que les pronoms, les prépositions, les conjonctions, etc. Ces termes peuvent être supprimés du fait, d'une part, qu'ils ne sont pas informatifs en tant que termes isolés, et, d'autre part, qu'ils sont distribués sur l'ensemble des documents et ne feront aucune discrimination entre ces documents. En général, on cherche également à supprimer les termes rares afin de réduire la dimension de l'espace de représentation des documents, puisque, d'après la loi de Zipf, ces termes rares sont très nombreux.

Pour extraire les termes pertinents, Luhn a utilisé l'hypothèse que la valeur de pertinence d'un terme peut s'exprimer sous la forme d'une gaussienne en fonction du rang des termes d'un document comme montré sur la figure 3.2. La sélection des termes se base sur la fixation de deux seuils de fréquence (S_{max} et S_{min}) pour éliminer les termes de forte et de faible fréquence. Seuls les termes qui se situent entre ces deux seuils seront donc utilisés pour représenter les documents. Le choix de ces seuils reste empirique, ce qui constitue un frein à l'utilisation de cette méthode[17].

En pratique, après le filtrage des mots vides, seuls les termes très rares qui n'apparaissent que dans très peu de documents sont ignorés lors de la sélection des termes pertinents. Cette approche est connue sous le nom de "*Document Frequency*" (DF) (Yang et Pedersen, 1997).

[17]Il est à note, cependant, que (Salton et al., 1975) ont trouvé que l'intervalle $[\frac{N}{10}, \frac{N}{100}]$, où N est le nombre de documents, peut le plus souvent être adéquat pour sélectionner des termes présentant un pouvoir de discrimination satisfaisant.

Une alternative connue sous le nome de *"mean TF-IDF"* repose sur la sélection des termes qui retiennent les valeurs les plus hautes au regard de la moyenne des scores TF-IDF sur l'ensemble des documents du corpus (Tang et al., 2005). Cette démarche favorise les termes apparaissant dans le document et n'apparaissant pas dans le reste de la collection.

D'autres méthodes ont également été envisagées, dont la plupart ont comme propriété d'indexer les documents au moyen de termes "artificiels", absents des données d'origines. Ces méthodes sont dites "*méthodes d'extraction des termes*". Elles cherchent à représenter les documents dans un espace de dimension réduite, dans lequel chaque dimension correspond à une combinaison des termes originaux. La méthode LSI, que nous avons présenté dans la section 3.1.2, est un exemple typique de telles méthodes. D'autres extensions moins coûteuses que LSI ont aussi vu le jour, telle que la méthode dite "*indexation aléatoire*" (random indexing) (Sahlgren, 2005). Les méthodes de clustering de termes, telles que les réseaux neuronaux sémantiques, font aussi partie des méthodes d'extraction des termes dont le but est de grouper les termes en clusters de sémantiques communes. Les clusters ainsi créés représentent les nouveaux termes à utiliser lors de la représentation finale des documents.

Les méthodes d'extraction des termes, telles que LSI, semblent être meilleures que les méthodes de sélection, telles que DF (Tang et al., 2005). Néanmoins, un des défauts majeurs des méthodes d'extraction des termes vient du fait que les termes utilisés dans la représentation finale ne sont pas directement présents dans les documents, ce qui rend l'interprétation des dimensions de l'espace de représentation relativement difficile. Pour une revue plus complète des méthodes non supervisées, se référer notamment à (Fodor, 2002) et (Liu et al., 2003).

Sélection des termes d'indexation : cas supervisé

Les méthodes supervisées cherchent à retenir les termes en fonction de leur valeur de discrimination, un terme étant discriminant s'il est capable de bien distinguer entre documents appartenant à des classes différentes. La plupart de ces méthodes utilisent des critères — plus sophistiqués que la fréquence des termes — basés sur la théorie de l'information. Nous présentons ici trois des critères les plus connus, à savoir, l'information mutuelle (IM), le gain d'information (GI), et la statistique du χ^2 qui correspondent de manière générale à l'intuition que les termes les plus discriminants sont ceux distribués le plus différemment dans les ensembles d'exemples positifs et négatifs des classes.

Dans les formules mentionnées ci-après nous utilisons les notations suivantes : Soit $\{t_k\}_{k=1}^{n}$ l'ensemble des termes originaux, et soit $\{c_i\}_{i=1}^{m}$ l'ensemble des classes, nous désignons par A le nombre de documents contenant t_k et appartenant à c_i ; B le nombre de documents contenant t_k et n'appartenant pas à c_i ; C le nombre de documents appartenant à c_i et ne contenant pas t_k ; D le nombre de documents n'appartenant pas à c_i et ne contenant pas t_k. Ces quatre valeurs sont utilisées pour estimer respectivement les probabilités d'avoir ou de ne pas avoir un terme sachant une classe. Nous désignons aussi par N le nombre total de documents dans la collection.

L'information mutuelle mesure le lien entre un terme t_k et une classe c_i. Elle compare la probabilité de co-occurrence de t_k et c_i aux probabilités d'occurrence de t_k et c_i indépendamment les uns des autres. L'information mutuelle est définie par :

$$IM(t_k, c_i) \quad = \quad log\frac{p(t_k, c_i)}{p(t_k) \times p(c_i)} \approx log\ \frac{A \times N}{(A+B) \times (A+C)} \tag{3.8}$$

La faiblesse de ce critère est qu'il est beaucoup trop influencé par la fréquence des termes dans les documents. Plus précisément, pour les termes avec la même probabilité conditionnelle ($p(t_k, c_i)$), plus les termes sont rares, plus la valeur de l'information mutuelle s'accroît, du fait que le risque qu'ils apparaissent en dehors de la classe est moins important.

Le gain d'information, de son côté, mesure le nombre de bits d'informations obtenus, concernant le choix de la classe d'appartenance d'un terme t, en connaissant sa présence ou son absence dans le document. Il est donnée par la formule suivant :

$$GI(t_k, c_i) = \sum_{c \in \{c_i, \bar{c_i}\}} \sum_{t \in \{t_k, \bar{t_k}\}} p(t, c) \; log \; \frac{p(t, c)}{p(t) \times p(c)} \tag{3.9}$$

Le gain d'information prend donc en compte l'information sur l'absence du terme tandis que l'information mutuelle l'ignore.

Enfin, la statistique du χ^2 mesure le manque d'indépendance entre un terme t et une classe c. Elle peut être calculée à l'aide des formules suivantes :

$$\chi^2(t_k, c_i) = \frac{N \left(p(t_k, c) \times p(\bar{t_k}, \bar{c}) - p(t_k, \bar{c}) \times p(\bar{t_k}, c)\right)^2}{p(t_k) \times p(\bar{t_k}) \times p(c) \times p(\bar{c})} \tag{3.10}$$

$$\approx \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)} \tag{3.11}$$

La valeur de $\chi^2(t_k, c_i)$ est donc naturellement nulle si t_k et c_i sont indépendants, i.e. si t_k apparaît avec la même fréquence dans les ensembles des exemples positifs et négatifs de c_i, ce qui se traduit par $AD = CB$. À l'inverse, la valeur maximale de $\chi^2(t_k, c_i)$ peut être atteinte si t_k apparaît systématiquement dans l'ensemble des exemples positifs (négatifs) de c_i mais jamais dans l'ensemble des exemples négatifs (positifs) de c_i, on a donc $C = B = 0$ ($A = D = 0$) et $\chi^2(t_k, c_i)$ vaut N. Plus $\chi^2(t_k, c_i)$ est forte, plus t_k peut être considéré comme discriminant au regard de c_i.

À ce stade, il est à noter que la tâche de sélection des termes peut être menée localement ou globalement (Özgür et al., 2005)[18]. Dans le mode local, un ensemble de termes est choisi pour chacune des classes en fonction des valeurs $f(t_k, c_i)$, où f est un critère de sélection. En revanche, selon le mode global, un seul ensemble de termes est choisi pour toutes les classes en calculant, pour chaque terme, une valeur globale $f(t_k)$ à partir des valeurs $f(t_k, c_i)$, en utilisant l'une des formules suivantes :

$$f_{avg}(t_k) = \sum_{i=1}^{m} f(t_k, c_i) \tag{3.12}$$

$$f_{wavg}(t_k) = \sum_{i=1}^{m} p(c_i) f(t_k, c_i) \tag{3.13}$$

$$f_{max}(t_k) = \max_{i=1}^{m} \{f(t_k, c_i)\} \tag{3.14}$$

[18]Il faut préciser ici que la sélection des termes peut être effectuée localement ou globalement aussi bien dans les méthodes de sélection supervisées que non supervisées. Dans les deux cas, et pour un nombre suffisant de termes d'indexation, le mode global d'indexation est le meilleur. Néanmoins, dans le cas où le nombre de termes retenus est très faible, les classes rares tendent à être sous-représentées du fait que la plupart des termes seront sélectionnés des grandes classes (Özgür et al., 2005).

Des études comparatives de différents critères de sélection des termes, y compris les critères précités, ont été menées dans (Yang et Pedersen, 1997) et (Rogati et Yang, 2002). Selon les résultats de ces études, la statistique du χ^2 est la plus efficace pour sélectionner les termes discriminants en classification automatique de documents. Les critères *GI* et χ^2 ont souvent des performances comparables et supérieures à tous les autres critères. Pour sa part, l'information mutuelle ne donne pas de très bons résultats. Dans ces études, la performance de la méthode de sélection non supervisée (*DF*) sur la base de la fréquence a aussi été comparée à celle des méthodes supervisées. Malgré sa simplicité, la performance de cette méthode s'approche de celle de *GI* et χ^2. Elle pourrait donc être avantageusement utilisée pour des raisons de temps de calcul.

3.2 Grandes familles de filtrage d'informations

Le filtrage d'informations (FI) est un processus dont le but est d'extraire, à partir d'un flux dynamique d'informations, celles qui sont susceptibles d'intéresser un utilisateur ou un groupe d'utilisateurs en fonction de leurs besoins en information exprimés à travers des profils. La modélisation du profil de l'utilisateur est donc une tâche centrale dans ce processus qui conditionne largement son efficacité. Cette tâche nécessite, d'une part, de représenter les centres d'intérêts de l'utilisateur dans le système, et, d'autre part, d'adapter cette représentation aux changements des centres d'intérêts de l'utilisateur au cours du temps.

Une revue de la littérature dans ce domaine montre qu'il existe deux grandes familles de filtrage d'informations : le *filtrage basé sur le contenu* et le *filtrage collaboratif*. Ces modes de filtrage se distinguent principalement suivant que l'on considère le contenu des documents ou des indicateurs plus subjectifs reflétant l'intérêt qu'ils présentent pour un utilisateur particulier. La réflexion sur les avantages et les inconvénients de chacune de ces deux approches a conduit à leur combinaison, ce qui en fait du filtrage dit "*hybride*". Parallèlement à ces approches, Malon et al. (1987) a mis en évidence l'existence du filtrage dit "*économique*" qui a initialement été développé dans le cadre d'applications précises, comme le courrier électronique. Le filtrage économique fait partie intégrante du filtrage basé sur le contenu et du filtrage collaboratif, mais en considérant des indications supplémentaires concernant l'évaluation des coûts et des rendements, comme, par exemple, le prix et le coût de transmission des documents.

Dans la suite de cette section, nous présentons ces modes de filtrage, en insistant particulièrement sur le filtrage basé sur le contenu (cognitif) qui est au centre des problématiques traitées dans le cadre de cette thèse.

3.2.1 Filtrage basé sur le contenu

Le filtrage basé sur le contenu, ou encore le filtrage cognitif, tient compte seulement du contenu du profil utilisateur et celui des documents entrant en flux (Oard et Marchionini, 1996). Dans ce type de filtrage, trois modes différents d'acquisition du profil utilisateur peuvent être distingués :

- Le mode manuel dans lequel le profil est complètement acquis par intervention directe de l'utilisateur. L'utilisateur exprime son besoin par un ensemble de mots clés, ou termes spécifiques, généralement choisis dans une liste pré-établie de termes d'indexation. Les termes choisis sont ensuite employés pour guider le processus de filtrage ;
- Le mode automatique qui utilise un ensemble des documents, fournis par l'utilisateur en tant qu'exemples positifs et/ou négatifs de son besoin, pour créer, puis pour affiner, le

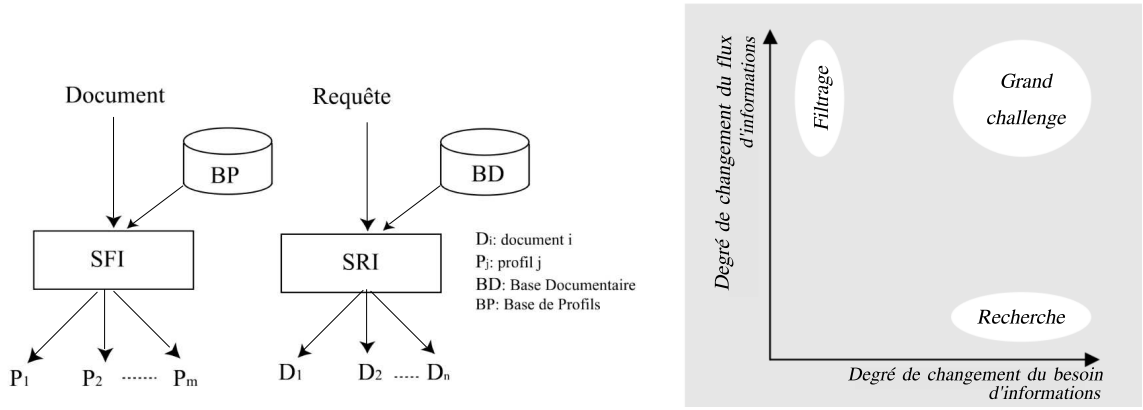
profil utilisateur ;
- Le mode mixte, dans lequel l'utilisateur conserve le contrôle sur le profil généré par le système. En d'autres termes, le profil est créé automatiquement par le système et l'utilisateur peut en prendre connaissance pour le modifier en ajoutant ou en supprimant des termes et/ou en changeant leurs poids.

Les expériences comparatives menées entre ces différents modes tendent à prouver que le mode automatique est le plus avantageux. Une des raisons est qu'il ne souffre pas du risque le plus important inhérent au mode manuel qui est que les utilisateurs ne parviennent pas à construire leur profil, ou soient lents à le mettre à jour. Dans le mode mixte, les utilisateurs ne semblent pas parvenir à améliorer de manière sensible la qualité d'un profil appris par le système (Annika, 2004).

Les sytèmes de filtrage d'informations ont vocation à traiter des données structurées aussi bien que non structurées. Ces données sont le plus souvent de type textuel, mais sont susceptibles de contenir des informations multimédia de type son, image et vidéo. Vu la nature des données documentaires, il faut procéder à un prétraitement pour transformer les documents entrant en flux en une représentation convenable pour les algorithmes destinés à la création du profil utilisateur et à la sélection des documents pertinents vis-à-vis du profil (cf. Section 3.1).

Le fonctionnement général d'un système de filtrage d'informations tel qu'il a été présenté dans (Belkin et Croft, 1992), peut se résumer comme suit : Les utilisateurs expriment leurs besoins en informations qui peuvent évoluer lentement au cours du temps, au fur et à mesure que les connaissances et/ou les centres d'intérêt des utilisateurs changent. Le système de filtrage d'informations garde une représentation des besoins d'informations des utilisateurs à travers des profils. En parallèle, les documents qui arrivent en flux doivent être traités de façon à leur associer une représentation interne. La représentation des documents est comparée aux profils afin de déterminer la similarité entre les documents et les profils et donc d'identifier les documents pertinents. Les documents retenus sont évalués par les utilisateurs en termes de réponse à leurs besoins. Cette évaluation peut mener, dans la plupart des cas, à la modification des centres d'intérêt et, par conséquent, des profils.

Historiquement, le filtrage basé sur le contenu trouve ses racines dans le domaine de recherche d'informations (RI). La recherche d'informations est une tâche très ancienne, dont le but est d'isoler, dans un ensemble de documents, un sous-ensemble de documents pertinents vis à vis du besoin d'un utilisateur formulé par une requête élémentaire (van Rijsbergen, 1979). La dualité entre le filtrage et la recherche d'informations a été rendue explicite par Belkin et Croft (1992). Cette dualité s'est traduite par les faits suivants :
- Un système de recherche d'informations gère une collection statique de documents, alors qu'un système de filtrage d'informations s'applique à un flux dynamique de documents ; La collecte et l'organisation des documents est donc une des fonctionnalités des systèmes de recherche d'informations, alors qu'il s'agit de la diffusion ou la distribution des documents à des utilisateurs ou des groupes d'utilisateurs dans le cas des systèmes de filtrage d'informations ;
- Les requêtes dans les systèmes de recherche d'informations reflètent des intérêts à court terme, alors que les profils dans les systèmes de filtrage d'informations représentent des intérêts à long terme ;
- Un système de recherche d'informations compare une requête avec un ensemble de documents et en sélectionne ceux qui sont pertinents vis-à-vis de la requête, alors qu'un système



(a) (b)

FIG. 3.3 – (a) Une comparaison entre les processus de recherche (SRI) et de filtrage d'Information (SFI). (b) Une distinction entre la recherche et le filtrage d'informations suivant le degré de changement des sources d'informations (les flux) et des besoins en informations (profils). Le plus grand challenge auquel sont confrontés les systèmes d'accès personnalisé aux informations est de s'adapter à des environnements caractérisés à la fois par une forte dynamicité des sources d'informations et par une dérive importante du besoin d'informations de l'utilisateur au cours du temps. Adapté de (Oard et Marchionini, 1996).

de filtrage d'informations compare un document avec un ensemble de profils et achemine le document aux profils adéquats (cf. Figure 3.3 (a)) ;

- Un système de recherche d'informations propose une liste de documents classée par ordre de pertinence ; alors qu'un système de filtrage d'information prend une décision binaire sur la pertinence ou non des documents.

D'une manière plus générale, la distinction entre les systèmes de recherche et de filtrage d'informations peut être établie suivant le degré de changement inhérent aux flux d'informations et aux besoins en informations (profils), cf. Figure 3.3 (b). En effet, plus le changement des flux d'informations croît (resp. décroît) et plus le changement des besoins en informations décroît (resp. croît), plus la tendance sera vers l'application d'un système de filtrage d'informations (resp. un système de recherche d'information). La distinction entre le filtrage et la recherche d'informations se complique lorsque le degré de changement des flux et des besoins en informations évolue constamment, ce que l'on appelle le "grand challenge" dans le processus d'accès à l'information (Oard et Marchionini, 1996). Dans le cadre de nos travaux, nous nous sommes intéressés plus particulièrement au problème du filtrage d'informations dans un environnement dynamique caractérisé à la fois par la dynamicité du flux d'informations et par l'évolution du besoin de l'utilisateur au cours du temps (cf. Chapitre 5). D'abord, nous mettrons en place une méthode d'apprentissage permettant la modélisation du profil de l'utilisateur ayant un besoin en informations relativement stable (cf. Sections 5.1 et 5.2). Puis, nous évaluerons cette méthode sous l'angle d'une dérive, progressive ou brusque, du besoin de l'utilisateur (cf. Section 5.3).

3.2.2 Filtrage collaboratif

Le filtrage collaboratif, ou social, est un mode de filtrage basé sur la notion de consensus. Il envoie automatiquement des recommandations aux utilisateurs en fonction des jugements effectués par des "communautés" auxquelles ils appartiennent, ces communautés étant des groupes d'utilisateurs qui partagent les mêmes centres d'intérêt.

Dans le filtrage collaboratif, les utilisateurs fournissent des annotations ou votes indiquant leur satisfaction vis-à-vis des documents qu'ils reçoivent, pour constituer leur profil. Pour pouvoir agréer efficacement les annotations des utilisateurs, ceux-ci sont souvent invités à donner des notations simples, qui peuvent être ramenées sur une échelle graduelle de valeurs numériques, par exemple de 1 ($\equiv$ très mauvais) à 5 ($\equiv$ très bien) (Miller et al., 1997). De telles annotations peuvent aussi être acquises de manière implicite en observant le comportement de l'utilisateur comme, par exemple, le temps passé sur un document, et la sauvegarde/suppression d'un document (Nichols, 1997). Les valeurs obtenues sont typiquement stockées sous la forme d'une matrice utilisateurs-votes.

Les votes des utilisateurs sont comparés et des similitudes sont mesurées pour créer des communautés d'utilisateurs aux goûts similaires. Des prévisions de scores des documents vis à vis de chaque utilisateur sont ensuite calculées en opérant la moyenne pondérée des avis d'autres utilisateurs avec des opinions soit similaires, soit complètement opposés. Au final, un utilisateur se verra distribuer les documents dont les scores le concernant sont les plus hauts. Dans ce type de filtrage, il n'y a donc pas d'analyse du sujet ou du contenu et un document n'est connu que par son identifiant. Le système de filtrage collaboratif comprend ainsi les fonctionnalités suivantes :

1. Le calcul de la similitude entre les utilisateurs ;
2. Le calcul de la pertinence d'un document vis-à-vis d'un utilisateur particulier à partir de la matrice des votes ;
3. La mise à jour continue des profils utilisateurs au fur et à mesure de la collecte de leurs annotations des documents.

Plusieurs algorithmes ont été développés pour le filtrage collaboratif. Selon Breese et al. (1998), ces algorithmes peuvent être principalement regroupés en deux grandes familles : Les algorithmes fondés sur la mémoire et les algorithmes fondés sur les modèles. Les algorithmes fondés sur la mémoire opèrent directement sur la matrice utilisateurs-votes pour effectuer la prédiction. Pour l'utilisateur en cours, la prédiction est faite à partir des votes de l'utilisateur, et un ensemble de poids calculés à partir des votes d'autres utilisateurs. L'influence d'un utilisateur y est d'autant plus forte que son degré de similarité avec l'utilisateur en cours est fort. Ce type d'algorithmes présente l'avantage d'être simple et très réactif, en intégrant immédiatement au système les modifications des profils utilisateurs. Néanmoins, la complexité des algorithmes en temps et en mémoire est beaucoup trop importante pour un nombre important d'utilisateurs et de documents. Pour leur part, les algorithmes fondés sur les modèles consistent à créer un modèle descriptif des votes des utilisateurs via un processus d'apprentissage qui est alors utilisé pour effectuer la prédiction. Les modèles plus répandus sont les modèles à base de clusters (Ungar et Foster, 1998), et les modèles à base de réseaux bayésiens (Miyahara et Pazsani, 2000). Une vue d'ensemble des algorithmes de filtrage collaboratif est notamment disponible dans (Breese et al., 1998) et (Lemire, 2003).

3.2.3 Filtrage hybride

Bien que le filtrage basé sur le contenu offre de multiples avantages tels que la rapidité et la précision des prévisions vis-à-vis des besoins des utilisateurs, il présente certaines limites dues à son principe, qui est de ne prendre en compte que le contenu des documents, ce qui ne permet pas d'intégrer d'autres facteurs de pertinence comme par exemple la qualité des faits présentés, ou la fiabilité de la source d'informations, etc. À son tour, le filtrage collaboratif présente aussi un certain nombre de limites. En effet, il soufre du problème de démarrage à froid. Les nouveaux utilisateurs commencent avec un profil vide ; une période d'apprentissage est donc nécessaire avant que le profil ne reflète concrètement les centres d'intérêts de l'utilisateur. De même, ce type de filtrage ne prend en considération que des documents qui ont déjà été consultés, les nouveaux documents ne peuvent être diffusés que si un minimum d'informations les concernant est collecté à partir des avis des utilisateurs. D'un autre côté, il existe toujours un risque de "sclérose" pour une communauté d'utilisateurs ayant des goûts peu fréquents.

Le filtrage hybride combine le filtrage basé sur le contenu et le filtrage collaboratif dans le but de tirer profit des avantages de chacune de ces deux approches en limitant les faiblesses qu'elles présentent. Il existe différents types de méthodes d'hybridation (Burke, 2002) ; toutes ces méthodes se basent de manière générale sur deux approches principales :

1. Le filtrage hybride peut s'effectuer en deux phases. La première phase consiste à appliquer séparément un filtrage basé sur le contenu et un filtrage collaboratif pour générer des recommandations candidates. La seconde phase consiste à combiner les deux ensembles de recommandations préliminaires selon certaines méthodes afin de produire les recommandations finales pour les utilisateurs. À titre d'exemples, Claypool et al. (1999) attribuent un poids égal aux prévisions des deux types de filtrage, mais ajuste graduellement la pondération suivant la confirmation/infirmation des prévisions par les utilisateurs. Billsus et Pazzani (2000) utilisent une stratégie de commutation entre les deux modes de filtrage en fonction de leur performance.
2. Le filtrage hybride peut gérer des profils utilisateurs orientés contenu, et effectue une mise en correspondance entre ces profils, dont le but est soit de former directement de communautés d'utilisateurs (Balabanovic, 1996), soit de remplir la matrice utilisateurs-votes (Melville et al., 2002), ce qui permettra dans un deuxième temps un filtrage collaboratif.

En plus d'être relativement simple à mettre en œuvre, l'intérêt que présente la première approche est que les fonctionnalités des modes de filtrage restent intactes tout en sachant s'affranchir de leurs limites. L'efficacité de telles approches dépendrait bien entendu des poids accordés à chaque mode de filtrage et de leur ajustement.

En conclusion de cette section, il apparaît à l'évidence que chaque mode de filtrage a ses propres avantages et faiblesses, et qu'ils peuvent être autant complémentaires que divergents. Comme nous l'avons précédemment énoncé, nous nous intéressons plus particulièrement dans le cadre de cette thèse au filtrage basé sur le contenu, mais nous présenterons aussi plus loin dans ce document, une stratégie de combinaison s'inscrivant dans l'orientation de la première approche d'hybridation décrite ci-dessus. La stratégie que nous avons développée, nous a permis de combiner les avantages du filtrage dirigé par le contenu avec ceux du filtrage collaboratif dans le cadre d'un système de distribution ciblée de sites Web par satellite (cf. Chapitre 5, Section 5.4). Pour des raisons de simplification, nous employons dans la suite du document le terme "filtrage" pour désigner, sauf indication contraire, le mode de filtrage basé sur le contenu.

3.3 Spécificités fonctionnelles du filtrage basé sur le contenu

3.3.1 Tâches de filtrage actuelles

Une des toutes premières formes de filtrage d'informations fut la dissémination sélective d'informations (SDI — Selective dissemination of information) à la fin des années 50. La notion de SDI a été introduite dans (Luhn, 1958) pour faire référence à un processus de filtrage d'informations, dont le but est de tenir constamment informés les utilisateurs d'une bibliothèque scientifique, des nouvelles références bibliographiques relatives à leurs domaines de spécialisation. Les systèmes utilisant la SDI suivaient généralement les étapes décrites par Luhn, à l'exception de la modélisation automatique des profils utilisateurs, qui sont plutôt construits manuellement ; des exemples de tels systèmes sont présentés dans (Housman, 1969). Le terme classique de "filtrage d'informations" a initialement été utilisé dans (Denning, 1982) pour désigner un processus visant à filtrer les informations arrivées par courrier électronique afin de séparer les messages urgents de ceux routiniers, et de limiter l'affichage des messages routiniers. Plus tard, la notion de filtrage a été étendue à d'autres domaines que le courrier électronique comme les articles de presse et les articles diffusés sur Internet (Foltz, 1990).

Depuis 1992, le programme international TREC (Text REtrieval Conference)[19], qui s'intéresse à l'évaluation des systèmes de recherche et de filtrage d'information, a joué un rôle fondamental dans le développement du domaine de filtrage d'informations (Harman, 1992a,b). Dans le cadre du programme TREC, trois sous-tâches de filtrage ont été principalement envisagées et mises en place : le routage (routing), le filtrage par lots (batch filtering), et le filtrage adaptatif (adaptive filtering) (Hull, 1998). La tâche de filtrage a cessé d'apparaître explicitement dans le programme TREC depuis la fin de l'année 2002 (Voorhees, 2002).

- Le routage est très semblable à la recherche d'informations. Le système dispose d'une collection de documents étiquetés et rangés dans des catégories/classes représentatives de différents besoins en informations. La collection est aussi divisée en deux parties distinctes, une pour l'apprentissage, l'autre pour le test. Le système doit apprendre à partir des documents d'apprentissage appartenant à une des catégories — que constituent des exemples positifs d'un besoin en informations — et probablement, à partir des documents d'apprentissage appartenant aux autres catégories — que constituent des exemples négatifs du besoin en informations — un profil utilisateur selon lequel les documents de test doivent ensuite être classés par ordre décroissant de pertinence. Le système ne doit donc pas prendre une décision binaire concernant l'appartenance ou non des documents de test à la catégorie en question, mais simplement être capable de leur associer un score de pertinence. Les systèmes de routage sont donc évalués en fonction des critères d'évaluation de l'ordonnancement qu'ils proposent (cf. Section 1.5.1). TREC ne considère souvent que les 1000 documents les plus pertinents dans l'évaluation des systèmes de routage.
- Le filtrage par lots est quasiment similaire au routage, à l'exception que les systèmes de filtrage doivent faire une décision binaire pour classer chaque document de test comme pertinent ou non pertinent vis-à-vis des différentes catégories. Le filtrage par lots peut donc être perçu comme un cas spécial de classification supervisée, à deux classes, pertinente et non pertinente. Dans certains cas, le système peut tirer parti de l'étiquetage des documents de test pour s'améliorer au fil du temps, et le filtrage est dit, dans de tels cas, un filtrage adaptatif par lots. Cette opération est en fait semblable au mécanisme de bouclage de

[19] Toutes les publications et les informations relatives aux conférences TREC sont disponibles en ligne à l'adresse suivante : http ://trec.nist.gov

pertinence (relevance feedback) exploité dans les systèmes de recherche d'informations lors de la reformulation automatique de requêtes (Rocchio, 1971).

- Le filtrage adaptatif consiste à construire, dans un premier temps, un profil initial à partir de très peu de documents d'apprentissage (trois documents par catégorie à la tâche de filtrage adaptatif de TREC 2002 (Robertson et Callan, 2002)). Pour chaque document de test, le système doit prendre une décision binaire quant à l'acceptation ou le rejet du document, et peut utiliser, comme précédemment, l'étiquetage des documents de test pour s'améliorer au fil du temps. Une des difficultés principales de ce type de filtrage vient donc du fait que le nombre d'exemples d'apprentissage au démarrage du processus de filtrage est très insuffisant et le système peut uniquement apprendre à partir des jugements de pertinence sur les documents qu'il a déjà filtrés.

Les critères d'évaluation supervisée cités dans la section 1.5.1 du chapitre 1 peuvent être utilisés dans l'évaluation des systèmes de filtrage adaptatif et par lots. Il peut être intéressant de noter que d'autres critères d'évaluation, que nous ne détaillerons pas ici, ont aussi été développés et utilisés dans le cadre du programme TREC, cf. (Robertson et Hull, 2001) et (Robertson et Soboroff, 2002).

Pour récapituler, la problématique de filtrage d'informations se résume ainsi à deux questions fondamentales : L'apprentissage du profil utilisateur et la détermination du seuil de décision propre à ce profil. Pour répondre à ces questions, la vaste majorité des travaux de recherche relatifs au filtrage d'informations sont essentiellement basés sur des modèles de recherche d'informations auxquels se sont ajoutées des fonctionnalités d'adaptation des profils et des fonctions de décision. Le nombre très important de ces travaux rend impossible une présentation exhaustive des méthodes existantes. C'est pourquoi nous nous limitons ici, dans un premier temps, à la présentation de quelques-uns des travaux les plus marquants qui mettent en œuvre des méthodes d'apprentissage numérique de profils utilisateurs. Ensuite, nous décrirons les méthodes de seuillage et les principales fonctions de décision dans les cadres du filtrage par lots et du filtrage adaptatif.

3.3.2 Méthodes d'apprentissage du profil utilisateur

Comme nous l'avons précédemment souligné, la majorité des méthodes d'apprentissage du profil utilisateur proposées dans la littérature sont inspirées des techniques de reformulation de requêtes en recherche d'informations. On y trouve principalement l'algorithme de Rocchio et ses variantes (Ault et Yang, 2001; Schapire et al., 1998; Schütze et al., 1995), des techniques d'apprentissage basées sur les classifieurs bayésiens (Chai et al., 2002), les classifieurs des k-plus-proches-voisins (Ault et Yang, 2000, 2001), la régression logistique (Schütze et al., 1995; Zhang, 2004), les réseaux de neurones (Habibi et Eberts, 1992; Schütze et al., 1995; Jennings et Higuchi, 1993).

Les méthodes d'apprentissage du profil utilisateur présentées dans cette section se situent dans le cadre du modèle vectoriel (cf. Section 3.1.2). Les profils et les documents sont donc représentés par des vecteurs de termes pondérés selon la formule standard TF-IDF ou l'une de ses variantes. La mesure cosinus est le plus souvent utilisée pour calculer la similarité entre les vecteurs représentatifs du profil et du document.

Algorithme de Rocchio

Initialement conçu comme une méthode de reformulation de requêtes dans le contexte du modèle vectoriel en recherche d'informations (Rocchio, 1971), l'algorithme de Rocchio a également connu des succès dans les domaines de classification et de filtrage des données textuelles (Schapire et al., 1998; Ault et Yang, 2001). En filtrage d'informations, un profil utilisateur est calculé selon :

$$\overrightarrow{P}_c = \frac{\beta}{|\mathcal{R}|} \sum_{d_i \in \mathcal{R}} \overrightarrow{d_i} - \frac{\gamma}{|\mathcal{N}|} \sum_{d_i \in \mathcal{N}} \overrightarrow{d_i} \tag{3.15}$$

où $\overrightarrow{d_i}$ correspond à la représentation vectorielle du document d_i ; $\mathcal{R}$ représente l'ensemble des documents représentant des exemples positifs du besoin en informations de l'utilisateur ; $\mathcal{N}$ représente l'ensemble des documents représentant des exemples négatifs du besoin de l'utilisateur ; les paramètres β et γ sont des paramètres positifs contrôlant respectivement l'importance des termes dans les exemples positifs et dans les exemples négatifs. Le classement des documents au regard du besoin de l'utilisateur s'opère en calculant la similarité cosinus entre le profil et les documents.

La formule de Rocchio (Eq. 3.15) requiért que les vecteurs des documents soient normalisés et que les composantes négatives de P_c soient remises à zéro. Le choix des valeurs adéquates des paramètres β et γ est un facteur important, mais pourtant négligé dans la littérature. Le plus souvent, on a $\gamma<\beta$. En effet, on considère généralement que les exemples positifs sont plus utiles et plus précis que les exemples négatifs lors de la formulation du profil. Par exemple, la formule de Rocchio a été utilisée avec les valeurs β=16 et γ=4 dans (Buckley et al., 1994), (Debole et Sebastiani, , 2005) et (Cohen et Singer, 1999). Moschitti (2003) a aussi montré qu'il est possible d'estimer les paramètres par validation croisée et qu'une telle estimation aboutirait logiquement à une meilleure performance que lorsque l'on utilise les paramétrisations les plus répandues dans la littérature (que nous venons de évoquer ci-avant). L'estimation des paramètres par validation croisée s'avère, néanmoins, très coûteuse en temps de calcul, ce qui peut avoir un impact non négligeable dans le contexte du filtrage d'informations. Il est également possible de mettre la valeur de γ à zéro (Ce qui correspond au cas où le profil sera uniquement représenté par le centroïde des exemples positifs), comme c'est le cas, par exemple, dans (Dumais et al., 1998), et (Schütze et al., 1995).

Perfectionnements de l'algorithme de Rocchio

L'algorithme de Rocchio est très simple à mettre en œuvre, de complexité linéaire, et s'est révélé plus performant que d'autres algorithmes, comme KNN et SVM, dans le cas d'un très faible nombre d'exemples d'apprentissage (Vinot, 2003). C'est grâce à ces propriétés que l'algorithme de Rocchio a été largement adapté au cas du filtrage adaptatif (Ault et Yang, 2001). Un des défauts de l'algorithme de Rocchio est la difficulté qu'il rencontre si les exemples positifs ont tendance à former des classes disjointes. Dans un tel cas, la majorité des données ne seraient pas classifiées correctement du fait que le profil qui est fondamentalement calculé comme une moyenne de ces données peut en être loin (cf. Figure 3.4). Une solution à ce problème a été proposée dans (Vinot, 2003) dont l'idée principale est de construire, de manière non supervisée, des sous-classes des exemples positifs permettant d'améliorer de manière significative les performances de classification, en l'occurrence de filtrage, tout en augmentant de manière minimale l'ensemble des profils à construire. En termes de performances, cette variante de l'algorithme de Rocchio — maintenant plusieurs profils par utilisateur — permet d'obtenir des performances significativement meilleures que celles obtenues avec la version originale de l'algorithme de Rocchio.

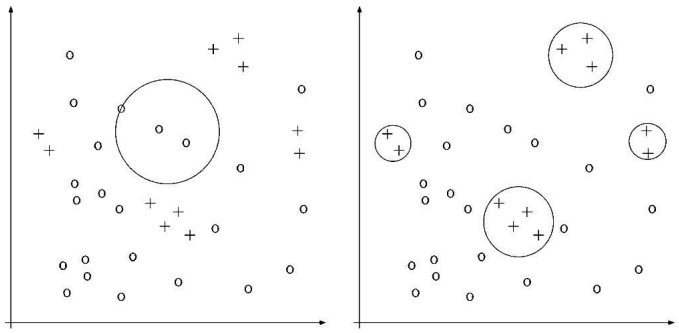


FIG. 3.4 – Une comparaison entre le comportement de (a) Rocchio et (b) KNN (méthode des k plus proches voisins). Les croix et les cercles dénotent respectivement les exemples positifs et négatifs d'apprentissage. Les grands cercles dénotent les profils descriptifs des exemples positifs. Notez que, pour des raisons de facilité d'illustration, la similarité entre données est ici vue en termes de distance euclidienne plutôt que, comme plus commun, en termes de similarité cosinus. D'après (Sebastiani, 2002).

Également, d'autres améliorations ont été apportées à l'algorithme de Rocchio. Il s'agit principalement de nouvelles méthodes de représentation et de pondération des vecteurs des documents (Buckley et Salton, 1995), (Singhal et al., 1996) ; et d'un choix plus sélectif des exemples négatifs intervenant dans la formule de Rocchio (Singhal et al., 1997; Schapire et al., 1998).

La méthode de "*Query Zoning*" (Singhal et al., 1997) consiste à améliorer la qualité du profil utilisateur en focalisant seulement sur les exemples négatifs "*positifs-proches*", qui sont les plus difficiles à distinguer des exemples positifs. Ces exemples sont identifiés comme étant les k exemples négatifs les plus proches du vecteur controïde des exemples positifs. Dans (Schapire et al., 1998), le nombre k est limité à MAX($|\mathcal{D}|/100$, $|\mathcal{R}|$), où $|\mathcal{D}|$ représente le nombre total de documents d'apprentissage et $|\mathcal{R}|$ représente le nombre de documents représentant les exemples positifs du besoin de l'utilisateur. L'idée d'identification des exemples négatifs positifs-proches a également été explorée avec succès dans le cadre de la classification hiérarchique de documents textuels, cf. (Wiener, 1995) et (Ng et al., 1997). En employant la méthode de "*Query Zoning*" avec d'autres perfectionnements[20], (Schapire et al., 1998) ont constaté que l'algorithme de Rocchio pouvait se hisser au niveau des meilleures méthodes, telles que le "*boosting*", tout en allant plus vite.

Les diverses améliorations qui y ont été apportées, ont provoqué un regain d'intérêt pour l'algorithme de Rocchio, avec des travaux récents en filtrage d'informations ; en voici quelques exemples (Hoashi et al., 1999), (Dumais et al., 1998) et (Wang et al., 2001). Des versions incrémentales de l'algorithme de Rocchio ont également été proposées pour l'apprentissage de profils utilisateurs dans le cadre du filtrage adaptatif. On y trouve, entre autres, les travaux de Al-

[20]Il s'agit d'une méthode d'optimisation dynamique de poids des termes qui modifie, au cours d'un processus de bouclage de pertinence, le poids d'un terme de la requête proportionnellement à la somme positive des poids de ce terme dans les documents jugés pertinents et la somme négative des poids du terme dans les documents jugés non pertinents (Buckley et Salton, 1995).

lan (1996), Arampatzis et al. (2001), et Tebri et al. (2005). L'algorithme de Rocchio en version incrémentale obéit au même principe opérationnel du bouclage de pertinence en recherche d'informations qui est celui de rapprocher le profil utilisateur d'un vecteur caractérisant les documents filtrés et jugés pertinents par l'utilisateur tout en l'éloignant d'un vecteur caractérisant les documents filtrés par le système mais rejetés par l'utilisateur. Selon ce principe, le profil reformulé $\overrightarrow{P}^{(t)}_{new}$ s'exprime sous le forme suivante :

$$\overrightarrow{P}^{(t)}_{new} = \alpha\ \overrightarrow{P}^{(t)}_{orig} + \frac{\beta}{|\mathcal{R}^{(t)}|} \sum_{d_i \in \mathcal{R}^{(t)}} \overrightarrow{d_i} - \frac{\gamma}{|\mathcal{N}^{(t)}|} \sum_{d_i \in \mathcal{N}^{(t)}} \overrightarrow{d_i} \tag{3.16}$$

où $\overrightarrow{P}^{(t)}_{orig}$ est le profil original ; $\mathcal{R}^{(t)}$ et $\mathcal{N}^{(t)}$ représentent respectivement l'ensemble des documents pertinents et non pertinents filtrés à l'étape t.

Modèles connexionnistes

Plusieurs modèles basés sur le principe des réseaux neuronaux ont été conçus pour la recherche d'informations (Wilkinson, 1991; Boughanem et Soulé-Dupuy, 1992), dont certains ont également été repris et exploités dans le cadre du filtrage d'informations (Boughanem et al., 2001, 2002). Comme nous avons vu dans la section 3.1.2, ces modèles peuvent contribuer à combler les lacunes du modèle vectoriel, en représentant les relations qui existent entre les termes (ex. synonymie, voisinage, etc.), entre les documents (ex. similitude, référence, etc), et enfin entre les termes et les documents (ex. fréquence, poids, etc). Il n'existe pas de représentation unique d'un réseau de neurones pour la recherche ou le filtrage d'informations, mais, d'une manière générale, un réseau de neurones est souvent construit à partir des représentations initiales du profil utilisateur et des documents. L'interaction entre les neurones descriptifs du profil et ceux des documents est au cœur du processus de filtrage d'informations : toute propagation d'activité des neurones descriptifs du profil vers ceux des documents peut être assimilée à une opération de mise en correspondance entre le profil et les documents, alors que toute propagation dans le sens inverse peut être assimilée à une opération de bouclage de pertinence adaptative.

Dans les paragraphes qui suivent, nous décrirons deux modèles connexionnistes, à savoir, le modèle MERCURE et le modèle RELIEFS, d'autres modèles plus simples peuvent être trouvés dans (Callan, 1998), (Lam et Yu, 2003), (Schütze et al., 1995) et (Lewis et al., 1996).

Le modèle MERCURE

Le modèle MERCURE (ModèlE de Réseau Connexionniste poUr la REecherche d'information) a initialement été conçu pour la recherche d'informations (Boughanem et Soulé-Dupuy, 1992), et puis, adapté dans le cadre du filtrage d'informations (Boughanem et al., 1998, 2001, 2002). Il s'agit d'un réseau associatif à trois couches : une couche profil (P), une couche terme (T) et une couche document (D) ; chaque couche est constituée d'un ensemble de neurones reliés par des connexions pondérées représentant des liens de similarité entre termes/documents (cf. Figure 3.5). Les poids des connexions entre les neurones documents (D) et les neurones termes (T) sont fixés à l'avance. Dans le cas du filtrage d'informations par lots, ces poids sont directement dérivés des fréquences des termes dans les documents d'apprentissage comme suit :

$$w_{ij} = \frac{(1 + log(tf_{ij})(h_1 + h_2\ log(\frac{N}{n_i}))}{h_3 + h_4\ \dfrac{dl}{avg\text{-}dl}} \tag{3.17}$$

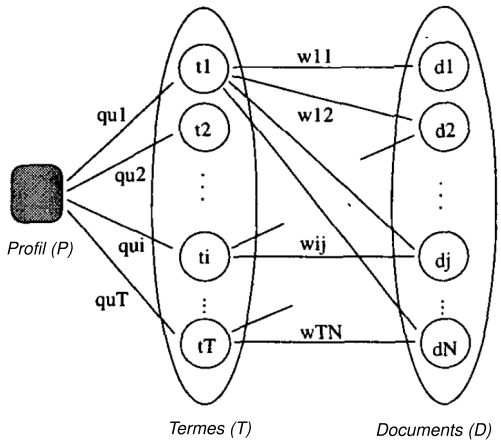


FIG. 3.5 – L'architecture générale du modèle MERCURE. Adapté de (Tmar et Boughanem, 2000).

où h_1, h_2, h_3 et h_4 sont des paramètres contants ; tf_{ij} représente la fréquence du terme t_i dans le document d_j ; dl est le nombre de termes dans le document d_j ; $avg\text{-}dl$ est la longueur moyenne des documents.

Les poids des connexions entre les neurones descriptifs des termes du profil (P) et les neurones équivalents dans la couche terme (T) sont aussi fixés selon la formule suivante :

$$q_{ik} = \begin{cases} \dfrac{(1+log(tf_{ik}))(log\frac{N}{n_i})}{\sqrt{\sum_{j=1}^{T}(1+log(tf_{ik}))(log(\frac{N}{n_j})^2}} & si \quad tf_{ik} \neq 0 \\ 0 \quad sinon \end{cases}$$

où tf_{ik} représente la fréquence du terme t_i dans le profil ; N est le nombre total des documents dans la collection d'apprentissage ; n_i est le nombre de documents pami N contenant le terme t_i.

La mise en correspondance entre le profil utilisateur et les documents est opérée en préactivant les neurones descriptifs des termes présents dans le profil avec une valeur d'activité unité et en propageant cette activité vers la couche D. La pertinence des documents est estimée selon les valeurs d'activité des neurones correspondants. MERCURE assure aussi la reformulation du profil via un processus de rétro-propagation du gradient de la couche D vers la couche P (Tmar et Boughanem, 2000).

MERCURE a aussi été expérimenté dans le cadre du filtrage d'information adaptatif dans les travaux de Boughanem et al. (2001, 2002). Des nouvelles méthodes de pondération des connexions document-terme et terme-profil — qui ne nécessitent aucune connaissance autre que le profil initial de l'utilisateur au démarrage du processus de filtrage — ont donc été proposées. Des améliorations permettant notamment de mieux adapter le profil utilisateur et la fonction de seuil au cours du processus de filtrage ont aussi été introduites dans (Boughanem et Tmar, 2002; Tebri et al., 2005).

Le modèle RELIEFS

Le modèle RELIEFS (RElevance Information Extraction Fuzzy System) (Brouard et Nie, 2004) est un système de filtrage adaptatif de documents. Il s'appuie sur la notion de résonance sous la forme où elle apparaît dans les réseaux type ART (cf. Section 2.2.4) ; il n'en est pas cependant une implémentation. L'architecture de RELIEFS oppose une couche de neurones représentant les termes des documents à un neurone de pertinence qui symbolise le profil utilisateur et qui représente la pertinence d'un document vis-à-vis de ce profil. Les poids des connexions bidirectionnelles, liant les neurones termes et le neurone de pertinence, sont mis à jour de façon continue (cf. Figure 3.6).

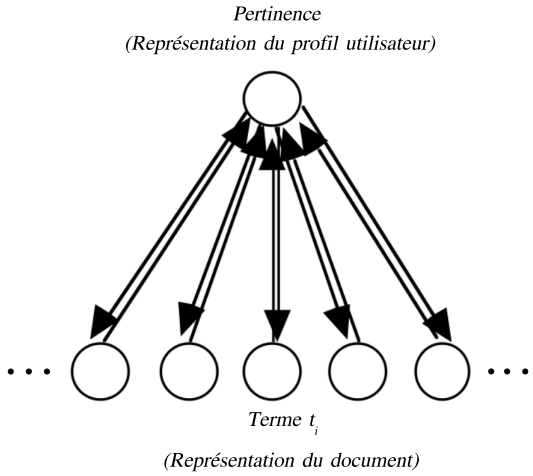


FIG. 3.6 – L'architecture générale du modèle RELIEFS. Adapté de (Brouard et Nie, 2004).

Le fonctionnement global du modèle se résume comme suit : Lorsqu'un document se présente, les termes présents dans ce document agissent comme des indices de la pertinence du document. Un bon indice est un terme dont la résonance avec le neurone de pertinence est importante (les deux implications $t_i \rightarrow P$ et $P \rightarrow t_i$ sont fortes). La pertinence globale du document est calculée sur la base de la résonance de l'ensemble des termes du document vis-à-vis du neurone de pertinence. La qualité de résonance d'un terme correspond à sa capacité à propager l'activation vers le neurone de pertinence et à recevoir une activation de ce même neurone. Soit W_{iP} le poids de la connexion orientée du terme t_i vers le neurone de pertinence P et W_{Pi} le poids de la connexion inverse. La résonance entre le terme t_i et le neurone de pertinence P est évaluée par le produit $W_{iP}.W_{Pi}$. L'absence d'un terme est donc d'autant plus pénalisant que ce terme est résonant avec le profil, c'est-à-dire que $W_{iP}.W_{Pi}$ est fort. Donc, la pertinence d'un document s'écrit :

$$R(d,P) = \frac{\sum_i W_{iP}.W_{Pi}}{\sum_j W_{jP}.W_{Pj}} \tag{3.18}$$

où i représente les indices des termes présents dans le document et j représente les indices de tous les termes reliés au neurone de pertinence.

Pour s'adapter au besoin de l'utilisateur, les poids des connexions sont modifiés en fonction des jugements fournis par l'utilisateur sur la pertinence des documents filtrés par le système. Soient $W_{AB_{k-1}}$ et W_{AB_k} les poids de la connexion orientée du neurone A vers le neurone B,

avant et après la prise en compte de la k-ème observation (que constitue une paire document-jugement de pertinence), où A et B correspondent respectivement à un terme et au neurone de pertinence ou inversement ; et soit $\mu_A(D_k)$ une valeur indiquant la présence ou l'absence d'un terme représenté par un neurone A dans un document D_k (1 si présent et 0 si absent). La règle d'apprentissage proposée est la suivante :

$$W_{PQ_k} = \frac{\alpha W_{AB_{(k-1)}} + \mu_A(D_k)\mu_B(D_k)}{\alpha + \mu_A(D_k)} \quad \text{avec} \quad \alpha = \sum_{i=1}^{k-1} \mu_A(D_k) \tag{3.19}$$

Cette règle correspond à une forme générale de la règle de Hebb dont le principe est le suivant :

- Si A et B sont présents, la connexion de A vers B et la connexion inverse sont renforcées. De cette façon, à chaque fois un document est jugé pertinent par l'utilisateur, le poids des termes qui y apparaissent augmente.
- Si A est présent et B ne l'est pas, la connexion de A vers B est affaiblie et la connexion inverse reste inchangée. D'un côté, si A est un terme et B le neurone de pertinence, ceci revient à affaiblir le poids attribué à un terme qui apparaît dans un document jugé comme non pertinent par l'utilisateur. De l'autre côte, si A est le neurone de pertinence et B est un terme, ceci revient à affaiblir le poids attribué à un terme qui n'apparaît pas dans un document jugé comme pertinent par l'utilisateur.

3.3.3 Méthodes de seuillage dans les systèmes de filtrage d'informations

Le seuillage permet, après le calcul des scores de pertinence des documents, de prendre une décision binaire quant à l'acceptation ou le rejet des documents : Si le score est supérieur au seuil, le document est accepté sinon il est rejeté. Les solutions proposées aujourd'hui sont de deux types : 1) Les solutions asynchrones ou différées, adoptées particulièrement dans le cas des systèmes de filtrage d'information par lots. Elles supposent avoir à leur disposition des collections de documents de référence que constituent des exemples positifs et/ou négatifs des besoins en informations des utilisateurs à partir desquelles elles peuvent estimer la valeur des seuils. Le seuillage est effectué soit en ligne soit de manière quasi-différée à la réception de chaque n documents. 2) Les solutions synchrones ou adaptatives, adoptées dans le cas des systèmes de filtrage adaptatif. Elles déduisent le seuil à partir des documents filtrés et cumulés au cours du processus du filtrage.

Cas de filtrage par lots

Les différentes stratégies de seuillage mises en œuvre dans le cadre de la classification supervisée peuvent être adoptées dans le filtrage par lots, celui-ci n'étant qu'un cas spécial de la classification à deux classes, pertinent et non pertinent. Dans ce cadre, trois principales stratégies de seuillage peuvent être distinguées dans la littérature (Yang, 1999) : 1) La stratégie de seuillage par validation croisée ; 2) la stratégie de seuillage proportionnelle orientée documents ; et 3) la stratégie de seuillage orientée classes.

La stratégie de *seuillage par validation croisée* consiste à diviser les exemples — positifs et négatifs — mis à la disposition du système de filtrage en deux ensembles disjoints : un pour l'apprentissage et l'autre pour la validation. Le seuil optimal au sens d'un critère d'évaluation (par exemple la mesure F_1 dans (Yang, 1999) et (Ault et Yang, 2001)) est déterminé sur l'ensemble de la validation. Plus précisément, tous les exemples de l'ensemble de la validation sont tirés par ordre décroissant du score de pertinence. En parcourant cette liste, il est possible de

calculer les valeurs d'un critère d'évaluation, et de chercher le document pour lequel cette valeur est maximale. Le score de ce document est utilisé comme seuil au-delà duquel un document sera jugé pertinent par le système.

Cette stratégie de seuillage a l'avantage d'être très simple à mettre en œuvre, et une fois qu'un seuil est déterminé (hors ligne), il peut être employé pour la tâche du filtrage en ligne. Néanmoins, un seuillage par validation croisée présente les inconvénients liés, d'un côté, à l'utilisation d'un ensemble de validation ce qui réduit le nombre d'exemples d'apprentissage, et, d'un autre côt, à l'absence de règles prédéfinies concernant la partition, i.e. la proportion d'exemples requise pour l'apprentissage et pour la validation. Cet effet peut-être particulièrement problématique dans le cas où le nombre d'exemples est insuffisant pour envisager d'en faire une partition. D'autre part, elle produit différents seuils au regard des différents critères d'évaluation, ou encore plusieurs seuils optimaux pour un même critère d'évaluation.

Une variante de cette stratégie est de maximiser un critère d'évaluation sur l'ensemble d'apprentissage en supposant que la distribution des scores sur cet ensemble est similaire à celle-ci sur l'ensemble de test (Schapire et al., 1998; Alpha et al., 2001). Dans de telles approches, les documents d'apprentissage — préalablement à leur utilisation pour l'optimisation du seuil — doivent être pondérés selon un schéma de pondération identique à celui utilisé pour les documents de test.

Pour sa part, la stratégie de *seuillage proportionnelle orientée documents* est moins adaptée au filtrage d'informations du fait qu'elle ne peut être appliquée que hors ligne. Elle consiste à attribuer dans un ensemble de documents à filtrer un nombre de documents pertinents proportionnel au nombre de ceux qu'il y avait dans l'ensemble d'apprentissage mis à la disposition du système. Les seuils de décision sont uniquement basés sur les probabilités a priori des exemples pertinents et non pertinents dans l'ensemble d'apprentissage ; ils ne sont pas appris ou optimisés en considérant les scores de pertinence ou le classement du système.

Enfin, la stratégie de *seuillage orientée classes* consiste à produire, pour chaque document, une liste des classes triée par ordre décroissant de pertinence vis-à-vis de ce document. Seules les t premières classes sont retenues. La valeur du paramètre t est comprise entre 1 et m (où m est le nombre de classes prédéfinies dans le système) et peut être estimée automatiquement en utilisant un ensemble de validation. Cette stratégie, bien que simple à mettre en ligne, présente l'inconvénient de ne pas pouvoir ajuster finement la différence entre la précision et le rappel. En effet, soit un ensemble de n documents, il existe soit n classes candidates, de même rang, qui seront toutes assignées aux documents correspondants soit aucune.

Une étude comparative entre les différentes stratégies de seuillage précitées a été menée dans (Yang, 2001). Selon cette étude, le seuillage par validation croisée est le plus adapté dès que le nombre d'exemples d'apprentissage est suffisant. Par contre, la stratégie de seuillage proportionnelle orientée documents semble mieux adaptée pour un nombre faible d'exemples d'apprentissage (petites classes), mais n'est pas applicable pour un fonctionnement en ligne. Enfin, la stratégie de seuillage orientée classes est la plus adaptée pour un fonctionnement en ligne, mais n'intègre aucune optimisation spécifique en terme de performances. C'est donc la stratégie de seuillage par validation croisée que nous adoptons dans nos expérimentations menées sur le filtrage par lots (cf. Section 5.1).

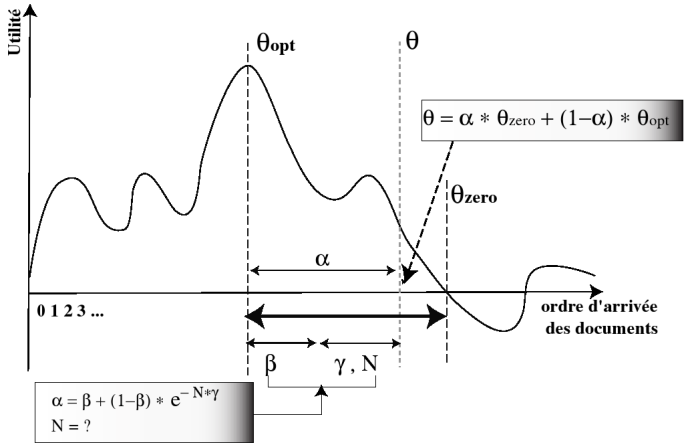


FIG. 3.7 – Une representation graphique de l'idée qui sous-tend la méthode de la régulation de beta-gamma pour la détermination du seuil θ par interpolation. Adapté de (Zhai et al., 1999).

Cas de filtrage adaptatif

En l'absence d'exemples d'apprentissage requis préalablement à l'initialisation du processus de filtrage, la détermination d'une fonction de décision est un des problèmes majeurs rencontrés dans le cas du filtrage adaptatif. La majorité des méthodes actuelles s'appuient sur des approches heuristiques (Zhai et al., 1999; Hoashi et al., 1999; Wu et al., 2001), la régression logistique (Robertson et Walker, 2000) ou la distribution des scores des documents déjà filtrés (Arampatzis et al., 2001; Zhang et Callan, 2001; Tebri et Boughanem, 2004). Par la suite, nous décrivons brièvement quelques-unes de ces méthodes, chacune ayant sa spécificité au niveau de l'initialisation et de l'adaptation de la fonction de décision ou de seuillage.

Dans le cadre du système CLARIT[21], Zhai et al. (1999) ont développé une méthode de seuillage appelée "*régulation de beta-gamma*". À l'étape initiale, le système utilise un seuil défini par un mécanisme dit "*taux de livraison*" dont le principe est d'estimer un seuil, à partir d'une collection de référence, de façon à permettre au système de délivrer une proportion bien déterminée de documents, r (p.ex. 3%), de la totalité des documents entrant en flux. Pour ce faire, les N documents de la collection de référence sont tirés par ordre décroissant de pertinence vis-à-vis du profil utilisateur. Le seuil initial est donc donné par le score du k-ème document ($k = r \times N$).

L'adaptation du seuil se fait avec la méthode de la régulation de beta-gamma qui calcule un seuil θ par interpolation linéaire entre un seuil optimal θ_{opt} et un seuil zéro θ_{zero} (cf. Figure 3.7). Ces deux seuils sont estimés à partir de l'ensemble des documents déjà filtrés par le système pour un profil utilisateur. Le seuil optimal (θ_{opt}) est le score qui permet d'obtenir une utilité maximale[22] sur cet ensemble de documents. Le seuil zéro, par contre, représente le plus grand seuil inférieur au seuil optimal permettant de fournir une utilité négative ou nulle, sur ce même

[21] L'apprentissage du profil dans le système CLARIT est basé sur une version incrémentale de l'algorithme de Rocchio, où seulement les documents pertinents sont considérés lors de l'apprentissage du profil. Plus précisément, le paramètre γ de Eq. 3.16 est mis à zéro et la reformulation du profil utilisateur est faite suivant Eq. 3.16 mais avec une sélection des k premiers termes dans le profil possédant les plus grands poids ; k varie en fonction du nombre de documents pertinents disponible pour l'apprentissage (Zhai et al., 1999).

[22] L'utilité est une mesure d'évaluation dédiée aux systèmes de filtrage d'information. Elle est fonction du nombre de documents pertinents et non pertinents filtrés par le système (Hull et Robertson, 2000).

ensemble de documents. L'interpolation entre ces deux seuils est faite de manière linéaire ; elle est donnée comme suit :

$$\theta = \alpha\ \theta_{zero} + (1 - \alpha)\ \theta_{opt} \tag{3.20}$$

Le facteur d'interpolation α est sensible au nombre de documents mis à la disposition du système pour calculer le seuil. Il est défini comme suit :

$$\alpha = \beta + (1 - \beta)\ e^{-\gamma N} \tag{3.21}$$

où β est un facteur de correction qui compense les scores relativement hauts des documents pertinents, et γ exprime l'hypothèse que le seuil optimal (θ_{opt}) approche la valeur optimale exacte d'autant plus que que le nombre d'exemples d'apprentissage dont le système dispose s'accroît. (Lors des expérimentations présentées dans (Zhai et al., 1999), les valeurs de β et γ sont respectivement fixés à 0.1 et 0.05). En fait, lorsque N est faible, la valeur de α augmente, ce qui permet de délivrer plus de documents pendant le filtrage. Avec le temps, le nombre de documents N augmente, et par conséquent, α décroît, ce qui fait augmenter la valeur du seuil et rend le système plus sélectif. La figure 3.7 illustre la représentation graphique de l'idée qui sous-tend les différentes formules précitées (Eq. 3.20 et Eq. 3.21).

Une autre manière pour la sélection et l'adaptation de la fonction de seuillage se base sur les distributions des scores des documents pertinents et non pertinents. L'idée de base est d'estimer les distributions des scores et puis d'optimiser une fonction d'utilité en fonction de ces distributions. Deux directions sont recensées dans la littérature. La première direction, SDS (Simple Distribution des Scores), suppose que la distribution des scores des documents pertinents suit une loi gaussienne tandis que la distribution des scores des documents non pertinents suit une loi exponentielle (Arampatzis et al., 2001; Zhang et Callan, 2001). L'estimation des paramètres des lois se fait de manière empirique à partir d'ensembles d'apprentissage (Arampatzis et al., 2001), ou à l'aide d'une méthode de maximum de vraisemblance (Zhang et Callan, 2001). L'inconvénient de cette direction est qu'il est difficile d'estimer la forme d'une distribution indépendamment des conditions expérimentales. De plus, elle nécessite souvent un nombre minimum de documents pour avoir des estimations non biaisées. La seconde direction, LDS (Linéarisation de la Distribution des Scores), suppose que la distribution des scores des documents pertinents et non pertinents est inconnue a priori, et tente de la "dessiner" ou de la construire en utilisant la méthode de régression linéaire permettant de transformer une distribution discrète en une densité de probabilités continue (Boughanem et al., 2001; Tebri et Boughanem, 2004).

Synthèse

Après cette présentation générale des méthodes d'apprentissage du profil utilisateur et des fonctions de seuillage dans le cadre du filtrage d'informations, la constatation que l'on peut faire actuellement est que la plupart de ces méthodes sont "*orientées pertinence*" plutôt que "*orientées utilisateur*". En d'autres termes, ces méthodes sont conçues dans le seul souci d'atteindre une fonctionnalité optimum en termes de critères d'évaluation de la pertinence du point du vue système, tels que la précision et le rappel (cf. Section 1.5.1). Bien qu'une telle fonctionnalité constitue une des caractéristiques essentielles que devraient satisfaire tous les systèmes de filtrage pour répondre efficacement aux besoins en informations des utilisateurs, d'autres caractéristiques importantes devraient également être recherchées pour une meilleure satisfaction des utilisateurs. Il s'agit notamment de pouvoir identifier le comportement de l'utilisateur à partir d'une analyse approfondie du contenu des documents qu'il a fournis en tant qu'exemples et/ou contre-exemples

de ses besoins spécifiques en informations. Ce type d'analyse peut aider à apporter des indications purement objectives sur les attentes réelles de l'utilisateur, qui se révèlent, à leur tour, utiles pour la mise en place d'un système de filtrage adapté aux caractères spécifiques de l'utilisateur et aux différents types de ses besoins en informations.

Notre travail s'inscrit donc dans cette direction. En fait, au delà de la mise en œuvre des méthodes dédiées à l'apprentissage du profil utilisateur, notre travail s'est orienté vers l'intégration de l'utilisateur comme une composante très importante dans la stratégie d'évaluation d'un système de filtrage d'informations. Le point de vue adopté est que le type de besoin de l'utilisateur peut aller d'un besoin très précis à un besoin très vague, voire contradictoire. En premier lieu, trois critères purement objectifs ont ainsi été développés pour caractériser le type de besoin de l'utilisateur en termes de précision, diversité et contradiction. Ces critères permettent ensuite de définir une méthode de seuillage basée sur la précision attendue par l'utilisateur du système et qui dépend directement du comportement de l'utilisateur et de son type de besoin en informations. Le détail de ces méthodes sera abordé dans le chapitre 5.

Nous allons maintenant aborder un autre domaine de recherche, qui s'intéresse aussi au traitement des flux documentaires mais qui dépasse largement le cadre des systèmes de filtrage traditionnels, en l'occurrence, la détection et suivi d'événements dans les flux documentaires.

3.4 Détection et suivi d'événements dans les flux documentaires

En 1996, la veille automatique des flux documentaires a été initiée par la DARPA (U.S. Defense Advanced Research Projects Agency) à travers le projet "*détection et suivi d'événements*" TDT (Topic Detection and Tracking). Ce projet a abordé, depuis son émergence jusqu'à son arrêt en 2004, certains problèmes particuliers liés à la détection et au suivi d'événements sur un flux documentaire de dépêches d'agences de presse ou de journaux télévisés (Allan et al., 1998). Dans ce cadre, les documents arrivent en flux soit de façon interrompue soit de façon continue sans aucune interruption entre eux. Chaque document, ou histoire, traite un événement de l'actualité. Un événement peut être vu comme un sujet d'actualité qui s'est produit à un moment bien précis ; les événements peuvent être prévus, ex. les élections politiques, ou imprévus, ex. éruption d'un volcan. Un thème, par contre, est défini par un ou plusieurs événements et constitue ainsi une des catégories utilisées pour classer les documents. Les thèmes peuvent être connus a priori, ou, au contraire, doivent être détectés par le système. En effet, certains des travaux menés dans ce cadre supposent que le flux à traiter porte sur un seul thème bien précis, tandis que d'autres travaux procèdent à une classification du flux en catégories thématiques — souvent en utilisant un algorithme d'apprentissage supervisé — et répètent séparément le même processus de détection et de suivi d'événements pour chacun des thèmes (Yang et al., 2002b).

Une des tâches du projet TDT est de segmenter un flux, si continu, en différents segments homogènes (documents ou histoires) traitant chacun un événement unique. La segmentation peut donc intervenir comme un préalable aux deux tâches suivantes référant à la détection de nouveaux événements survenant sur un flux de documents ainsi qu'à leur suivi. Dans les tâches TDT, la modélisation du temps joue un rôle important du fait que l'évolution temporelle des événements présente des variations importants. De manière similaire au cas des flux de données, la dimension temporelle est le plus souvent prise en compte en utilisant des fenêtres temporelles classiques ou pondérées (Bingham et al., 2003; Yang et al., 1998). Dans ce qui suit, nous pré-

sentons brièvement ces différentes tâches ainsi que les études menées dans le cadre du projet TDT.

3.4.1 Segmentation

La segmentation a fait l'objet de très nombreux travaux dans des domaines connexes concernant l'organisation de l'information, tels que le résumé automatique de documents, ou la recherche de passages pertinents dans les systèmes de questions-réponses, ou encore la recherche et le filtrage d'informations[23]. Dans le cadre du projet TDT, la segmentation est notamment concernée par le découpage d'un flux continu en parties homogènes successives traitant chacune un événement unique. En d'autres termes, il s'agit de détecter automatiquement les limites entre les documents (ou histoires) d'un flux continu de textes (Allan et al., 1998).

Les méthodes de segmentation thématique de textes, que ce soit dans le cadre du projet TDT ou d'autres applications, se répartissent en deux groupes dont la distinction tient principalement au type des connaissances qu'elles utilisent. D'une part, on trouve les méthodes qui se concentrent uniquement sur les caractéristiques inhérentes aux textes, telles que la récurrence des mots (Hearst, 1997), ou la réitération des entités nommées (Kan et al., 1998). Elles ne font pas appel à des connaissances externes et peuvent donc être utilisées sans restriction du domaine thématique des textes. Néanmoins, leurs performances peuvent être médiocres si le contenu thématique des textes est exprimé sous des formes trop diverses (synonymes, hyperonymes, etc.). D'autre part, on trouve des méthodes faisant appel à des connaissances sur les relations de cohésion lexicale des textes. Ces connaissances peuvent elles-aussi présenter l'avantage de ne pas dépendre d'un domaine thématique particulier, en prenant la forme d'un thésaurus (Morris et Hirst, 1991), ou d'un dictionnaire (Kozima, 1993). Or, la solution la plus simple mais aussi la plus précise consiste à exploiter des connaissances directement liées au domaine restreint des textes. Ces connaissances peuvent, par exemple, être apprises automatiquement à partir d'un ensemble de documents de référence traitant des thèmes susceptibles d'être rencontrés dans les textes à segmenter. C'est en particulier l'approche retenue dans le cadre du projet TDT (Yamron et al., 1998; Blei et Moreno, 2001; Beeferman et al., 2002).

Enfin, des approches hybrides combinant les différentes méthodes précitées ont également été envisagées et ont prouvé leur intérêt. À titre d'exemples, Ferret (2002) a proposé une méthode combinant la répétition de mots et la cohésion lexicale des textes pour la segmentation et la détection de liens entre événements. Jobbins et Evett (1998) exploitent conjointement la récurrence lexicale, l'utilisation de co-occurrences et celle d'un thésaurus.

3.4.2 Détection et suivi d'événements

La tâche de détection vise à identifier un nouvel événement au regard de ceux qui ont déjà été rencontrés dans un flux de documents. Cette tâche est réalisée de manière non supervisée, sans apport de connaissances a priori sur les événements que l'on cherche à détecter. La détection peut être effectuée de deux manières : une détection rétrospective ou une détection en ligne.

[23] En fait, la segmentation des documents en parties thématiquement homogènes peut améliorer la performance des systèmes de recherche et de filtrage d'informations en délivrant uniquement les parties pertinentes des documents qui correspondent au besoin en informations de l'utilisateur.

La première consiste à identifier rétrospectivement tous les événements à partir d'un ensemble de documents accumulés que représente le flux de documents. Les documents doivent donc être regroupés en clusters en fonction de l'événement qu'ils traitent. En partant de l'hypothèse que chaque document traite un événement unique, un document ne peut appartenir qu'à un seul cluster représentant d'un seul événement. Le fait qu'on dispose de l'intégralité des documents renvoie naturellement à l'utilisation des méthodes statiques non supervisées de type clustering pour accomplir cette tâche de détection.

La détection en ligne consiste, de son côté, à détecter instantanément l'apparition d'un nouvel événement dans un flux de documents arrivant en continu. La nouveauté d'un événement est définie par rapport aux événements déjà rencontrés sur le flux. Les documents arrivent donc dans leur ordre chronologique et doivent être classés à la volée. Un nouvel événement s'identifie à un document que l'on ne peut classer dans aucun des clusters déjà construits. Il est important de réduire le temps d'exécution de l'algorithme de détection, et d'accorder un score de pertinence à la décision qu'il prend sur la nouveauté ou non des événements.

D'une manière générale, on peut identifier deux composantes d'un système de détection d'événements : La première est dédiée à la représentation des documents ; la seconde est dédiée à la détection d'événements. L'approche majoritaire utilise le modèle vectoriel pour la représentation des documents, et des variantes du schéma de pondération TFIDF (Yang et al., 1998). Néanmoins, il est le plus souvent nécessaire d'avoir recours à des techniques particulières d'extraction de ce qu'on appelle "entités nommées". Les entités nommées dans un texte sont tout ce qui fait référence à un identifiant unique, tels que les noms de personnes, d'organisations, de lieux, mais aussi les dates, les quantités, les valeurs monétaires, les pourcentages etc. En fait, l'utilisation des entités nommées permet de mieux paramétrer la représentation des différents événements, du fait qu'étant des identifiants uniques, ils ont tendance à se répéter moins fréquemment d'un événement à l'autre.

Des considérations du même ordre interviennent lorsque l'on s'intéresse à la tâche du suivi d'événements. En effet, une fois qu'un événement a été détecté, on cherche à classer tous les documents se relatant à cet événement. Contrairement à la tâche de détection, les événements à classer sont connus a priori. Plus précisément, on fournit un certain nombre de documents que constituent des exemples positifs et négatifs relatifs à l'événement suivi. Les approches adoptées pour accomplir cette tâche se sont naturellement orientées vers des méthodes de classification supervisée d'événements.

Pour conclure

Il est important de souligner que les différentes tâches TDT sont fondamentalement non thématiques. En fait, la majorité des approches développées dans ce cadre sont orientées événements plutôt que thèmes ; et la représentation des événements n'est pas du ressort de l'analyse thématique des documents. Les méthodes correspondantes ne semblaient donc pas être adaptées de manière optimale pour résoudre les problèmes relatifs à la prospective de thèmes émergents et au suivi de l'évolution de ces derniers, que nous évoquerons plus loin dans le chapitre 6.

3.5 Conclusion

Dans ce chapitre, nous avons fait un tour d'horizon sur les différents travaux relatifs à l'analyse des flux de données documentaires. Dans la première partie de ce chapitre, nous avons présenté les éléments clés relatifs au prétraitement et à la représentation des données documentaires. Il a principalement été question des différentes stratégies d'indexation et des modèles de représentation des documents, ainsi que des méthodes de sélection et d'extraction des termes permettant la réduction de la dimensionnalité des données documentaires.

La deuxième partie du chapitre a essentiellement porté sur la problématique générale de l'accès personnalisé à l'information, en faisant notamment état des fondements du filtrage d'informations. Le problème majeur du filtrage d'information vient, d'une part, de la nature dynamique des données arrivant en flux et, d'autre part, de la variation continue des besoins en informations des utilisateurs. La résolution de ce problème a été envisagée selon deux modes principaux — le filtrage basé sur le contenu et le filtrage collaboratif — dont la distinction tient principalement au fait que l'on considère le contenu des documents ou des indicateurs plus subjectifs reflétant l'intérêt qu'ils présentent pour un utilisateur particulier. La réflexion sur les avantages et les inconvénients de chacun de ces deux modes a conduit à leur combinaison, ce qui en fait un filtrage dit "*hybride*". Après une brève présentation de ces différents modes de filtrage, nous nous sommes concentrés sur les étapes nécessaires à l'aboutissement d'un processus de filtrage basé sur le contenu, qui entre dans l'un des cadres applicatifs principaux de ce travail de thèse. Dans ce cadre, nous avons vu que la majorité des approches proposées dans la littérature se basent principalement sur des modèles issus de la recherche d'information auxquels se sont ajoutées des fonctionnalités d'adaptation des profils et des fonctions de seuillage. La constatation que nous avons faite à propos des approches existantes est qu'elles sont "*orientées système*" plutôt que "*orientées utilisateur*", ce qui rend impossible d'assurer l'entière satisfaction des utilisateurs. La tendance aujourd'hui est donc plutôt tournée vers l'intégration de l'utilisateur comme une dimension importante qui permet la conception de systèmes de filtrage orientés utilisateur en vue d'adapter son fonctionnement au contexte spécifique de l'utilisateur tout en maintenant la pertinence comme critère fondamental de la qualité des systèmes de filtrage d'informations.

Dans une troisième partie de ce chapitre, nous avons exposé la problématique de la détection et du suivi d'événements dans les flux documentaires, qui se rapproche fortement d'une des problématiques abordées dans le cadre de notre travail de thèse. Tandis que les approches et stratégies développées dans le cadre du projet TDT sont orientées événements, notre travail est plutôt relatif à l'analyse purement thématique des documents. En privilégiant l'aspect thématique du problème, notre but est de mettre en place une méthodologie générale de traitement des changements intervenants dans les flux documentaires, en termes d'émergence ou de déclin de thèmes, tout en permettant de rendre les méthodes développées suffisamment génériques pour pouvoir être appliquées indifféremment à n'importe quel type de données.

Les trois chapitres suivants présentent en détail notre contribution relative à l'analyse des flux de données multidimensionnelles, et en particulier, à celle des flux documentaires.

Chapitre 4

Modèle d'apprentissage fondé sur le principe de la détection de nouveauté

Tout au long des chapitres précédents, nous avons réalisé une analyse panoramique de l'état de l'art qui constitue le cadre théorique de notre travail. C'est à partir de l'étude de cet état de l'art que nous avons pu constater un manque évident de méthodes dotées d'un fonctionnement en ligne sans répétition d'apprentissage, adaptées aux spécificités du traitement des flux de données, notamment au niveau de la détection des changements susceptibles de survenir sur les flux de données proprement dits.

Ce chapitre présente un nouveau modèle d'apprentissage en ligne fondé sur le principe de la détection de nouveauté avec un mode d'apprentissage relatif qui n'exige typiquement que des exemples positifs issus d'une classe pour apprendre un modèle de classification. Le modèle d'apprentissage que nous avons développé est une adaptation d'un modèle de détection de nouveauté (NDF) proposé dès les années 1976. En plus de sa simplicité, ce modèle possède la caractéristique recherchée de fonctionner en ligne et sans répétition de l'apprentissage. Une première série d'expérimentations de validation en vraie grandeur, menées sur les données de l'annuaire ouvert DMOZ, ont démontré les bonnes capacités du modèle NDF dans le cas de classification avec des classes disjointes. Cependant, l'adaptation du modèle aux contraintes d'une classification avec un taux considérable de recouvrement entre les classes a nécessité une modification en profondeur du modèle initial, de manière à concevoir un modèle plus général, baptisé ILoNDF (acronyme de "Incremental data-driven Learning of Novelty Detector Filter"). De nouvelles méthodes d'initialisation, de nouvelles règles d'apprentissage et de nouvelles stratégies pour l'exploitation du modèle pour des fins de classification ont donc été proposées. Le grand avantage de la modification de la règle d'apprentissage du modèle original est lié à la capacité du modèle ILoNDF d'acquérir constamment de nouvelles connaissances relatives aux fréquences d'occurrence des variables et à leurs dépendances de co-occurrence dans les données utilisées pour faire l'apprentissage, ce qui rend le modèle robuste au bruit qui peut être présent dans la description des données d'apprentissage. En outre, le modèle ILoNDF ne comporte aucun paramètre à régler avant ou pendant l'apprentissage ; il n'y a donc aucun besoin de faire des calculs supplémentaires et coûteux en matière d'optimisation de paramètres.

La validation du modèle ILoNDF pour des fins de classification à partir d'une seule classe a été opérée sur deux collections standard de données textuelles : Reuters et WebKB. Le choix du traitement de ce type de données représente une option intéressante pour généraliser les résultats

obtenus à l'ensemble des données de type bruité et multidimensionnel.

Dans la discussion qui suit, une première partie est consacrée à la description des mécanismes de base qui gouvernent le fonctionnement du modèle NDF. Une deuxième partie introduit le modèle ILoNDF et illustre comment traiter les problèmes liés à l'apprentissage du modèle de base, NDF. Enfin, une expérimentation pilote a été mise en place afin d'étudier le bon fonctionnement de notre modèle d'apprentissage et de le comparer à celui d'autres méthodes typiques de classification à partir d'une seule classe.

4.1 Le modèle de filtre détecteur de nouveauté

4.1.1 Principe

En 1976, Kohonen et Oja (1976) ont introduit le premier modèle de détection de nouveauté comme modèle d'orthogonalisation qui laisse passer seulement les propriétés nouvelles d'une donnée par rapport à un ensemble de données de référence déjà connues ; les propriétés sont dites nouvelles si elles ne sont pas représentées dans les données de référence. Sur le plan théorique, le modèle de filtre détecteur de nouveauté se fonde sur les propriétés des opérateurs de projection orthogonale dans l'espace vectoriel $\mathbb{R}^n$: Soient $x_1, x_2, \ldots, x_m$, m vecteurs distincts de $\mathbb{R}^n$ engendrant un sous-espace $\zeta \subset \mathbb{R}^n$. Le sous-espace complémentaire de ζ, noté $\zeta^\perp$, est engendré par l'ensemble des vecteurs de $\mathbb{R}^n$ orthogonaux à ζ. N'importe quel vecteur $x \in \mathbb{R}^n$ peut uniquement être décomposé en une somme de deux vecteurs orthogonaux $\hat{x} \in \zeta$ et $\tilde{\mathrm{x}} \in \zeta^\perp$. Une propriété caractéristique des opérateurs de projection orthogonale est que, parmi toutes les décompositions possibles de x, $x = \dot{x} + \ddot{x}$ où $\dot{x} \in \zeta$, celle indiquée ci-dessus est sujette à :

$$\|\tilde{x}\| = \min_{\|\dot{x}\|} \|\ddot{x}\|$$

C'est-à-dire, la norme du vecteur $\tilde{\mathrm{x}}$ est minimum et équivalente à la distance de x au sous-espace ζ. La composante $\hat{\mathrm{x}}$ du vecteur x représente la projection orthogonale de x sur ζ tandis que la composante $\tilde{\mathrm{x}}$ représente la projection orthogonale de x sur $\zeta^\perp$. La représentation matricielle des opérateurs de projection orthogonale fournit une indication pour le calcul des vecteurs $\hat{\mathrm{x}}$ et $\tilde{\mathrm{x}}$, comme suit. Soit une matrice $X \in \mathbb{R}^{n \times m}$ dont les colonnes sont les vecteurs x_i ; et soit X^+ la pseudo-inverse de X[24] ; alors XX^+ est un opérateur orthogonal qui permet de représenter la projection d'un vecteur x sur ζ sous la forme suivante :

$$\hat{x} = XX^+x \tag{4.1}$$

De manière analogue, $I - XX^+$ est l'opérateur qui représente la projection de x sur $\zeta^\perp$:

$$\tilde{x} = (I - XX^+)\, x \tag{4.2}$$

où I désigne la matrice identité d'ordre n.

En vertu des principes d'orthogonalisation évoqués ci-dessus, la composante $\tilde{\mathrm{x}}$ peut être considérée comme la contribution résiduelle de x qui subsiste après l'application d'une forme de

[24]Les matrices pseudo-inverses de Moore-Penrose représentent une généralisation des matrices inverses au cas de matrices singulières ou rectangulaires. Pour toute matrice, il a été prouvé dans (Penrose, 1955) que la pseudo-inverse existe et est unique. Il existe beaucoup de méthodes pour trouver la pseudo-inverse de Moore-Penrose dont le théorème de Greville qui sera présenté en section 4.1.2.

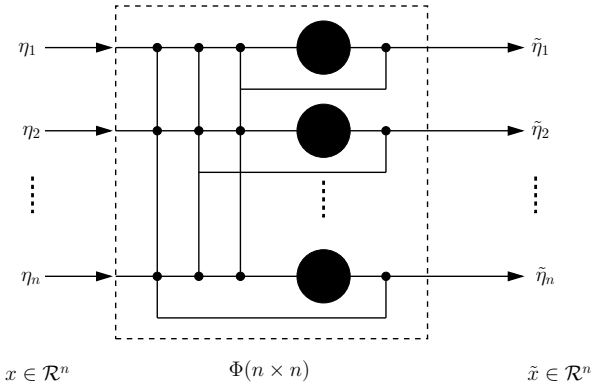


FIG. 4.1 – L'architecture neuronale du modèle de filtre détecteur de nouveauté NDF

traitement particulière à x. Ainsi, si un système est conçu dont la fonction de transfert est la matrice $(I - XX^+)$ définie précédemment dans Eq. 4.2, il sera possible de dire que la sortie du système ($\tilde{\mathrm{x}}$) correspond à la composante de x qui est maximalement nouvelle ou indépendante en regard des autres vecteurs x_i. Dans (Kohonen et Oja, 1976), un modèle de réseau neuronal, NDF, est conçu pour mettre en application le principe de la détection de nouveauté. Le comportement du modèle s'avère équivalent, sous certaines conditions, à celui des opérateurs de projection orthogonale. La section suivante décrit l'architecture du modèle neural, à savoir du modèle de filtre détecteur de nouveauté NDF.

4.1.2 Implémentation du modèle NDF

L'implémentation physique du modèle NDF est faite par un réseau récurrent de n neurones élémentaires, étroitement connectés en forme de boucles rétroactives, ou feedback, entre les neurones (cf. Figure 4.1). Les neurones sont à la fois des neurones d'entrée et de sortie. Ils reçoivent les données en entrée sous la forme de vecteurs de $\mathbb{R}^n$; et constituent respectivement la sortie du réseau. La sortie de chaque neurone est déterminé par une combinaison linéaire de l'entrée externe η_i et du feedback qu'il reçoit de la sortie :

$$\tilde{\eta}_i = \eta_i + \sum_j m_{ij}\tilde{\eta}_j \tag{4.3}$$

Les poids des connexions rétroactives m_{ij} caractérisent l'état interne du réseau. Ils sont initialisés à zéro et, ensuite, mis à jour après la présentation de chaque donnée en entrée du réseau selon une règle d'apprentissage de type anti-Hebbian :

$$\frac{dm_{ij}}{dt} = -\alpha\tilde{\eta}_i\tilde{\eta}_j \tag{4.4}$$

où α est un coefficient positif qui peut être modifié de manière adaptative au cours de l'apprentissage. Cette règle revient à atténuer l'activation des neurones fortement corrélés en renforçant leurs interconnexions inhibitrices. Cette décorrélation peut ainsi conduire à supprimer l'activation simultanée et redondante des neurones correspondant à des variables qui décodent le recouvrement entre données semblables. En plus, les connexions réctroactives bouclées sur les neurones eux-mêmes ont pour effet de réduire l'activation correspondant à des variables apparaissant individuellement dans les données d'entrée. Le module ci-dessus peut maintenant être exprimé en

tant qu'une matrice $\Phi \in \mathbb{R}^{n \times n}$ selon les équations suivantes :

$$\tilde{x} = x + M\tilde{x} = (I - M)^{-1}x = \Phi x \tag{4.5}$$

$$\frac{dM}{dt} = -\alpha \tilde{x}\tilde{x}^T$$

Ce qui permet de dériver l'équation différentielle pour Φ qui est donnée par :

$$\frac{d\Phi}{dt} = -\alpha \Phi^2 x x^T \Phi^T \Phi \tag{4.6}$$

Cette équation est identifiable à une équation de Bernouilli du 4-ième degré. Bien que sa résolution soit difficile dans le cas général, Kohonen et Oja (1976) montrent l'existence de solutions asymptotiques stables si $\alpha \geq 0$. Des solutions approximatives peuvent être obtenues en faisant certaines hypothèses sur les données d'entrée et les conditions initiales de la matrice Φ. Ces hypothèses sont les suivantes :

1. Chaque donnée d'entrée x est présentée au réseau pendant une période suffisamment longue ;
2. La matrice Φ_0 est initialisée comme une matrice de projection, i.e. symétrique et idempotente[25]. De ce fait, la matrice Φ peut approximativement converger vers une autre matrice de projection dans l'espace $\mathcal{R}(\Phi_0) \cap \zeta^{\perp}$, où $\mathcal{R}(\Phi_0)$ est l'espace de Φ_0 et $\zeta^{\perp}$ est l'espace complémentaire de l'espace engendré par les vecteurs x_i (cf. section 4.1.1).

En particulier, un type spécial de matrices de projection qui concorde avec la définition du modèle de filtre de nouveauté, est la matrice identité. En effet, il est logique que celui-ci se comporte dans les conditions initiales comme un filtre identité, puisque toute donnée présentée au filtre peut être considérée comme entièrement nouvelle[26]. Dans ces conditions, il vient que la valeur de stabilité de Φ est :

$$\Phi_c = I - XX^+ \tag{4.7}$$

L'équation Eq. 4.7 représente l'équation caractéristique de l'opérateur de projection sur l'espace vectoriel $\mathcal{R}(I) \cap \zeta^{\perp} = \zeta^{\perp}$ qui est orthogonal à l'espace engendré par les vecteurs x_i. Le lecteur intéressé peut se référer à (Kohonen, 1989) pour plus de détails et de justifications théoriques du modèle.

Sur le plan du calcul numérique, ce modèle neural ne constitue pas une solution entièrement satisfaisante pour le traitement des données fortement multidimensionnelles, compte tenu du fait que le réseau est entièrement connecté et le nombre de neurones est égal à la dimensionnalité de l'espace d'entrée. Pour réduire la complexité du calcul, une solution plus simple et surtout plus rapide consiste à calculer directement l'état stable du filtre Φ_c en utilisant un des algorithmes existants pour trouver la pseudo-inverse de Moore-Penrose (Ben-Israel et Greville, 2003). Dans notre travail nous nous sommes plus particulièrement intéressés par le théorème de Greville (Greville, 1960), pour deux raisons : 1) il s'agit d'un algorithme applicable en ligne ; 2) il s'avère être le plus rapide en temps de calcul (Noda, 1997). Le théorème de Greville fournit un calcul récursif de la pseudo-inverse de Moore-Penrose qui mène à une expression récursive de la fonction de transfert du modèle de filtre détecteur de nouveauté, NDF, cf. Eq. 4.7, comme décrit ci-dessous.

[25] Une matrice P est dite idempotente si elle possède la propriété $P^2 = P$.

[26] Cela revient également à considérer que les connexions entre les neurones ne sont pas initialement opérationnelles du fait que leurs poids sont mis à zéro.

Partitionnons une matrice X_k dont les colonnes sont les vecteurs $x_1, x_2, \ldots, x_k$, sous la forme :

$$X_k = [x_1, x_2, \ldots, x_k] = [X_{k-1}, x_k] \tag{4.8}$$

de sorte que la matrice X_{k-1} comporte les $k-1$ données précédemment apprises par le filtre, tandis que x_k ne l'est pas encore. Le théorème de Greville déclare que :

$$X_k^+ = \begin{bmatrix} X_{k-1}^+(I - x_k \mathcal{P}_k^T) \\ ---------- \\ \mathcal{P}_k^T \end{bmatrix}, \qquad \text{où} \tag{4.9}$$

$$\mathcal{P}_k = \begin{cases} \dfrac{(I - X_{k-1}X_{k-1}^+)x_k}{\|(I - X_{k-1}X_{k-1}^+)x_k\|^2} & \text{si le numérateur} \neq 0 \\ \\ \dfrac{(X_{k-1}^+)^T X_{k-1}^+ x_k}{1 + \|X_{k-1}^+ x_k\|^2} & \text{sinon.} \end{cases}$$

et la récursion est initialisée par

$$X_1^+ = \begin{cases} x_1^T (x_1^T x_1)^{-1} & \text{if} \quad x_1 \neq 0 \\ \\ 0^T & \text{if} \quad x_1 = 0 \end{cases}$$

De là, il suit donc que

$$X_k X_k^+ = X_{k-1} X_{k-1}^+ (I - x_k \mathcal{P}_k^T) + x_k \mathcal{P}_k^T \tag{4.10}$$

Notons que, si $(I - X_{k-1}X_{k-1}^+)x_k$ est un vecteur nul, l'équation (4.10) se résume à $X_k X_k^+ = X_{k-1}X_{k-1}^+$; autrement, la partie supérieure de $\mathcal{P}_k$ dans Eq. 4.9 y est appliquée. Dans les deux cas, l'expression récursive de la valeur du filtre Φ, après la présentation de k données d'apprentissage, $x_1, \ldots, x_k$, peut s'écrire sous la forme suivante :

$$\begin{aligned} \Phi_k &= I - X_k X_k^+ \\ &= (I - X_{k-1}X_{k-1}^+) - \frac{(I - X_{k-1}X_{k-1}^+)x_k x_k^T (I - X_{k-1}X_{k-1}^+)}{\|(I - X_{k-1}X_{k-1}^+)x_k\|^2} \end{aligned} \tag{4.11}$$

Si l'on considère que $\Phi_{k-1} = I - X_{k-1}X_{k-1}^+$ représente la fonction de transfert de NDF après la présentation des $k-1$ premières données, il est possible de réécrire Eq. 4.11 sous la forme simplifiée suivante :

$$\Phi_k = \Phi_{k-1} - \frac{\tilde{x}_k \tilde{x}_k^T}{\|\tilde{x}_k\|^2} \tag{4.12}$$

où $\tilde{x}_k = \Phi_{k-1} x_k$ représente la projection orthogonale du vecteur x_k sur un espace orthogonal à l'espace défini par les $k-1$ premières données apprises par le filtre ; et la valeur initiale de la matrice du filtre s'écrit : $\Phi_0 = I$.

La proportion de nouveauté qu'il est possible d'associer à une donnée d'entrée x_i, relativement à un ensemble de données déjà apprises par le filtre x_k, peut être obtenue à partir de la norme du vecteur de nouveauté $\tilde{x}_i$ associé à cette donnée :

$$N_{x_i} = \frac{\|\tilde{x}_i\|}{\|x_i\|} \tag{4.13}$$

La proportion complémentaire, à savoir la proportion d'habituation $H_{x_i} = 1 - N_{x_i}$, peut être considérée comme indicateur de similarité (habituation) de x_i vis-à-vis des données précédemment apprises.

Pour récapituler, le fonctionnement du modèle NDF dont la fonction de transfert est donnée par Eq. 4.12, et dont l'entrée et la sortie sont reliées par $\tilde{x} = \Phi_k x$, peut se résumer comme suit. Pendant la phase d'apprentissage, le modèle NDF s'habitue aux données de référence présentées en entrée. Une fois l'apprentissage terminé, si une des données de référence ou une de leurs combinaisons linéaires est appliquée à l'entrée du modèle, la sortie correspondante sera nulle. D'autre part, si une donnée n'appartenant pas à l'espace formé par les données de référence est choisie comme entrée, la sortie correspondante ne sera pas nulle et elle peut être vue comme représentative des variables nouvelles extraites à partir de la donnée d'entrée vis-à-vis des données de référence qui ont été déjà apprises.

4.2 Réflexion sur les forces et les faiblesses du modèle NDF

Le principe du modèle NDF est particulièrement intéressant pour l'analyse des flux de données du fait de son mode de fonctionnement en ligne sans répétition de l'apprentissage. Par ailleurs, NDF ne comporte aucun paramètre à régler avant ou pendant l'apprentissage, il n'y a donc aucun besoin de faire des calculs supplémentaires et coûteux en matière d'optimisation de paramètres.

D'inspiration biologique de la capacité d'orthogonalisation de l'hippocampe[27], NDF tend à agir en tant que mémoire à long terme reconnaissant rapidement de données préalablement apprises. Néanmoins, NDF n'est pas biologiquement plausible dans le sens où il ne reflète pas la capacité d'oubli lié au temps qui est caractéristique de tous les mécanismes de mémorisation biologiques. Compte tenu de ces caractéristiques, le modèle NDF aboutit au comportement suivant :

- Les données qui se présentent occasionnellement au modèle, seront toujours reconnues comme familières après leur première occurrence, quelle soit la fréquence d'occurrence et quel que soit l'intervalle de temps entre les deux dernières occurrences. Dans le cadre de la classification à partir d'une seule classe, un tel comportement requiert que les exemples positifs utilisés pour l'apprentissage ne soient pas du tout bruités, ce qui n'est certainement pas réaliste, autrement, NDF ne sera pas en mesure de fournir un modèle fiable de la classe positive.
- Le fait que NDF n'a pas la capacité d'oubli des données déjà apprises rend inefficace le traitement des données issues des environnements dynamiques où la distribution des données positives change au cours du temps, tels que dans beaucoup d'applications robotiques (Marsland et al., 2000).
  Il est à noter à ce titre qu'une adaptation du modèle NDF dans le but de lui conférer une capacité d'oubli a été envisagée dans (Kohonen, 1989). Cette adaptation visait à introduire

[27] L'hippocampe est une des régions cérébrales qui paraît jouer un rôle important dans les processus d'apprentissage, de mémorisation, et de rappel, tels que la formation des mémoires à long terme et la récupération des traces mémorielles. L'hippocampe s'est également avéré jouer un rôle important dans la faculté de détection de nouveauté qu'a le cerveau (Sirois et Mareshal, 2004). Une telle faculté implique évidemment la constitution de nouvelles mémoires et la récupération des stimuli déjà reçus. Le cortex périrhinal est une autre région cérébrale appartenant au système hippocampique, qui est probablement impliquée dans la détection de nouveauté (Brown et Xiang, 1998).

un facteur d'oubli ($\beta > 0$) qui fait diminuer les poids de toutes les connexions rétroactives selon un taux directement proportionnel à leurs valeurs. La règle d'apprentissage, en notation matricielle, alors, devient :

$$\frac{dM}{dt} = -\alpha\tilde{x}\tilde{x}^T - \beta M \tag{4.14}$$

Dans ce cas, on obtient, au lieu de Eq. 4.6, une équation différentielle plus générale de la forme :

$$\frac{d\Phi}{dt} = -\alpha\Phi^2 x x^T \Phi^T \Phi + \beta(\Phi - \Phi^2) \tag{4.15}$$

Malheureusement, cette équation n'est pas facile à résoudre et ne semblait pas compatible avec les propriétés nécessaires à la convergence (Aeyels, 1990). En outre, Kohonen (1989) montre que la solution asymptotique de l'équation (4.15) — dans les mêmes conditions que NDF — est identique à celle obtenue avant l'intégration du facteur d'oubli, et ainsi, cette solution conduit aussi à une matrice représentant approximativement une opérateur de projection.

- Le temps caractéristique de l'habituation du modèle NDF à une variable — qui correspond au nombre de données d'apprentissage contenant cette variable et qui ont été présentées à l'entrée du modèle — est très court (inversement propositionnel au nombre de variables présentes dans les données). Une telle habituation, très rapide, présente un inconvénient pour certaines applications où les données d'apprentissage peuvent être caractérisées par des variables non pertinentes apportant du bruit, comme c'est particulièrement le cas des données documentaires et bioinformatiques. De fait, la règle d'apprentissage du modèle NDF, cf. Eq. 4.12, est principalement destinée à distinguer d'une manière quasi absolue les variables nouvelles des variables habituées ou apprises dans une donnée d'entrée au regard des données qui ont été déjà apprises. Par conséquent, seules les nouvelles variables sont passées à travers le filtre et ce sont elles seules qui contribueront à la modification de son état interne à travers les connexions rétroactives. Non seulement cela permettrait d'accélérer l'habituation du filtre aux variables dès sa première présence dans une donnée d'entrée, mais aussi cela ne lui permet pas d'améliorer ses connaissances concernant les variables déjà apprises. Dans un tel cas, la plupart des variables pertinentes seront totalement habituées après la présentation d'un certain nombre très restreint de données d'apprentissage et, par conséquent, leur présence dans les données qui suivent sera complètement ignorée, ce qui permet une habituation plus rapide aux variables moins pertinentes ou bruitées qui peuvent être présentes dans les données d'apprentissage[28]. Il en ressort que toutes les variables, qu'elles soient pertinentes ou non, ont à peu près le même niveau d'habituation, et donc, ont une égalité d'importance dans la modélisation des données d'apprentissage.

Pour illustrer les différents points évoqués ci-dessus, nous donnons deux exemples concrets démontrant le comportement du modèle NDF à travers l'étude de deux cas simples.

[28] Ceci peut expliquer la tendance du modèle NDF à être moins efficace lorsque la dimensionnalité des données est plus élevée, comme il sera démontré dans la section 4.8, dédiée aux résultats expérimentaux.

Cas 1 : Jeux de données #1 et #2

Supposons que les données d'apprentissage du modèle sont constituées de trois données, notées d_1, d_2, d_3, $\in \Re^3$, dans le tableau ci-dessous :

Jeu de données #1

	d_1	d_2	d_3
v_1	1	1	1
v_2	0	1	1
v_3	0	0	1

À l'issue de l'apprentissage du modèle sur les données ci-dessus, la fonction de transfert devient une matrice nulle, $\Phi_3 = 0$. De ce fait, l'espace de nouveauté est l'espace nul, ce qui signifie que toutes les données d'apprentissage et leurs combinaisons linéaires sont totalement habituées ($\tilde{d}_1, \tilde{d}_2$ et $\tilde{d}_3$ sont des vecteurs nuls). En particulier, si l'on calcule les proportions de nouveauté et d'habituation des variables utilisées dans la représentation des données d'apprentissage, v_1, v_2 et v_3, en projetant le vecteur unité $\overrightarrow{u_v}$ associé à chacune des variables sur l'espace de nouveauté :

$$N_v = \frac{\|\Phi_3 \overrightarrow{u_v}\|}{\|\overrightarrow{u_v}\|} , \quad H_v = 1 - N_v$$

On obtient :

$$H_{v_1} = H_{v_2} = H_{v_3} = 1$$

Cela signifie que toutes les variables v_1, v_2 et v_3 ont été totalement habituées et auront donc une importance équivalente dans la modélisation des données d'apprentissage, et ceci malgré le fait que v_1 apparaisse plus fréquemment dans les données que v_2 et v_3, et que v_3 (la moins fréquente) pourrait être due au bruit. Ce comportement, comme expliqué précédemment, est dû à l'apprentissage rapide des variables, caractéristique du modèle NDF. Ainsi, une fois qu'une variable est totalement habituée par le modèle, elle ne sera plus considérée lors de la modification de l'état interne du filtre, c'est à dire que la présence ou l'absence de telles variables dans les données qui suivent n'a pas d'effet lors des étapes ultérieures de l'apprentissage. Ce problème vient du fait que l'apprentissage est guidé uniquement par le vecteur de nouveauté. En d'autres termes, l'apprentissage du modèle NDF sur les données suivantes :

Jeu de données #2

	d_1	d_2	d_3
f_1	1	0	0
f_2	0	1	0
f_3	0	0	1

aboutira aux mêmes résultats obtenus avec les données précédentes (Jeu de données #1).

Pour conclure sur ce point, il faut noter que le problème de l'apprentissage du modèle NDF ne peut pas être résolu en augmentant le nombre de données d'apprentissage puisque la matrice du filtre est nulle, et ainsi, la projection d'une donnée quelle qu'elle soit, sur cette matrice est certainement un vecteur nul.

Cas 2 : Jeu de données #3

Supposons maintenant que les données suivantes soient utilisées pour l'apprentissage du modèle NDF :

Jeu de données #3

	d_1	d_2	d_3	d_4	d_5	d_6	d_7 ...
v_1	1	1	0	0	0	0	0
v_2	0	1	0	0	0	0	0
v_3	0	0	1	0	1	1	1
v_4	0	0	0	1	1	1	0
v_5	0	0	0	0	1	0	1

Il est facile de vérifier qu'après l'apprentissage du modèle NDF avec les deux premières données d_1 et de d_2, le modèle devient totalement habitué à ces données aussi bien qu'à leurs variables descriptives v_1 et v_2. Si maintenant l'apprentissage poursuit avec le reste des données qui sont représentées par des variables différentes, le modèle NDF s'y habituera tout en se souvenant des variables précédemment apprises (v_1 et v_2). Un tel comportement n'est pas pertinent pour la classification du fait que les variables sont autant apprises, qu'elles soient fréquemment vues ou non dans la description des données d'apprentissage. Notons aussi qu'un tel comportement est particulièrement problématique dans le cas de la classification des flux de données dynamiques où il est nécessaire d'oublier progressivement les caractéristiques liées aux données qui n'apparaissent plus, ou moins fréquemment, dans les flux entrants.

4.3 Le modèle d'apprentissage incrémental ILoNDF

Il est évident, au vu de la discussion ci-dessus, qu'une modification de la règle d'apprentissage du modèle NDF est absolument nécessaire. Lors de la conception de la nouvelle règle d'apprentissage, deux points critiques devraient être pris en considération : 1) toutes les variables présentes dans les données d'entrée devraient être considérées au cours de l'apprentissage, qu'elles soient précédemment apprises ou non. Pourtant 2) les connaissances acquises concernant la nouveauté des données/variables devraient constamment être prises en compte pour continuer à maintenir le principe de la détection de nouveauté. Pour répondre aux exigences précitées, nous avons envisagé de nombreuses adaptations de la règle d'apprentissage du modèle NDF ; deux parmi elles se sont avérées concluantes pour corriger les défauts majeurs de la règle originale du modèle évoqué précédemment.

La première adaptation que nous avons envisagée a été guidée par la réflexion que le problème du modèle NDF est directement lié à l'habituation rapide aux variables des données d'apprentissage, qui mène très souvent à une saturation totale du modèle (ce qui correspond au cas où Φ=0, i.e. l'espace de nouveauté est équivalent à l'espace nul). La raison principale en est que la fonction de transfert du modèle est initialisée à une matrice identité Φ_0=I. Ainsi, si une variable v_i apparaît une fois seule dans une des données d'apprentissage (ou quelques fois avec d'autres variables) elle devient totalement habituée par le modèle (Φ_{ik}=0, Φ_{ki}=0, $\forall k$), et ainsi, elle ne sera plus prise en considération pendant les étapes ultérieures de l'apprentissage. De ce fait, la solution la plus adaptée serait de conserver la règle d'apprentissage originale du modèle NDF mais d'initialiser la récurrence à une matrice scalaire dont les éléments de la diagonale principale

sont égaux au nombre de données d'apprentissage. Les résultats obtenus expérimentalement avec cette solution sont un peu inférieurs à ceux obtenus avec notre seconde adaptation du modèle que nous présentons ci-après (cf. Eq. 4.16). De plus, dans cette première solution, le nombre de données d'apprentissage est supposé connu à l'initialisation du processus d'apprentissage ce qui n'est certainement pas le cas pour les applications fonctionnant en ligne.

De son côté, la seconde direction s'est posée du fait que l'apprentissage du modèle NDF est guidé uniquement par la nouveauté : L'adaptation de l'état interne du modèle NDF décrite par Eq. 4.12 ne considère que les variables représentant la nouveauté qu'apportent les données d'entrée (vecteur de nouveauté $\tilde{x}$). Pour faire participer toutes les variables présentes dans les données d'entrée au processus d'apprentissage, la meilleure stratégie que nous avons trouvée est d'introduire la matrice identité à chaque étape de l'apprentissage et de projeter les données d'entrée à la fois sur la matrice identité et sur la matrice du filtre. La nouvelle expression de la règle d'apprentissage s'écrit sous la forme :

$$\Phi_k = I + \Phi_{k-1} - \frac{\tilde{x}_k \tilde{x}_k^T}{\|\tilde{x}_k\|^2} \tag{4.16}$$

où $\tilde{x}_k = (I + \Phi_{k-1})x_k$ et Φ_0 est une matrice nulle. De cette façon, nous pouvons prétendre satisfaire la première exigence en projetant chacune des données d'apprentissage x_k sur la matrice identité, ce qui permet à chacune des variables présentes dans les données de contribuer au processus de mise à jour de l'état interne du filtre, indépendamment de la quantité de nouveauté ou de redondance qu'elles exhibaient. Cette démarche est particulièrement importante pour traiter les situations dans lesquelles des variables/données sont totalement apprises et ainsi leur projection sur l'espace engendré par la matrice du filtre est nulle. La seconde exigence est également remplie en projetant simultanément les données d'entrée x_k sur la matrice du filtre correspondant aux données antérieures. Pendant que l'apprentissage progresse, deviennent de plus en plus habituées les variables qui apparaissent fréquemment dans les données d'apprentissage par rapport à celles moins fréquentes. Cela permet, par exemple dans le contexte de la classification, de distinguer plus facilement les données positives des données négatives. Du fait que l'apprentissage est maintenant guidé non seulement par la nouveauté ($\Phi_{k-1}x_k$) mais également par les données elles-mêmes ($Ix_k \equiv x_k$) ; le modèle peut constamment acquérir de nouvelles connaissances relatives aux données d'apprentissage, et de ce fait, il peut être justifié d'appeler ce modèle "*Incremental data-driven Learning of NDF*" (ILoNDF).

L'étude du comportement du modèle ILoNDF dans l'espace des vecteurs propres de la matrice représentant la fonction de transfert du modèle fournit des éléments de description supplémentaires mettant en évidence les apports essentiels de ce modèle par rapport au modèle original. En fait, il est bien connu que si $\{\lambda_i\}$ sont les valeurs propres d'une matrice carrée X, alors $\{\lambda_i + \lambda\}$ sont les valeurs propres de la matrice $X + \lambda I$; en outre, les matrices X et $X + \lambda I$ ont une base orthonormée commune des vecteurs propres $\{u_i\}$. De ce fait, les matrices $\{\Phi_{k-1}\}$ et $\{I + \Phi_{k-1}\}$ intervenant dans l'équation 4.16 ont des vecteurs propres communs et leurs vecteurs propres correspondantes diffèrent par un facteur de 1. En d'autres termes, l'introduction de la matrice identité ne change pas la direction des vecteurs propres des matrices $\{\Phi_{k-1}\}$ mais les transforme en matrices définies positives $\{I + \Phi_{k-1}\}$, avec des valeurs propres différentes de zéro, tout en conservant l'ordre relatif des valeurs propres. En conséquence, les vecteurs propres qui ne sont pas orthogonaux à l'espace engendré par les données d'apprentissage résident dans l'espace engendré par le modèle mais restent toujours moins significatifs que d'autres vecteurs propres de directions orthogonales à l'espace des données. À présent, il est bien évident que les conditions

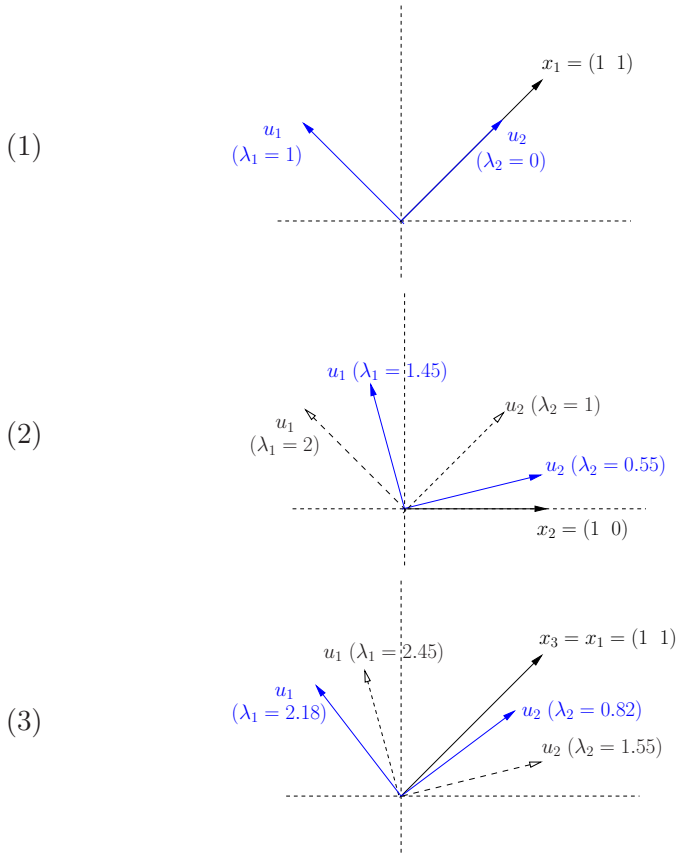


FIG. 4.2 – Illustration du comportement du modèle ILoNDF dans l'espace des vecteurs propres de la matrice correspondant à son état interne. Les données d'apprentissage x_1=(1 1), x_2=(1 0) et x_3=(1 1) sont respectivement présentées aux étapes (1), (2) et (3). Les vecteurs propres sont notés par u_1, u_2 et les valeurs propres correspondantes par λ_1, λ_2.

d'orthogonalité entre l'espace engendré par le modèle et celui engendré par les données d'apprentissage ne sont plus remplies ; au contraire, la dimensionnalité de l'espace de nouveauté engendré par le modèle sera toujours du même ordre que celle de l'espace de représentation des données.

La figure 4.2 montre, à titre d'illustration, le comportement du modèle ILoNDF dans l'espace des vecteurs propres de la matrice d'état interne du modèle. Le modèle agit à la première étape comme NDF : après l'apprentissage sur x_1, la matrice Φ_1 possède deux vecteurs propres ; l'un est orthogonal à la direction de ladite donnée x_1, l'autre est dans la direction de x_1 avec une valeur propre de 0. Comme nous l'avons mentionné plus haut, l'introduction de la matrice identité fait augmenter les valeurs propres de la matrice Φ_1 de 1. C'est à dire que la quantité d'informations portée par le vecteur propre u_1 devient seulement deux fois plus grande que la quantité de l'information portée par le vecteur propre u_2. Dans la deuxième étape, après l'apprentissage du modèle sur x_2, les vecteurs propres de la matrice s'orientent dans différentes directions avec différentes valeurs propres pour s'adapter à la représentation globale des données d'apprentissage. Plus précisément, u_1 représente 72.4% d'informations, le reste est représenté par u_2 qui ne couvrent que 27.6% d'informations. Maintenant si x_1 (ou x_2 ou encore toute combinaison linéaire de x_1 et x_2) est présenté à nouveau au modèle, il ne sera pas considéré comme redondant. Le modèle parvient

à adapter continuellement ses connaissances relatives aux données d'apprentissage et à leurs variables associées. Si l'on contraste ce comportement avec celui du modèle NDF, on trouve que, après son apprentissage sur x_1 et x_2, le modèle NDF s'est complètement habitué à l'ensemble des variables de l'espace de représentation de ces données et que l'espace engendré par ce NDF est l'espace nul, $\Phi_2 = 0$. Il n'est donc plus possible d'extraire des informations nouvelles si l'on présente l'une des combinaisons linéaires de x_1 et de x_2 au modèle.

Un autre aspect intéressant est survenu suite à la modification de la règle d'apprentissage du modèle original, NDF. Cet aspect concerne la capacité du modèle ILoNDF de capturer les relations de co-occurrence existant entre les variables des données d'apprentissage. En fait, une analyse rigoureuse de l'apprentissage du modèle révèle que, pendant l'apprentissage, si deux termes i et j apparaissent ensemble dans une des données d'apprentissage, la valeur du jk-ième élément de la matrice du filtre diminuera. Dans les étapes d'apprentissage qui suivent, si la variable i apparaît avec une autre variable k mais sans la variable j, la valeur du jk-ième élément de la matrice diminuera tandis que les valeurs des ij-ième et ik-ième augmenteront. Une fois que l'apprentissage est terminé, la valeur négative d'un élément, Φ_{ij}, de la matrice indique une relation de dépendance de type co-occurrence entre les variables i et j. Cela signifie qu'elles sont apparues au moins une fois ensemble dans une des données d'apprentissage. À l'inverse, la valeur positive d'un élément, Φ_{ij}, de la matrice indique une relation de indépendance entre les deux variables. Ainsi, nous pouvons interpréter la matrice représentant la fonction de transfert du modèle ILoNDF, Φ_k, comme matrice variable-variable exprimant les relations de co-occurrence entre variables. Cette capacité du modèle ILoNDF lui permet d'identifier les variables corrélées dans la représentation des données d'apprentissage, ce qui rend le modèle plus robuste en présence de variables bruitées ou aberrantes qui pourraient être présentes dans la représentation des données. Pour finir sur ce point, il faut noter que la règle d'apprentissage peut aussi détecter les relations de co-occurrence entre variables. Cependant, une fois qu'une variable devient complètement habituée par le modèle, toutes les informations concernant sa relation avec les autres variables seront perdues. La différence du comportement entre les deux modèles peut se voir plus clairement en comparant les structures internes des matrices Φ^{NDF} et Φ^{ILoNDF}, présentées à travers l'exemple de la section 4.5.

Pour saisir encore mieux comment la modification de la règle d'apprentissage, prescrite par Eq. 4.16, peut pallier les défauts majeurs du modèle NDF, revenons maintenant sur les cas simples présentés dans la section 4.2.

Cas 1 : L'apprentissage du modèle ILoNDF sur le jeu de données #1 aboutit à la formation de la matrice Φ_3 représentant la fonction de transfert du modèle. Cette matrice est :

$$\Phi_3 = 3 \times \begin{pmatrix} 0.556 & -0.192 & -0.099 \\ -0.192 & 0.657 & -0.127 \\ -0.099 & -0.127 & 0.789 \end{pmatrix}$$

Si l'on calcule maintenant les proportions d'habituation des variables divisées par le nombre de données d'apprentissage (cf. Section 4.4, Eq. 4.18), on obtient :

$$H_{v_1} = 0.405 > H_{v_2} = 0.304 > H_{v_3} = 0.195$$

Comme on peut constater, la présence de v_1 dans toutes les données d'apprentissage s'est traduite par une habituation plus importante que celles relatives aux autres variables. De manière

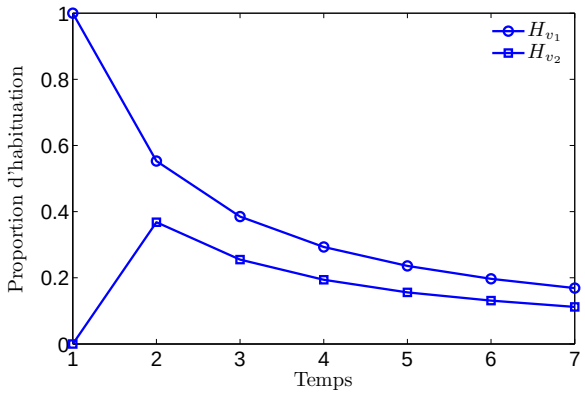


FIG. 4.3 – La capacité d'oubli du modèle ILoNDF. L'apprentissage du modèle ILoNDF est fait à partir du jeu de données #3. Du fait que les variables v_1 et v_2 n'apparaissent plus dans la description des données dès l'étape 3, on voit que l'habituation du modèle ILoNDF au regard de ces variables diminue de façon monotone avec le temps.

générale, on peut en conclure que plus une variable est corrélée aux données d'apprentissage, plus forte est l'importance qui lui sera accordée lors de la modélisation des données d'apprentissage.

Évidemment, l'apprentissage du modèle ILoNDF à partir du jeu de données #2 n'aboutirait pas aux mêmes résultats. Il est facile de vérifier que toutes les variables auront une égalité d'habituation du fait qu'elles ont la même fréquence d'occurrence dans les données d'apprentissage.

Cas 2 : Ce cas sert à montrer comment le modèle ILoNDF peut incorporer de manière implicite quelques effets d'oubli sur les données occasionnelles ou anciennes. La figure 4.3 montre les proportions d'habituation des variables v_1 et v_2 au cours des étapes successives du processus d'apprentissage opéré à partir du jeu de données # 3. À l'opposé du modèle NDF, l'habituation du modèle ILoNDF aux variables v_1 et v_2 tend à diminuer avec le temps.

4.4 Utilisation du modèle ILoNDF pour la classification

L'application du modèle ILoNDF au problème de la classification à partir d'une seule classe, appelée classe positive, est très simple. De fait, l'apprentissage du modèle ILoNDF sur un ensemble d'exemples positifs mène au développement de la fonction de transfert du filtre sous une représentation matricielle, Φ. À chaque fois qu'un exemple est présenté à l'entrée du modèle, il produit un vecteur à la sortie du modèle représentant la nouveauté qu'apporte le vecteur d'entrée après avoir pris en considération la fréquence d'occurrence des variables et leurs relations dans les données d'apprentissage. Après l'apprentissage, les scores de classification de nouvelles données peuvent être obtenus de l'une ou l'autre des manières suivantes : une méthode de projection directe (DPM), une synthèse élémentaire de l'apprentissage (V-PM) et une méthode combinatoire (CS). L'apprentissage du modèle ILoNDF est synthétisé dans l'algorithme 1.

4.4.1 Méthode de projection directe

La manière la plus évidente est de projeter chaque nouvelle donnée, x_i, sur la matrice du filtre Φ pour produire un vecteur de nouveauté $\tilde{x}_i = \Phi \cdot x_i$. Deux proportions peuvent ainsi être

Algorithme 1 Algorithme d'apprentissage du modèle ILoNDF : LearnILoNDF(X, V, Φ_0)

Entrées :

$X = \{x_1, \ldots, x_n\}$ un ensemble de données positives pour l'apprentissage ;

$V = \{v_1, \ldots, v_m\}$ un ensemble de variables utilisées pour la représentation des données d'apprentissage ;

Φ_0 la matrice initiale du modèle ILoNDF qui est, par défaut, une matrice nulle .

Sorties :

Φ_n la matrice finale du modèle ILoNDF ;

P_v le vecteur représentatif de la classe cible ou positive.

début

$P_v^0 = 0$ {un vecteur nul}.

pour chaque $x_k \in X$ **faire**

$$\tilde{x}_k = (I + \Phi_{k-1})x_k$$

$$\Phi_k = I + \Phi_{k-1} - \frac{\tilde{x}_k \tilde{x}_k^T}{\|\tilde{x}_k\|^2}$$

fin pour

pour chaque $v_i \in V$ **faire**

$$H_{v_i} = 1 - \frac{\|\Phi_n \overrightarrow{u_{v_i}}\|}{n \times \|\overrightarrow{u_{v_i}}\|}$$

$$P_v^{(i)} = P_v^{(i-1)} + H_f \overrightarrow{u_v}$$

fin pour

fin

calculées :

– La "*proportion de nouveauté*" qui quantifie la nouveauté qu'apporte une donnée d'entrée au regard des données qui ont été précédemment vues pendant l'apprentissage.

$$N_{x_i} = \frac{\|\tilde{x}_i\|}{n \times \|x_i\|} \tag{4.17}$$

où n est le nombre d'exemples positifs utilisés pour l'apprentissage.

– La "*proportion d'habituation*" qui quantifie la redondance (ou la similarité) que présente une donnée d'entrée par rapport à celles précédemment apprises.

$$H_{x_i} = 1 - \frac{\|\tilde{x}_i\|}{n \times \|x_i\|} \tag{4.18}$$

Et cette dernière pourrait être retenue comme "*score de classification*" pour la donnée x_i indiquant la probabilité que cette donnée appartienne à la classe positive : plus la proportion d'habituation est importante, plus la probabilité d'appartenance d'une donnée à la classe est forte.

4.4.2 Synthèse élémentaire de l'apprentissage

Comme alternative à la méthode précédente, il est également possible de créer à partir de la représentation matricielle du filtre, Φ, un vecteur représentatif des exemples positifs utilisés

pour l'apprentissage, P_v. Le *score de classification* des nouvelles données est alors calculé par comparaison à ce vecteur comme expliqué ci-dessous.

La projection du vecteur unité $\overrightarrow{u_v}$ — associé à la direction d'une variable v dans l'espace de représentation des données — sur la matrice du filtre Φ peut être utilisée pour le calcul de la proportion d'habituation de cette variable comme suit :

$$H_v = 1 - \frac{\|\Phi\overrightarrow{u_v}\|}{n \times \|\overrightarrow{u_v}\|} \tag{4.19}$$

Le vecteur représentatif des exemples positifs s'exprime de manière unique comme combinaison linéaire des vecteurs unités liés aux variables de l'espace de représentation des données $\mathcal{V}$, sous la forme suivante :

$$P_v = \sum_{v \in \mathcal{V}} H_f \overrightarrow{u_f} \tag{4.20}$$

En d'autres termes, le vecteur P_v est défini comme vecteur de poids dont les composantes correspondent aux proportions d'habituation des variables de l'espace de représentation des données, $\mathcal{F}$. Enfin, un score de classification peut être calculé pour chaque donnée x_i en utilisant la similarité cosinus comme suit :

$$Cos(x_i, P_v) = \frac{x_i \cdot P_v}{\|x_i\| \ \|P_v\|} \tag{4.21}$$

où le symbole $\cdot$ désigne le produit scalaire de deux vecteurs.

4.4.3 Méthode combinatoire

Bien évidemment, les deux méthodes mentionnées ci-dessus se comportent différemment et peuvent avoir un impact significativement différent sur les résultats de classification. De manière générale, DPM est une procédure spécifique de *mise en correspondance entre données* qui détermine la nouveauté qu'apporte une donnée d'entrée au regard des données utilisées pour l'apprentissage. Dans le cas du modèle NDF, cette procédure de mise en correspondance est établie entre la donnée d'entrée et la donnée la plus proche parmi toutes les données d'apprentissage. En revanche, le modèle ILoNDF considère les relations de co-occurrence entre variables dans toutes les données d'apprentissage lors de la mise en correspondance entre données (nous expliciterons ce point dans l'exemple de la section suivante). Pour sa part, V-PM est plutôt une procédure de *mise en correspondance entre variables* qui tire profit de la diversité des proportions d'habituation des variables apprises pour faire la distinction entre les données positives et négatives. De ce fait, cette méthode favorise les données contenant un nombre important de variables dont la proportion d'habituation est forte. Dans nos recherches, nous avons trouvé que la méthode V-PM semble être plus précise que DPM, en particulier, lorsque peu de données sont disponibles pour faire l'apprentissage (Kassab et Lamirel, 2006, 2007). Or, si les variables ont approximativement une égalité d'habituation (ce qui peut être en raison d'une mauvaise représentation des données d'apprentissage, comme, par exemple, une pauvre sélection des variables ou un nombre insuffisant de variables), la qualité du vecteur représentatif des données d'apprentissage serait très pauvre. Dans un tel cas, la variance des valeurs des composantes du vecteur serait très faible, dès lors cette variance peut être utilisée comme indicateur de son utilité pour la classification.

La méthode combinatoire (CS) est donc une solution de compromis entre les deux méthodes précitées, qui représente une combinaison pondérée des scores de classification en un score de

classification global (CS). Elle s'exprime comme suit :

$$\mathrm{CS}(x_i) = (1-\lambda)H_{x_i} + \lambda\, Cos(x_i, P_v) \tag{4.22}$$

avec

$$\lambda = \frac{\text{écart type des composantes du vecteur } P_v}{\text{valeur max} - \text{valeur min des composantes du vecteur } P_v}$$

4.5 Illustration

Pour plus de clarté, nous présentons dans cette section un exemple mettant en évidence la différence de fonctionnement entre le modèle NDF et sa version modifié ILoNDF dans le cadre précis de la classification. Dans cet exemple, les trois premières données d_1, d_2 et d_3 du jeu de données #4 présenté ci-dessous, sont choisies pour faire l'apprentissage des modèles NDF et ILoNDF.

Jeu de données #4

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8
v_1	1	0	1	0	1	0	0	1
v_2	1	1	1	0	1	1	0	1
v_3	1	1	0	1	0	0	1	1
v_4	1	1	1	0	0	0	0	1
v_5	0	0	1	0	0	1	1	1

À l'issue de l'apprentissage, la fonction de transfert associée au modèle NDF est :

$$\Phi^{\mathrm{NDF}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & -0.2 & -0.4 & -0.2 \\ 0 & -0.2 & 0.4 & -0.2 & 0.4 \\ 0 & -0.4 & -0.2 & 0.6 & -0.2 \\ 0 & -0.2 & 0.4 & -0.2 & 0.4 \end{pmatrix}$$

et la fonction de transfert associée au modèle ILoNDF est :

$$\Phi^{\mathrm{ILoNDF}} = 3\times \begin{pmatrix} 0.792 & -0.089 & 0.018 & -0.089 & -0.107 \\ -0.089 & 0.769 & -0.152 & -0.231 & -0.079 \\ 0.018 & -0.152 & 0.798 & -0.152 & 0.050 \\ -0.089 & -0.231 & -0.152 & 0.769 & -0.079 \\ -0.107 & -0.079 & 0.050 & -0.079 & 0.871 \end{pmatrix}$$

Sur la base des matrices ci-dessus, les vecteurs représentatifs des données d'apprentissage liés aux modèles NDF et ILoNDF peuvent être construits en utilisant Eq. 4.20. Ils sont respectivement donnés par :

$$P_v^{\mathrm{NDF}} = \begin{pmatrix} 1 & 0.225 & 0.368 & 0.225 & 0.368 \end{pmatrix}^T$$
$$P_v^{\mathrm{ILoNDF}} = \begin{pmatrix} 0.191 & 0.174 & 0.172 & 0.174 & 0.114 \end{pmatrix}^T$$

Rappelons que les valeurs des composantes des vecteurs représentatifs des données d'apprentissage correspondent aux proportions d'habituation des variables de l'espace de représentation

TAB. 4.1 – Ordonnancement et scores de classification du jeu de données #4 calculés selon les méthodes DPM, V-PM et CS pour chacun des modèles NDF et ILoNDF.

	DPM		V-PM		CS	
	Doc.	Score	Doc.	Score	Doc.	Score
NDF	d_1, d_2, d_3	1	d_8	0.835	d_1, d_3	0.907
	d_8	0.717	d_1, d_3	0.776	d_8	0.766
	d_5, d_6	0.452	d_5	0.740	d_2	0.752
	d_4	0.368	d_7	0.444	d_5	0.572
	d_7	0.106	d_2	0.403	d_6	0.413
			d_6	0.358	d_4	0.345
			d_4	0.314	d_7	0.246
ILoNDF	d_2	0.560	d_8	0.987	d_8	0.704
	d_1	0.530	d_1	0.952	d_1	0.692
	d_8	0.527	d_3	0.874	d_2	0.653
	d_3	0.511	d_2	0.804	d_3	0.650
	d_5	0.255	d_5	0.691	d_5	0.422
	d_6	0.209	d_6	0.544	d_6	0.338
	d_4	0.172	d_7	0.541	d_4	0.282
	d_7	0.083	d_4	0.460	d_7	0.258

des données. D'un côté, on peut constater à partir de la représentation du vecteur P_v^{NDF}, que la variable v_1 est beaucoup plus habituée que les autres et en conséquence, elle aura donc un poids plus important dans la modélisation des données d'apprentissage. Par ailleurs, la variable v_5 qui apparaît moins fréquemment est plus habituée que d'autres variables plus fréquentes telles que v_4. D'un autre côté, l'application de la règle d'apprentissage du modèle ILoNDF (Eq. 4.16) réduit assez la différence entre la proportion d'habituation associée à v_1 et les autres variables (v_2, v_3 et v_4) tandis qu'elle fait rétrograder l'habituation de v_5.

Nous pouvons maintenant calculer les scores de classification du jeu de données #4 en utilisant les méthodes DPM, V-PM et CS, et puis, les ordonner en fonction de leur similarité aux données utilisées pour l'apprentissage. Le tableau 4.1 montre le classement des données par ordre de similarité décroissant selon les différentes méthodes précitées.

Il est clair à partir du tableau 4.1, que seule la méthode de projection directe des données permet au modèle NDF d'aboutir à une habituation totale vis-à-vis des données d'apprentissage (d_1, d_2, d_3). Ce résultat est parfaitement compatible avec la conception du modèle NDF et la stratégie de mise en correspondance au niveau des données de la méthode DPM. Il est aussi possible de constater que le modèle NDF n'est pas capable de distinguer nettement entre les variables corrélées à la description des données d'apprentissage (variables pertinentes) et celles qui ne le sont pas (variables non pertinentes). Cette dernière situation est particulièrement claire dans le cas des données d_5 et d_6 auxquelles a été attribué le même score de classification, malgré le fait que d_5 est plus similaire aux données utilisées pour faire l'apprentissage.

De son côté, le modèle ILoNDF couplé avec la méthode de projection directe n'atteint pas

une valeur maximale d'habituation vis-à-vis des données d'apprentissage, et ne leur attribue pas non plus un score de classification équivalent. La raison en est que la mise en correspondance est effectuée en comparant la donnée d'entrée en cours à toutes les données d'apprentissage, et pas seulement à la donnée la plus proche, comme le fait le modèle NDF. Cette démarche est particulièrement utile pour établir la distinction entre les variables pertinentes et non pertinentes, et en conséquence, pour la distinction entre les données pertinentes et non pertinentes. Par exemple, il ressort du tableau 4.1 que le modèle ILoNDF a assigné à d_6 un score de classification inférieur à celui qu'il a assigné à d_5.

En opposant maintenant les méthodes DPM, V-PM et CS l'une à l'autre, on constate dans le cas du modèle NDF que V-PM pourrait résoudre certains des problèmes de discrimination inhérents à la stratégie de la méthode DPM. Précisons, par exemple, que d_5 est considérée, en utilisant la méthode V-PM, comme étant plus proche des données d'apprentissage que d_6. Or, d'autres problèmes de discrimination sont apparus. Par exemple, d_2 a un score de classification inférieur à ceux de d_5 et d_7. Ce problème provient du fait que la qualité du vecteur représentatif des données d'apprentissage relative au modèle NDF est très médiocre. Pour sa part, la méthode de combinaison (CS) est réalisée avec $\lambda = 0.416$, ce qui favorise légèrement DPM par rapport à V-PM. Généralement, la méthode CS s'avère plus précise mais les résultats obtenus restent encore insatisfaisants.

En ce qui concerne le modèle ILoNDF, l'application de la méthode V-PM a changé le classement des quatre premières données (d_1, d_2, d_3 et d_8) de sorte que plus le nombre de variables pertinentes présentes dans une des données est grand, plus le score de classification est fort ; or, le changement le plus important concerne les données d_4 et d_7. Notons que si la variable v_5 était une variable non pertinente liée au bruit dans la représentation des données d'apprentissage, il vaudrait mieux privilégier la méthode DPM, autrement, la méthode V-PM sera préférable. Du fait que la variance du vecteur représentatif des données d'apprentissage associé au modèle ILoNDF est faible, la méthode de combinaison favorisera DPM avec un facteur $\lambda = 0.383$ afin d'empêcher des problèmes probables.

4.6 Méthode de seuillage

Jusqu'à ce point, les scores de classification (cf. Eq. 4.18, Eq. 4.21 et Eq. 4.22) ont seulement été utilisés pour le classement des données par ordre de leur pertinence pour ce qui concerne la classe positive. La séparation entre les données positives ou négatives peut être réalisée par seuillage des scores de classification : Toutes les données dont les scores de classification sont supérieurs (resp. inférieurs) à un seuil déterminé, sont classifiées en tant que positives (resp. négatives). La valeur du seuil influence donc de façon très considérable les résultats de classification.

Comme avec beaucoup d'algorithmes d'apprentissage, le problème de déterminer automatiquement un seuil approprié entre deux classes en utilisant des donnes issues d'une seule classe est non trivial. En effet, certains algorithmes adaptés à la tâche de classification se contentent de fournir simplement une liste de données ordonnées par rapport à leur pertinence, sans prendre une décision binaire quant à l'appartenance ou non d'une donnée à une des classes, comme cela a été fait dans (Žižka et al., 2006). D'autres comportent un seuil à définir par l'utilisateur : une pratique courante est de fournir la fraction de données qui devrait être admise dans la classe cible (Schölkopf et al., 2001). Pour autant, la définition d'une telle fraction n'est pas triviale. En

Algorithme 2 Algorithme de détermination automatique du seuil : Seuillage(X, T)

Entrées :

$X = \{x_1, \ldots, x_n\}$ un ensemble de données positives pour l'apprentissage ;
$F = \{f_1, \ldots, f_m\}$ un ensemble de variables utilisées pour la représentation des données d'apprentissage.

Sorties :

s la valeur seuil.

début

Initialisation :
Φ_0=0 ; $subLX_0$={} ; $subNLX_0 = X$.
s=0, nbEtapes=0 ;
pour i=0 to n **faire**
si $((i \ \% \ \frac{n}{10}) \ != 0)$ **alors**
$subLX_i = subX_{i-1} \cup x_i$
$subNLX_i = subNLX_{i-1} \setminus x_i$
sinon
$(\Phi_i, P_i) = LearnILoNDF(subLX_i, T, \Phi_{i-1})$
$s_1 = MCScores(subLX_i, P_i)$
$s_2 = MCScores(subNLX_i, P_i)$
$s = s + (|subLX_i| \times s_1 + |subNLX_i| \times s_2)/n$
nbEtapes=nbEtapes+1
fin si
fin pour
$s = \tau \dfrac{s}{nbEtapes}$

fin

Note :

MCScores(X, P) renvoie la moyenne des scores de classification des données dans l'ensemble X calculés en utilisant une des formules Eq. 4.18, Eq. 4.21 or Eq. 4.22.

ce qui concerne le modèle ILoNDF, nous proposons une stratégie de seuillage simple qui peut être appliquée à toute approche d'apprentissage d'un fonctionnement en ligne. En développant notre stratégie de seuillage, nous nous fondons sur les réflexions suivantes :

1. Les scores de classification attribués aux données après leur utilisation pour l'apprentissage peuvent être utilisés comme un bon indicateur des scores de données qui peuvent être positives et qui sont faciles à détecter du fait qu'elles sont fortement semblables aux données utilisées pour l'apprentissage du modèle de la classe positive. Par conséquent, la moyenne de ces scores peut être admise comme une borne supérieure pour le seuil de classification.
2. Les scores de classification attribués aux données disponibles pour l'apprentissage avant qu'elles ne soient apprises peuvent être utilisés comme un bon indicateur des scores de données qui sont positives mais qui sont moins faciles à détecter du fait qu'elles ne sont pas fortement semblables aux données utilisées pour l'apprentissage du modèle de la classe positive. Par conséquent, la moyenne de ces scores peut être admise comme une borne inférieure pour le seuil de classification.

Ainsi, à chaque étape de l'apprentissage, nous calculons une valeur combinant les deux valeurs

correspondant aux deux bornes extrêmes en considérant la densité (quantité) des données qui sont déjà apprises et celle des données qui ne le sont pas encore, puisque la fiabilité des scores de classification calculés au cours de l'apprentissage dépend de la qualité du modèle appris et ainsi du nombre de données utilisées pour effectuer l'apprentissage de ce modèle. Le seuil final est un pourcentage prédéterminé τ de la moyenne des valeurs obtenues à toutes les étapes de l'apprentissage. L'algorithme 2 présente les détails du calcul de la valeur du seuil.

4.7 Méthodes typiques pour la classification à partir d'une seule classe

Le but des sections précédentes était de décrire les aspects théoriques et l'utilisation du modèle ILoNDF dans le cadre de la classification à partir d'une seule classe. L'intérêt fondamental du modèle ILoNDF au regard des méthodes actuelles est qu'il permet un fonctionnement en ligne et en un seul passage sur les données, ce qui le rend particulièrement utile pour l'analyse des flux de données. Mais aussitôt une question s'impose naturellement à ce stade concernant l'apport du modèle du point de vue de la qualité de la classification, comparativement aux autres méthodes existantes. C'est donc dans le but de répondre à cette question que nous aborderons dans cette section une revue de détail des quatre méthodes couramment utilisées pour résoudre le problème de la classification à partir d'une seule classe. Il s'agit des méthodes basées sur l'analyse en composantes principales (ACP), les réseaux de neurones auto-associatifs et les SVM monoclasses. L'étude comparative entre notre modèle et les méthodes précitées, ainsi que les résultats expérimentaux, seront présentés et discutés dans la section qui suit.

4.7.1 Modèles statistiques multivariés

Comme nous l'avons indiqué précédemment dans le chapitre 1, l'analyse en composantes principales (ACP) est une des techniques de base de l'analyse multivariée des données (Jolliffe, 1986). Il s'agit d'une transformation linéaire de l'ensemble des données de l'espace originel dont les variables sont corrélées, vers un espace orthogonal où tous les axes, qui sont des combinaisons linéaires des variables d'origine, sont non corrélés deux à deux. Dans ce nouvel espace, les axes sont construits de telle sorte que le premier axe explique la variance maximale des données, et le deuxième axe, orthogonal au premier, explique la partie la plus grande de la variance résiduelle, et ainsi de suit. Les k premiers axes comportent donc l'essentiel de la variance des données, et c'est à ce titre qu'ils sont dits "*composantes principales*" ; l'élimination des axes où la variance des données est minimale permet alors une réduction de la dimensionnalité avec une perte minimale d'informations.

En termes mathématiques, l'ACP se ramène à calculer les valeurs propres et les vecteurs propres de la matrice de covariance des données : Les vecteurs propres de la matrice donnent les directions des composantes principales de l'ACP, et les valeurs propres associées représentent la variance expliquée par les composantes. La détermination du nombre optimal de composantes principales à retenir pour représenter les données dans le nouvel espace du modèle ACP peut s'effectuer de différentes manières (Valle et al., 1999). Une manière particulièrement simple consiste à fixer un seuil en termes de variance totale cumulée par les k premières composantes successives, permettant d'affirmer que l'on représente, par exemple, 90% de la variance totale. Une autre manière, adoptée dans nos expérimentations, s'en tient au critère de la valeur propre moyenne. Elle consiste à retenir toutes les composantes dont les valeurs propres sont supérieures à la valeur

propre moyenne et à éliminer celles dont les valeurs propres sont au-dessous de la moyenne. En pratique, le nombre de composantes retenues dépend fortement des données analysées et paraît être beaucoup plus petit que le nombre de variables dans l'espace original.

Dans les deux paragraphes qui suivent nous décrivons deux méthodes statistiques multivariées se fondant sur l'utilisation de l'ACP en mettant l'accent sur leur application à la classification à partir d'une seule classe.

Les résidus de l'ACP

L'ACP peut être vue comme une décomposition de l'espace des données positives en deux sous-espaces orthogonaux : le sous-espace factoriel (le modèle de l'ACP) et le sous-espace résiduel (Les résidus de l'ACP). Soit $X \in \Re^{n \times p}$ un ensemble de n exemples positifs d'apprentissage représentés dans un espace de p variables. Après l'application de la transformation de l'ACP, le sous-espace représentatif du modèle de l'ACP — qui est engendré par les k première composantes principales (P_k) — est associé aux variations systématiques des données. En revanche, le sous-espace résiduel — qui est engendré par les résidus de l'ACP , i.e. les n-k composantes principales — est associé aux variations aléatoires dues aux erreurs ou au bruit dans les données. Par conséquent, toute donnée x_i peut être décomposée sous la forme :

$$x_i = \hat{x}_i + \tilde{x}_i$$

où $\hat{x} = P_k P_k^T x$ et $\tilde{x} = (I - P_k P_k^T)x$ représentent respectivement les projections de x_i sur le sous-espace modèle et le sous-espace résiduel. La correspondance des données à la classe positive peut alors être évaluée par l'erreur carrée de prévision du résiduel (SPE), également appelée l'index de Q-statistique, et définie comme :

$$\text{SPE} = \|\tilde{x}_i\|^2 = x_i^T \left(I - P_k P_k^T\right) x_i$$

Les données sont jugées négatives lorsque SPE dépasse un seuil prédéfini adapté au type de la Q-statistique (Jackson et Mudholkar, 1979).

Le test T^2 de Hotelling

Tandis que l'analyse des résidus offre une manière d'identifier les données qui débordent de l'espace du modèle de l'ACP, le test T^2 de Hotelling fournit une indication de la variabilité inhérente aux données par rapport à l'espace du modèle de l'ACP. En effet, après la mise en place d'un modèle ACP à partir des données positives d'apprentissage, le test T^2 de Hotelling peut être calculé sur la base des k premières composantes principales du modèle (Kim et Beale, 2002; Detroja et al., 2007) comme suit. Soit une donnée x_i représenté par un vecteur centré par la moyenne des données positives d'apprentissage, la statistique T^2 correspondant à cette donnée est :

$$T_i^2 = t_i^T \Lambda^{-1} t_i = x_i^T P_k \Lambda^{-1} P_k^T x_i$$

où $t_i = P_k^T x_i$ est la projection orthogonale de x_i sur le sous-espace du modèle ACP défini par les k premières composantes principales, et Λ est une matrice diagonale contenant les k premières valeurs propres de la matrice de covariance des données positives d'apprentissage.

Une grande valeur de T^2 indique une grande déviation des données d'apprentissage de la classe positive. Un seuil (T_α^2) peut être obtenu en utilisant la distribution F (Detroja et al., 2007).

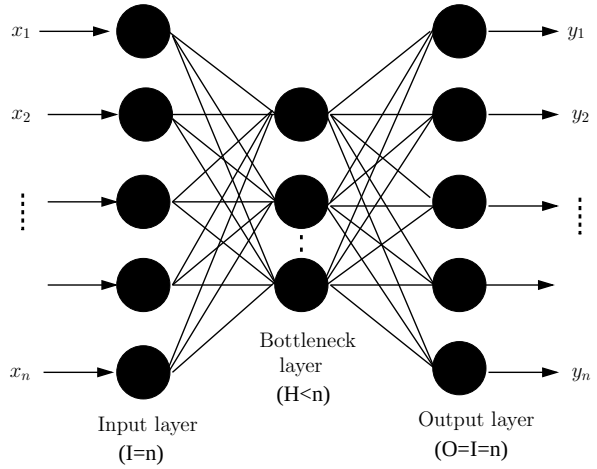


FIG. 4.4 – Une représentation schématique d'un réseau de neurones auto-associatif de type MLP à une seule couche cachée constituant un goulet d'étranglement dans le réseau.

4.7.2 Réseaux de neurones auto-associatifs de type MLP

Les réseaux de neurones auto-associatifs (AANNs), également connus sous le nom d'auto-encodeurs, sont un type spécial de réseaux multicouches de type feedforward. Ces réseaux, introduits à l'origine par Rumelhart et al. (1986), sont entraînés pour produire une approximation de la fonction identité entre les entrées et les sorties du réseau. Sous sa forme conventionnelle, un réseau de neurones auto-associatif consiste en une couche de neurones d'entrée, suivie d'une ou plusieurs couches cachées, et une couche de neurones de sortie dont la dimension est égale à celle de la couche d'entrée (cf. Figure 4.4). La dimension d'une des couches cachées — la couche centrale — devrait être plus petite que celles des couches d'entrée ou de sortie (ou encore des autres couches cachées si le réseau en comporte plusieurs). Cette restriction du nombre de neurones est imposée en vue de capturer les variables significatives (similaires aux composantes principales) dans la représentation des données d'apprentissage en comprimant leurs redondances sans chercher à mémoriser les données [29,30].

La mise en place d'un AANN pour résoudre le problème de la classification à partir d'une seule classe se fonde sur deux processus : l'apprentissage des poids des connexions du réseau et la détermination d'une fonction de seuillage. En phase d'apprentissage, on présente les exemples positifs en entrée, et on adapte les poids des connexions du réseau de telle sorte que les sor-

[29]Typiquement, les AANNs sont conçus sous la restriction que la dimension de la couche cachée soit inférieure à celle de la couche d'entrée (ou de sortie). La restriction du nombre de neurones sur la couche cachée présente un intérêt particulier pour l'extraction de variables et la compression de données. Or, la conception d'un AANN tel que la dimension de la couche cachée soit plus élevée que celle de la couche d'entrée — ce qui est manifestement faisable (Japkowicz et al., 2000) — peut également être exercée dans le cadre de la classification du fait que l'on ne s'intéresse dans ce cas qu'à la sortie du réseau ; la transformation effectuée au niveau de la couche cachée n'y est pas directement exploitée. Cependant, sur le plan expérimental, nous avons constaté que l'augmentation du nombre de neurones cachés au delà du nombre de neurones d'entrée ne fait qu'accroître la complexité du réseau AANN ; et souvent les résultats sont légèrement inférieurs.

[30]La transformation (entrée-sortie) peut être linéaire ou non linéaire suivant l'architecture du réseau et le type de fonctions de transfert de la couche de sortie. Lorsque la fonction de transfert est linéaire, l'AANN réalise une analyse en composantes principales. Lorsque des fonctions non linéaires sont utilisées, AANN pourrait résoudre des problèmes que l'analyse en composantes principales ne parvient pas à résoudre (Japkowicz et al., 2000; Cottrell et Munro, 1988).

ties répliquent les entrées. Les entrées sont donc les sorties désirées. L'adaptation des poids des connexions se fait le plus souvent à l'aide de l'algorithme de la rétropropagation dont le principe consiste à minimiser l'erreur quadratique (MSE) entre les sorties désirées (X) et les sorties observées (Y) en utilisant une descente de gradient (Baldi et al., 1995) :

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} \|X - Y\|^2$$

où N représente le nombre d'exemples positifs utilisés pour l'apprentissage du réseau.

Après l'apprentissage, la classification peut être effectuée en partant du principe que les données positives seront reconstruites de façon quasi exacte alors que les données négatives ne le seront pas. Seulement, la définition d'une frontière exacte (seuil de décision) entre les données positives et négatives est assez problématique. Lors de nos expériences préliminaires, nous avons repris la démarche de Japkowicz et al. (1995), qui consiste à considérer une borne supérieure sur l'erreur de reconstruction des données positives à chaque époque de l'apprentissage du réseau et à relaxer ensuite cette borne par un certain pourcentage (par exemple 25 %). Les données sont alors classées en comparant leur erreur de reconstruction à la frontière relaxée. Si l'erreur relative à une donnée dépasse la frontière dans au moins une certaine proportion des époques considérées, celle-ci sera classée négative, autrement elle sera classée positive. Selon ce que nous avons pu constater la précision des résultats obtenus avec cette méthode est extrêmement basse, ainsi, nous avons développé une stratégie légèrement différente consistant à calculer la moyenne des erreurs de reconstruction des données d'apprentissage à chacune des k premières époques (dans nos expériences, k est égale à deux fois l'indice de la meilleure époque correspondant à l'erreur de validation la plus basse). La moyenne de ces k valeurs moyennes obtenues est utilisée comme seuil de décision.

Pour les expériences qui seront rapportées plus tard dans ce chapitre, nous utilisons un réseau auto-associatif à trois couches utilisant un algorithme standard de rétropropagation. Le nombre de neurones des couches entrée/sortie (I=O) est déterminé par la dimensionnalité des données utilisées pour l'apprentissage. En ce qui concerne le nombre de neurones cachées (H), nous expérimentons différentes valeurs sous forme des pourcentages mettant en relation le nombre de neurones cachées et celui de neurones d'entrée ($\frac{H}{I}$: 10%, 25%, 50% et 75%). Le taux d'apprentissage et le terme de momentum sont respectivement fixés à 0.1 et 0.9.

4.7.3 Les SVM monoclasses

Une version des machines à vecteurs supports adaptée à la classification à partir d'une seule classe (1-SVM) a été proposée par Schölkopf et al. (2001). Son idée est de séparer les exemples positifs de l'origine, le seul exemple négatif, avec une marge maximale dans l'espace de redescription. En termes plus précis, 1-SVM cherche une hypersphère de volume minimal qui englobe la plupart des exemples positifs disponibles pour l'apprentissage (cf. Figure 4.5). L'algorithme peut être brièvement décrit comme suit.

Soit un ensemble de données d'apprentissage comprenant n exemples positifs $\{x_1, \cdots, x_l\}$, $x_i \in \Re^n$ et soit $\Phi : \mathcal{X} \rightarrow \mathcal{F}$ une transformation des données de l'espace d'origine $\mathcal{X}$ vers l'espace de redescription $\mathcal{F}$. Le calcul de la solution de 1-SVM consiste à résoudre le problème primal suivant :

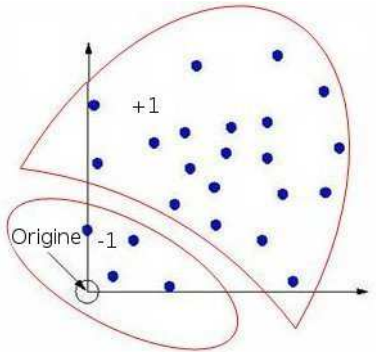


FIG. 4.5 – Interprétation géométrique de la classification par 1-SVM dans un espace à deux dimensions

$$\min_{w,\xi,\rho} \qquad \frac{1}{2}w.w + \frac{1}{\nu l}\sum_{i=1}^{l}\xi_i - \rho \tag{4.23}$$

$$avec\ (w.\Phi(x_i)) \geq \rho - \xi_i,\ \xi_i \geq 0,\ \ i = 1, \cdots, l$$

où ξ_i sont les variables de relâchement et $\nu \in [0,1]$ est un paramètre qui permet de quantifier l'impact des variables de relâchement, i.e. la fraction des données d'apprentissage qui peuvent se situer en dehors de l'hypersphère (Pour $\nu = 0$, toutes les données d'apprentissage sont situées à l'intérieur de l'hypersphère ; et dans ce cas 1-SVM peut être qualifié de marge dure).

En introduisant les multiplicateurs de Lagrange, et en appliquant les conditions de Karush-Kuhn-Tucker, on obtient :

$$w = \sum_i \alpha_i \Phi(x_i) \tag{4.24}$$

Ce qui conduit à la formulation duale du problème sous la forme :

$$\min_{\alpha}\ \frac{1}{2}\sum_{ij}\alpha_i\alpha_j K(x_i,x_j) \tag{4.25}$$

$$avec \quad 0 \leq \alpha_i \leq \frac{1}{\nu l}, \quad \sum_i \alpha_i = 1$$

où $K(x_i,x_j) = \Phi(x_i)\cdot\Phi(x_j)$ est la fonction noyau et les x_i qui interviennent dans la solution ($\alpha_i \neq 0$) sont les vecteurs supports.

Une fois que les valeurs optimales des paramètres sont trouvées, on peut classer les nouvelles données selon la fonction de décision suivante : $f(x) = sgn(\sum_i \alpha_i K(x_i,x) - \rho)$ telle que les données vérifiant $f(x) \geq 0$ se trouvent à l'intérieur de l'hypersphère et sont donc à classer positives. Autrement, elles se trouvent en dehors de l'hypersphère et sont à classer comme négatives.

1-SVM présente les mêmes avantages que SVM, mais, il exige beaucoup plus de données d'apprentissage pour pouvoir définir une frontière de séparation précise. La raison en est que les vecteurs supports sont issus seulement des exemples positifs[31].

[31] Dans nos expériences, nous utilisons une implémentation des SVM, nommée mySVM, par Stefan Rüping (http ://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/).

TAB. 4.2 – Répartition des documents d'apprentissage et de test dans les différentes catégories des corpus Reuters-21578 et WebKB.

Reuters			WebKB		
Category	Train	Test	Category	Train	Test
earn	2877	1087	student	1513	128
acq	1650	719	faculty	1089	34
money-fx	538	179	course	886	44
grain	433	149	project	484	20
crude	389	189	depart	181	1
interest	347	131	staff	116	21
trade	369	117			
wheat	212	71			
ship	197	89			
corn	181	56			

4.8 Expérimentations

Dans cette section, nous présentons des résultats expérimentaux montrant l'efficacité et la robustesse de notre modèle dans le cadre précis de la classification à partir d'une seule classe. Nos expérimentations sont menées sur deux collections de test : Reuters-21578 et WebKB (voir Annexe A pour une description plus complète de ces collections) :

- Le corpus Reuters-21578 est constitué de 21578 documents extraits à partir de dépêches de l'agence de presse Reuters. Les documents sont classés manuellement dans l'une ou plusieurs des 135 catégories sémantiques. Les résultats de nos expérimentations sont rapportés pour les 10 catégories les plus fréquentes du corpus Reuters. Le nombre de documents d'apprentissage associés à ces catégories varie de 181 à 2877 avec une moyenne de 719.3 documents par catégorie (cf. Tableau 4.2).
- Le corpus WebKB contient 8280 documents représentant des pages web recueillies de sites web de départements d'informatique de plusieurs universités américaines, classés en sept catégories. Contrairement au corpus Reuters-21578, chaque document est associé à une seule catégorie. Comme beaucoup d'autres travaux (Yang et al., 2008; Hoi et al., 2006; Mihalcea et Hassan, 2005), nous avons fait le choix d'exclure la catégorie "other" du corpus WebKB à cause de sa définition très générale. Le nombre de documents d'apprentissage varie de 116 à 1090 avec une moyenne de 711.7 documents par catégorie (cf. Tableau 4.2).

Tous les documents utilisés durant les phases d'apprentissage et de test ont subi un prétraitement standard des textes : tokenisation, filtrage des mots vides et lemmatisation. Nous ne retenons que des termes simples comme termes d'indexation. Pour chaque catégorie, les termes d'indexation sont sélectionnés selon la méthode de sélection non supervisée "*Document Frequency*" (DF) de telle sorte que seuls les termes apparaissant le plus fréquemment dans les documents d'apprentissage associés à la catégorie en question sont retenus. Dans le but d'étudier l'influence de la dimensionnalité de l'espace de représentation (i.e. le nombre de termes retenus pour l'indexation des documents) sur la qualité de la classification, nous avons décidé de considérer quatre seuils de fréquence lors de la sélection des termes :

- T10 : Seulement les 10 termes les plus fréquents sont retenus ;
- T20 : Seulement les 20 termes les plus fréquents sont retenus ;

- >10% : Seulement les termes qui apparaissent dans au moins 10% des documents d'apprentissage sont retenus ;
- >5% : Seulement les termes qui apparaissent dans au moins 5% des documents d'apprentissage sont retenus ;

Les documents d'apprentissage et de test sont ensuite représentés sous forme de vecteurs de termes. La pondération des termes est aussi faite selon quatre différents schémas (cf. Section 3.1.2) :

- NTF : La fréquence d'occurrence des termes dans le document normalisée par le cosinus ;
- logNTF : Une légère variation de NTF considère le logarithme de la fréquence des termes $\log_2(\text{TF}+1)$ dans le but d'atténuer les effets de larges différences entre les fréquences d'occurrence des termes ;
- ANTF : Une pondération à fréquence augmentée $(0.5 + 0.5(\frac{TF}{maxTF}))$, suivie par une normalisation de type cosinus ;
- Binary : Une pondération binaire des termes.

Il est important d'insister sur le fait que les processus de la sélection et de la pondération des termes sont effectués de manière totalement indépendante pour chacune des catégories. C'est parce que seulement les informations liées à la classe positive — représentée par l'une des catégories — sont censées être disponibles dans le cadre de la classification à partir d'une classe, la classe positive. Ainsi, différents ensembles de termes sont sélectionnés pour chacune des différentes catégories à partir des documents d'apprentissage qui y sont associés uniquement. Les documents d'apprentissage associés à chaque catégorie et tous les documents de test ont alors été représentés en utilisant l'ensemble de termes spécifiques à la catégorie en considération.

L'évaluation de la qualité de la classification a été faite en utilisant les critères classiques basés sur la précision et le rappel décrits dans la section 1.5.1. En premier lieu, nous mesurons la qualité de l'ordonnancement des données produit par les différentes méthodes de la classification. Cela permet d'évaluer la qualité de l'apprentissage proprement dit des méthodes indépendamment du seuil de décision. En outre, certaines applications exigent que les données soient ordonnées par pertinence au lieu — ou en plus — d'être simplement classées comme positives ou négatives (c'est par exemple le cas dans des applications médicales). À cet effet, la performance est mesurée à l'aide de courbes Rappel-Précision (R-P) et de la précision moyenne non-interpolée (MAP). En second lieu, nous considérons des mesures intégrant l'effet du choix des seuils de décision, à savoir la mesure F_1, dans le but d'évaluer les différentes stratégies de seuillage et la performance globale des méthodes de classification en conditions réalistes. Les mesures de moyennes de type macro-moyenne et micro-moyenne sont utilisées pour évaluer la performance sur plusieurs catégories.

La première série d'expérimentations que nous avons menées consiste à évaluer la robustesse des méthodes de classification vis-à-vis des différents aspects liés aux conditions initiales (dimensionnalité de l'espace de représentation et schémas de pondération) et au réglage de paramètres. Dans le seconde série d'expérimentations, nous nous concentrons sur la comparaison du modèle ILoNDF avec les meilleures méthodes de classification identifiées dans la première série d'expérimentations. Avant de poursuivre, il convient de préciser que les résultats que nous allons présenter ici sont obtenus avec une incertitude de moins de ±2% sur le corpus Reuters et de moins de ±5% sur le corpus de WebKB, pour un intervalle de confiance à 95%.

Tab. 4.3 – Moyenne des scores MAP correspondant au modèle NDF sur les corpus Reuters et WebKB obtenus selon les différents schémas de pondération et les différentes dimensionnalités de l'espace de représentation des documents. Les entrées en gras indiquent les scores les plus élevés statistiquement, alors que les entrées soulignées indiquent les scores correspondant aux conditions référentielles retenues selon les résultats globaux du modèle NDF

		Reuters				WebKB			
		T10	T20	>10%	>5%	T10	T20	>10%	>5%
DPM	NTF	0.1634	0.1861	0.2105	0.2891	0.2382	0.23	0.1937	0.2650
	logNTF	0.2008	0.2088	0.2023	0.3312	0.2258	0.2154	0.2066	0.3681
	ANTF	0.2561	0.2383	0.2110	0.3277	0.2073	0.2168	0.1941	0.2792
	binary	0.1850	0.2112	0.2015	0.3117	0.2391	0.2165	0.1798	0.2646
V-PM	NTF	0.4798	0.4833	0.3646	0.3222	0.3238	0.3135	0.2809	0.4038
	logNTF	0.5064	0.4781	0.3233	0.2710	0.4010	0.4802	0.32	0.3264
	ANTF	0.5012	0.4636	0.3109	0.2556	**0.5261**	0.4859	0.3031	0.2831
	binary	0.4791	0.4490	0.3101	0.2525	0.5244	0.3972	0.3291	0.2770
CS	NTF	0.4811	0.4838	0.3645	0.3556	0.3234	0.3136	0.2802	0.2891
	logNTF	**0.5068**	0.4779	0.3233	0.3529	0.4009	0.4804	0.3516	0.4055
	ANTF	0.5006	0.4635	0.3110	0.3512	0.5257	0.4863	0.2897	0.4075
	binary	0.4770	0.4532	0.3099	0.3302	0.5208	0.4091	0.2516	0.4115

4.8.1 Résultats pour différentes conditions expérimentales

NDF vs. ILoNDF

Notre première expérimentation consiste à comparer le comportement du modèle NDF avec celui du modèle ILoNDF. Les tableaux 4.3 et 4.4 résument les scores MAP obtenus selon les différents schémas de pondération et les différentes dimensionnalités de l'espace de représentation des documents. Les figures 4.6 et 4.8 montrent les courbes Rappel-Précision correspondant aux modèles pour les différentes dimensionnalités de l'espace de représentation. Les figures 4.7 et 4.9 montrent les courbes Rappel-Précision correspondant aux modèles pour les différents schémas de pondération. Les résultats sont rapportés pour les différentes méthodes de calcul des scores de classification des modèles, à savoir la méthode de projection directe (DPM), la méthode de synthèse élémentaire de l'apprentissage (V-PM), et la méthode combinatoire (CS), sur les corpus de test Reuters et WebKB.

D'après les résultats obtenus, nous constatons que la méthode DPM n'est pas la stratégie la plus appropriée pour le modèle NDF dans le contexte de la classification. La méthode V-PM apparaît sensiblement meilleure que DPM. Une légère amélioration peut encore être attendue en appliquant la méthode combinatoire (CS), cela vaut en particulier pour le cas de la forte dimensionnalité de l'espace de représentation des documents (>5%). Dans le cas du modèle ILoNDF, la différence entre DPM et V-PM est relativement minime. La méthode V-PM donne des résultats sensiblement meilleurs que DPM sur le corpus Reuters, mais des résultats moins satisfaisants sur le corpus WebKB. L'explication tient au fait que la méthode V-PM favorise les documents contenant une forte proportion de termes apparaissant dans les documents utilisés pour faire l'apprentissage. De ce fait, si les termes utilisés pour la représentation des documents sont mal

TAB. 4.4 – Moyenne des scores MAP correspondant au modèle ILoNDF sur les corpus Reuters et WebKB obtenus selon les différents schémas de pondération et les différentes dimensionnalités de l'espace de représentation des documents. Les entrées en gras indiquent les scores les plus élevés statistiquement, alors que les entrées soulignées indiquent les scores correspondant aux conditions référentielles retenues selon les résultats globaux du modèle ILoNDF

		Reuters				WebKB			
		T10	T20	>10%	>5%	T10	T20	>10%	>5%
DPM	NTF	0.4390	0.4448	0.4770	0.5254	0.2268	0.2428	0.3378	0.3443
	logNTF	0.5130	0.5226	0.6001	0.6699	0.2882	0.3645	0.4814	0.5038
	ANTF	0.5584	0.5560	0.6236	0.6857	0.3592	0.4796	0.5094	0.5102
	binary	0.5755	0.5494	0.6044	0.6496	0.5221	0.5077	0.5117	0.5014
V-PM	NTF	0.5322	0.5294	0.5308	0.5492	0.2495	0.3262	0.3702	0.3901
	logNTF	0.6161	0.6267	0.6540	0.6748	0.3720	0.4414	0.3886	0.4903
	ANTF	0.6125	0.6434	0.6773	0.6937	0.4964	0.3836	0.4865	0.4969
	binary	0.6133	0.6189	0.6451	0.6556	0.5167	0.4040	0.4929	0.4901
CS	NTF	0.4903	0.4883	0.5032	0.5376	0.2466	0.2667	0.3528	0.3568
	logNTF	0.5765	0.5820	0.6318	0.6762	0.3168	0.4512	0.4813	0.5067
	ANTF	0.6081	0.6117	0.6544	**0.6962**	0.4963	0.4767	0.5051	0.5118
	binary	0.6107	0.6107	0.6380	0.6599	**0.5285**	0.4170	0.4970	0.4982

TAB. 4.5 – Le pourcentage de dispersion des termes sur les catégories des corpus Reuters et WebKB

	T10	T20	>10%	>5%
Reuters	21.7%	20.6%	27.4%	32.3%
WebKB	37%	42.6%	45.5%	49.1%

choisis (i.e. il n'existe que peu de termes pertinents et beaucoup de termes non pertinents), V-PM subit une dégradation de performance. Comme il ressort du tableau 4.5, les termes dans le corpus WebKB sont très dispersés. Statistiquement, le pourcentage de dispersion des termes sur les catégories du corpus WebKB est approximativement deux fois supérieur à celui qui est constaté sur le corpus Reuters. Cela souligne bien le fait qu'il y a beaucoup moins de termes discriminatifs que de termes non discriminatifs dans le cas du corpus WebKB que dans celui du corpus Reuters. Ceci est particulièrement clair si l'on compare les cas des dimensionnalités T10 et T20 sur le corpus WebKB. L'effet négatif des termes non discriminants peut être partiellement réduit par certains des schémas de pondération qui considèrent les fréquences d'occurrence des termes dans les documents, mais en tout cas les résultats demeurent insatisfaisants. La méthode CS présente une solution raisonnable devant une telle situation.

En vertu des résultats précédents, la discussion qui suit ne concerne que le comportement des modèles NDF/ILoNDF couplés avec la méthode CS.

En comparant l'impact des différents aspects expérimentaux liés aux schémas de pondération

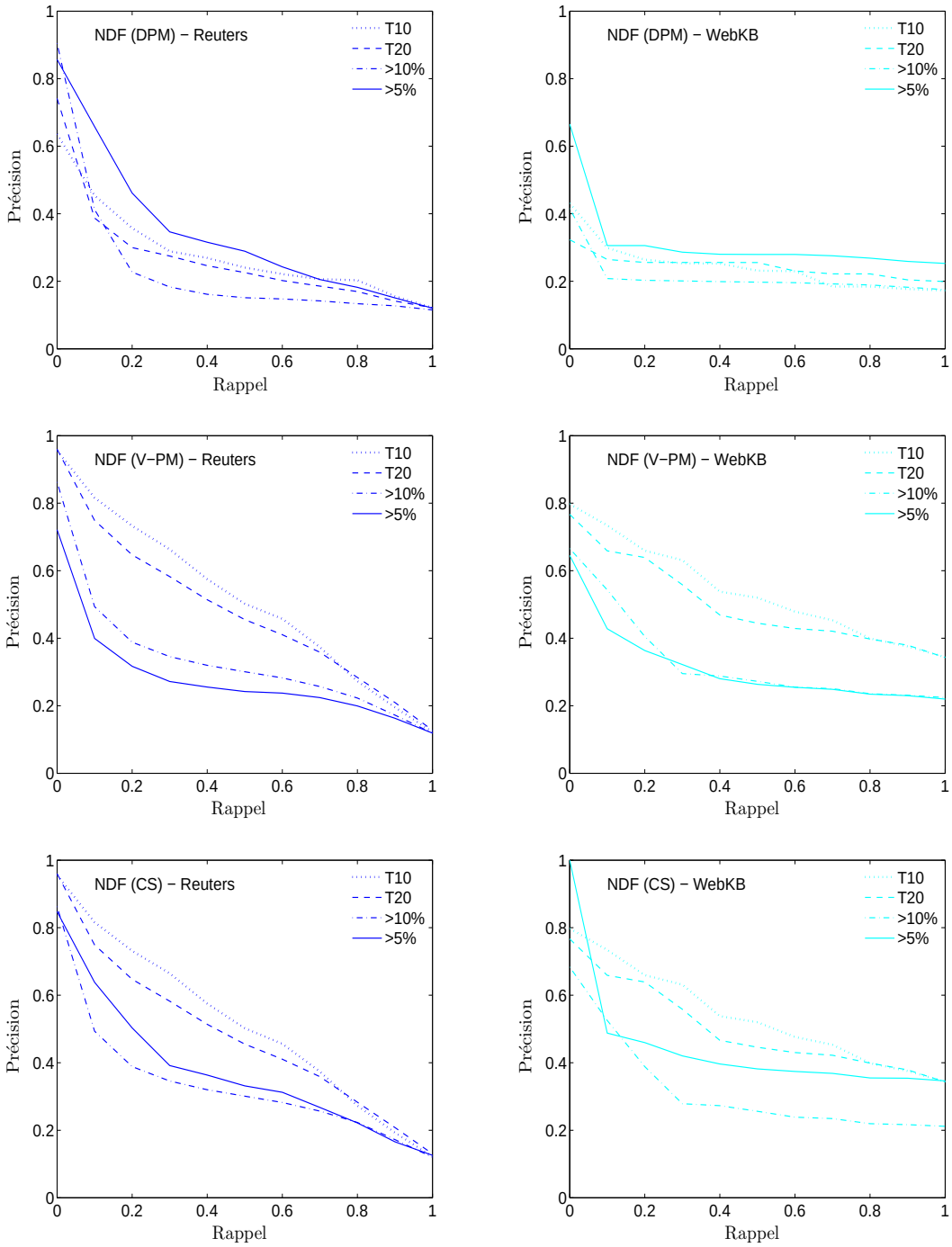


FIG. 4.6 – Les courbes Rappel-Précision correspondant au modèle NDF selon les différentes dimensionnalités de l'espace de représentation sur les corpus Reuters et WebKB. Les résultats sont rapportés en utilisant le schéma de pondération référentielle ANTF.

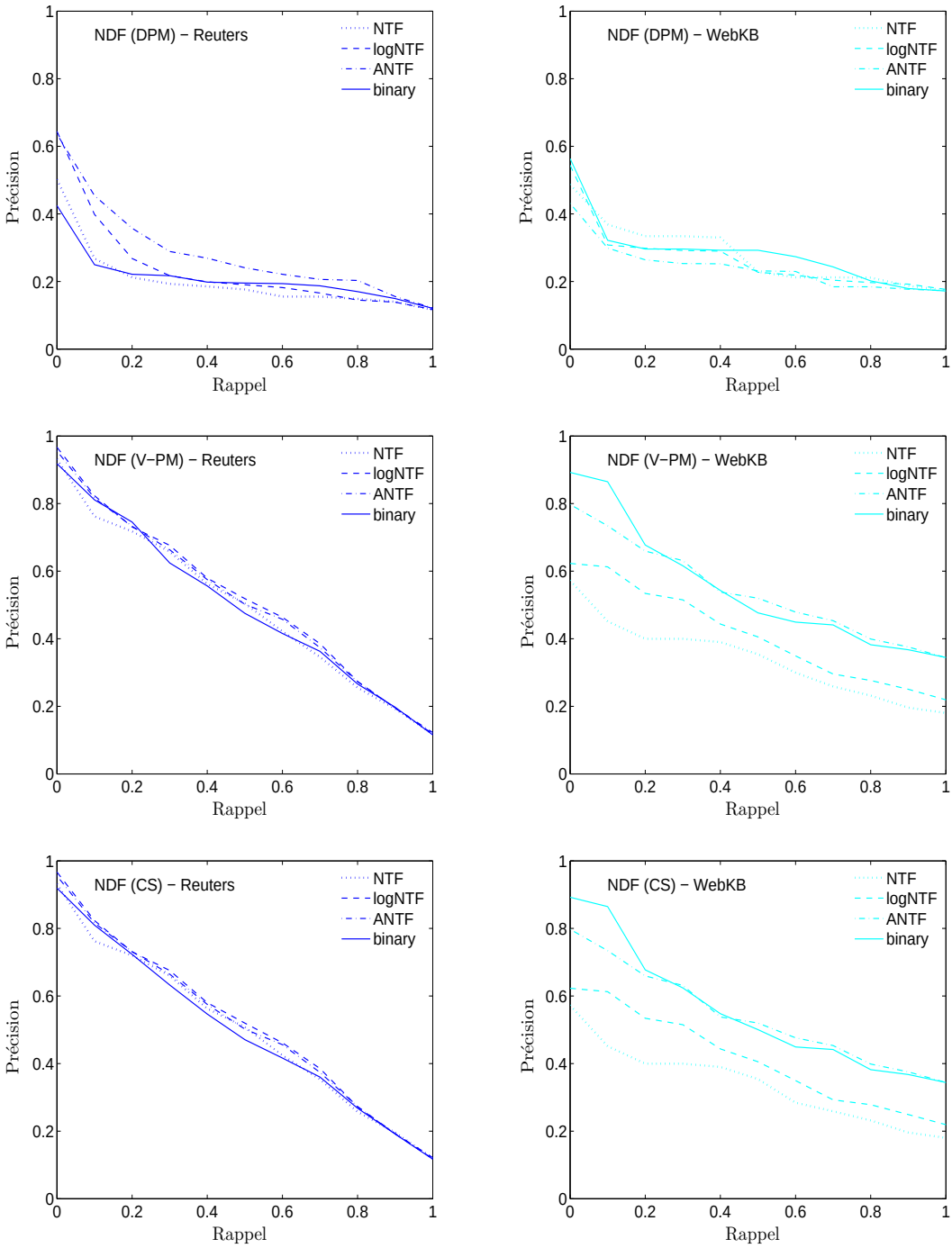


FIG. 4.7 – Les courbes Rappel-Précision correspondant au modèle NDF selon les différents schémas de pondération sur les corpus Reuters et WebKB. Les résultats sont rapportés en utilisant la dimensionnalité $T10$ de l'espace de représentation des documents.

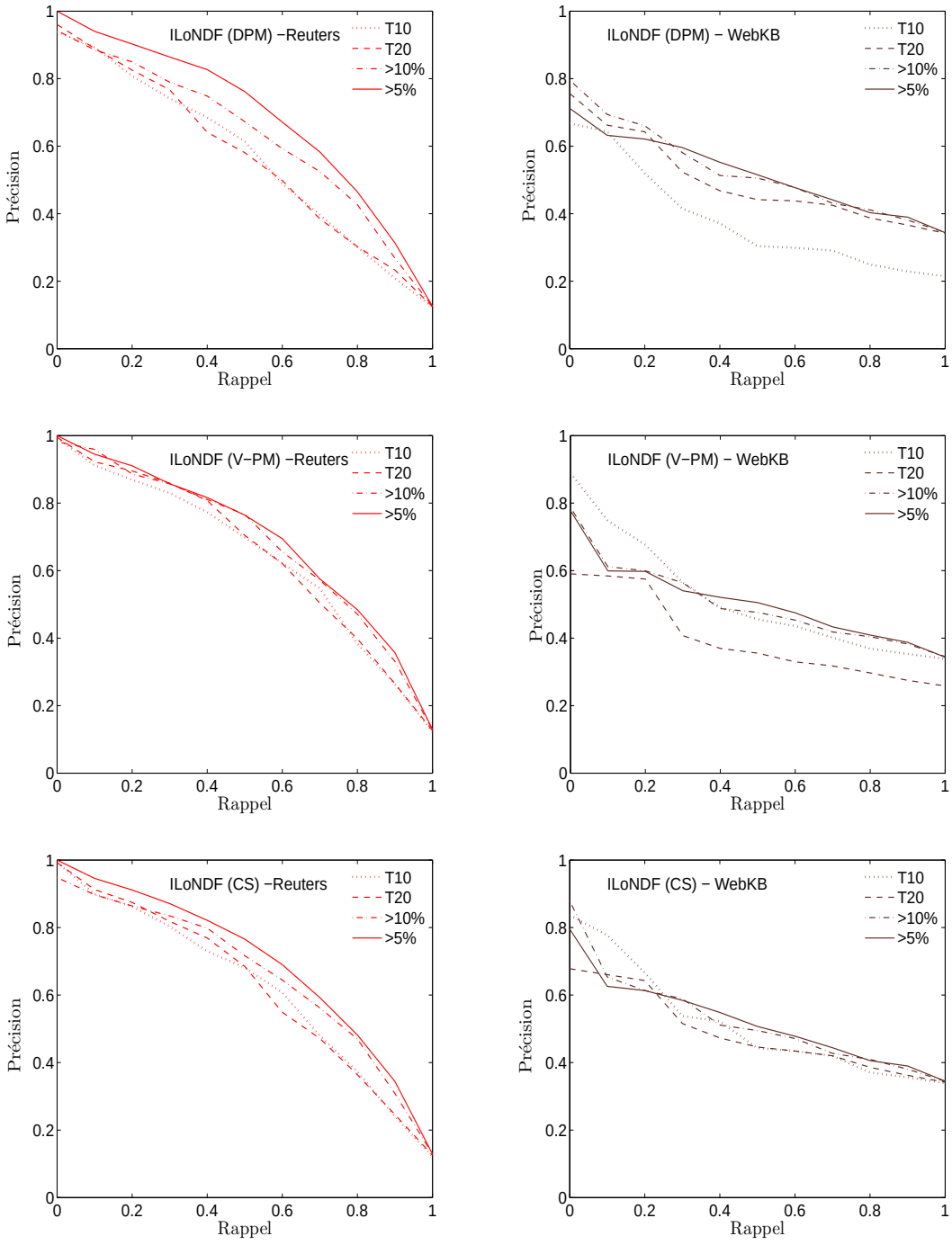


FIG. 4.8 – Les courbes Rappel-Précision correspondant au modèle ILoNDF pour les différentes dimensionnalités de l'espace de représentation sur les corpus Reuters et WebKB. Les résultats sont rapportés pour la méthode de pondération référentielle ANTF.

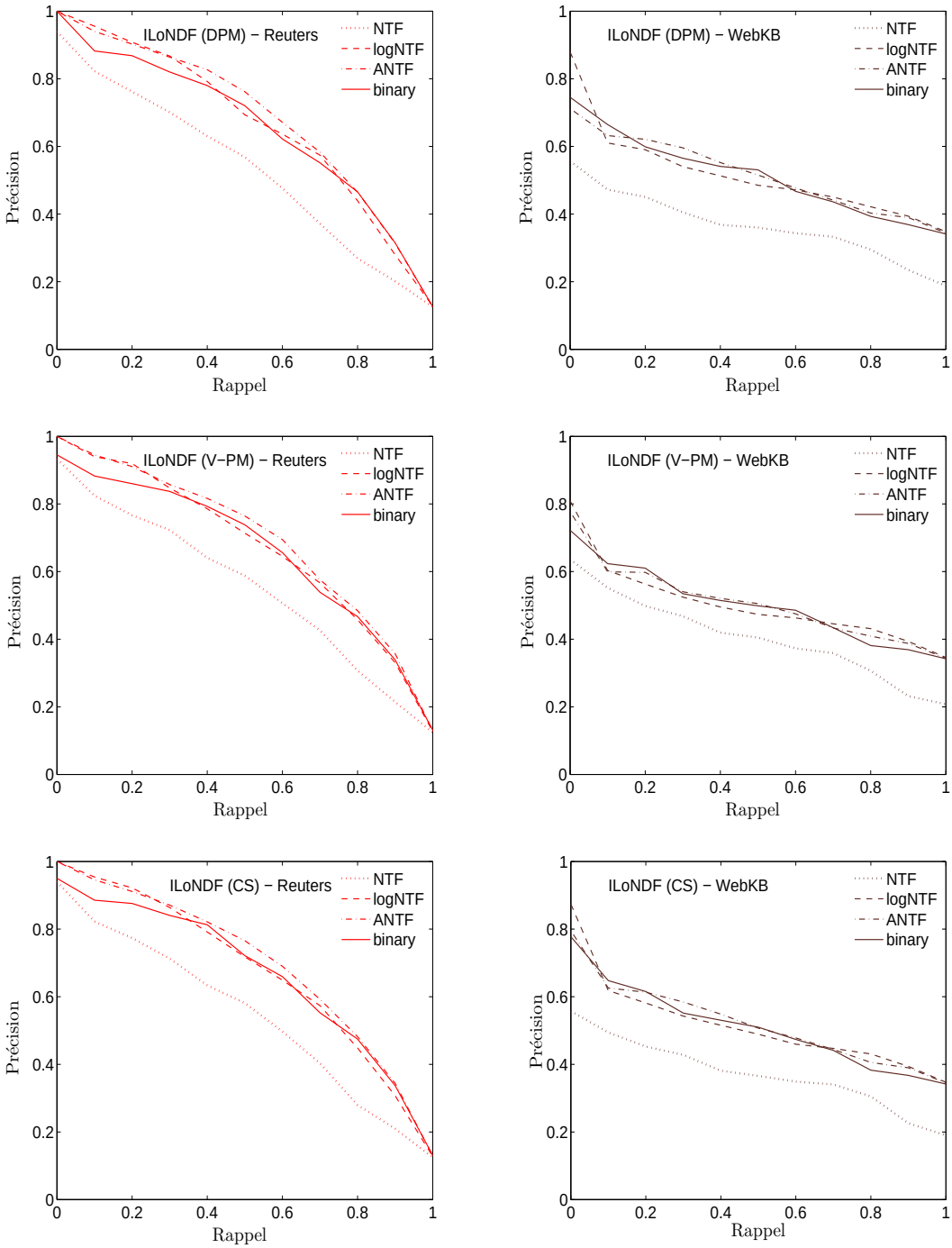


FIG. 4.9 – Les courbes Rappel-Précision correspondant au modèle ILoNDF pour les différentes méthodes de pondération sur les corpus Reuters et WebKB. Les résultats sont rapportés pour la dimensionnalité >5% de l'espace de représentation des documents.

et aux différentes dimensionnalités de l'espace de représentation des documents, nous observons en premier lieu que les modèles NDF et ILoNDF se comportent de manière similaire vis-à-vis des différents schémas de pondération (cf. Figures 4.7 et 4.9). Contrairement au cas de la classification à deux classes, la méthode de pondération NTF fournit les résultats les moins satisfaisants dans le cas de la classification à partir d'une seule classe. Une meilleure performance est atteinte par la méthode logNTF mais demeure insatisfaisante et inférieure à celle des autres méthodes de pondération. La méthode ANTF n'avait pas encore fait l'objet d'une évaluation dans le cadre de la classification à partir d'une seule classe. Lors de nos expérimentations, elle s'est avérée plus performante que la méthode de pondération binaire dans quasiment toutes les situations. En conséquence, la méthode de pondération ANTF est considérée comme la méthode de référence à la fois pour les modèles NDF et ILoNDF.

L'examen des courbes Rappel-Précision des figures 4.6 et 4.8 montre que les modèles NDF et ILoNDF se comportent différemment au regard de la dimensionnalité de l'espace de représentation des documents. Le modèle ILoNDF offre une performance stable et peu affectée par la dimensionnalité de l'espace de représentation, et tend à réaliser de meilleurs résultats dans le cas des espaces de relativement grande dimension (>5%). En revanche, la dimensionnalité de l'espace de représentation peut sérieusement affecter la performance du modèle NDF, qui s'aggrave très rapidement lorsque la dimension de l'espace augmente.

De manière générale, si l'on compare la performance du modèle NDF à celle de sa version améliorée le modèle ILoNDF, on constate que ILoNDF surpasse largement NDF, en particulier dans le cas des espaces de grande dimension.

Autres méthodes candidates pour la classification à partir d'une seule classe

L'objectif de notre deuxième expérimentation est d'examiner le comportement des autres méthodes de classification sous différents aspects expérimentaux. Les méthodes étudiées sont celles décrites dans la section 4.7, à savoir les résidus de l'ACP, le test T^2 de Hotelling, les réseaux de neurones auto-associatifs (AANN) et les SVM monoclasses (1-SVM). Le tableau 4.6 résume les scores MAP obtenus par les méthodes précitées pour les différentes méthodes de pondération et les différentes dimensionnalités de l'espace de représentation. La figure 4.10 montre les courbes Rappel-Précision correspondant aux méthodes pour la pondération de référence associée à chacune de ces méthodes et en variant la dimensionnalité de l'espace de représentation sur les corpus Reuters et WebKB. La figure 4.11 montre les courbes Rappel-Précision correspondant aux méthodes pour la dimensionnalité de référence associée à chacune des méthodes et en variant les méthodes de pondération.

En observant de manière globale les résultats obtenus, il apparaît que les méthodes basées sur l'ACP (les résidus de l'ACP et le test T^2 de Hotelling) rendent une performance très médiocre sur les corpus Reuters et WebKB. La performance de la méthode des résidus est légèrement meilleure que celle du test T^2 sur le corpus Reuters mais plus mauvaise sur le corpus WebKB. Il est difficile d'identifier les meilleurs résultats selon les différents aspects expérimentaux. De manière générale, les schémas de pondération logNTF et ANTF semblent être les meilleurs pour la méthode des résidus de l'ACP, en particulier dans le cas des espaces de grande dimension. En revanche, la méthode du test T^2 peut produire de meilleurs résultats dans le cas des espaces de faible dimension et le meilleur schéma de pondération varie selon le corpus de test : ANTF sur le corpus de Reuters, et binary sur le corpus WebKB. Dans une certaine mesure, les résultats

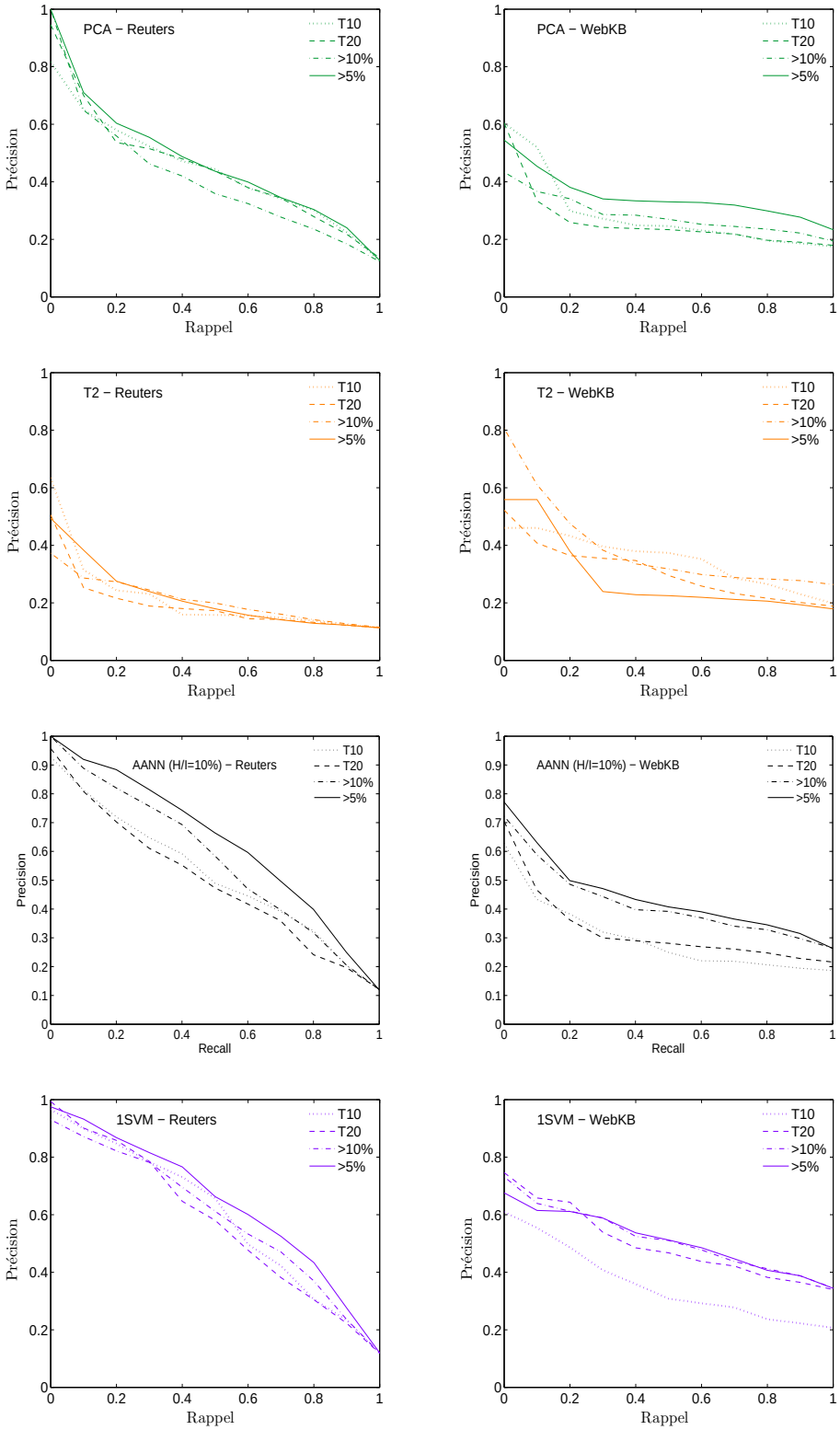


FIG. 4.10 – Les courbes Rappel-Précision correspondant aux méthodes typiques de la classification à une classe, à savoir les résidus de l'ACP, le test T^2, AANN, et 1-SVM, pour les différentes dimensionnalités de l'espace de représentation sur les corpus Reuters et WebKB. Les résultats sont respectivement rapportés pour les méthodes de pondération : logNTF, ANTF, logNTF et binary.

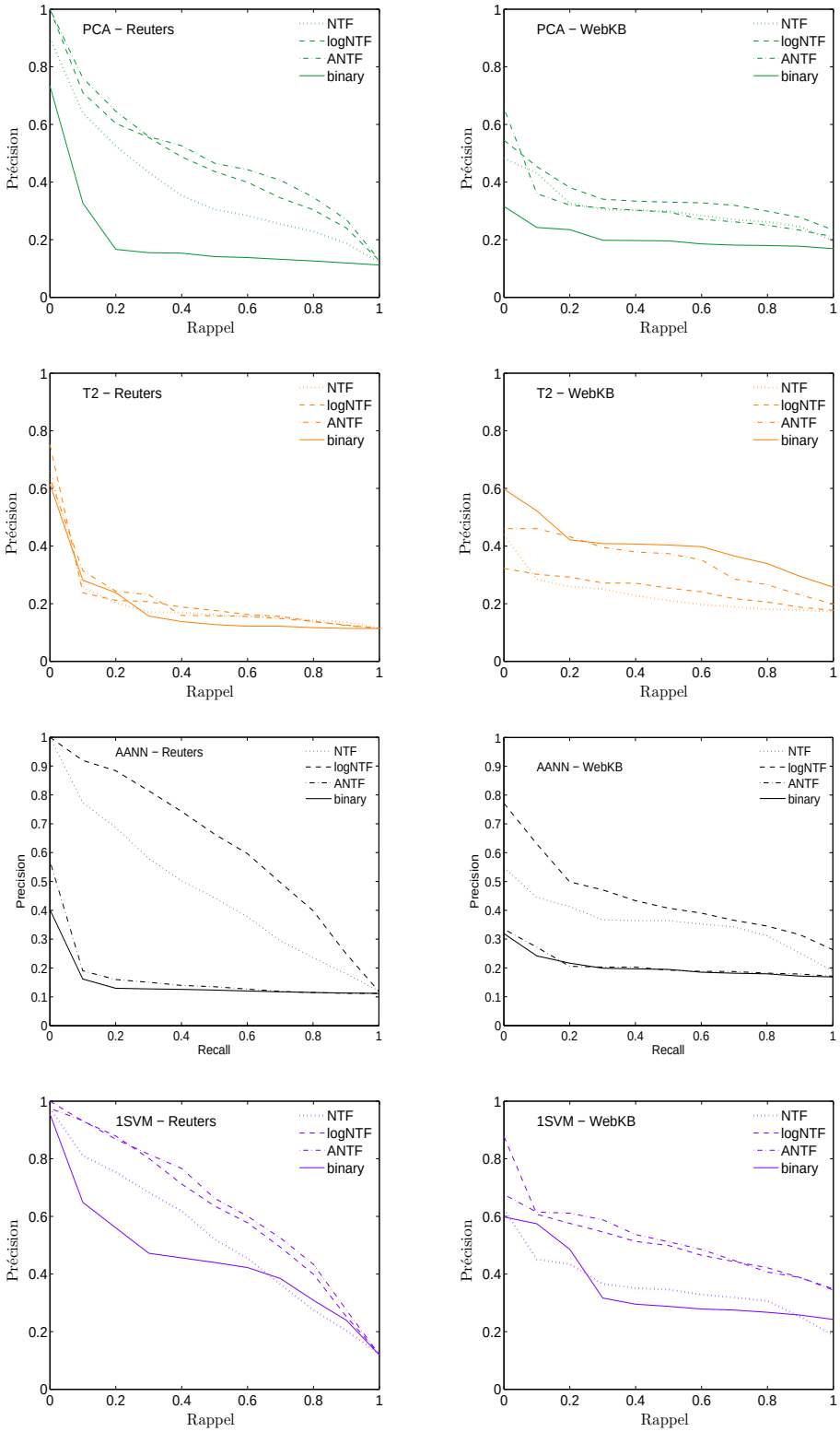


FIG. 4.11 – Les courbes Rappel-Précision correspondant aux méthodes typiques de la classification à une classe (les résidus de l'ACP, le test T^2, AANN, et 1-SVM) pour les différentes méthodes de pondération sur les corpus Reuters et WebKB. Les résultats sont rapportés pour la dimensionnalité $T10$ en ce qui concerne le test T^2, alors qu'ils sont rapportés pour la dimensionnalité >5% en ce qui concerne les autres méthodes.

TAB. 4.6 – Moyenne des scores MAP correspondant aux méthodes typiques de la classification à une classe (les résidus de l'ACP, le test T^2, AANN, et 1-SVM). Les résultats sont rapportés sur les corpus Reuters et WebKB et obtenus selon les différentes méthodes de pondération et les différentes dimensionnalités de l'espace de représentation des documents. Les entrées en gras indiquent les scores les plus élevés statistiquement, alors que les entrées soulignées indiquent les scores correspondant aux conditions référentielles retenues selon les résultats globaux des méthodes étudiées.

		Reuters				WebKB			
		NTF	logNTF	ANTF	binary	NTF	logNTF	ANTF	binary
PCA residuals	T10	0.3167	0.4202	0.4680	0.3255	0.2529	0.2654	0.2342	0.2650
	T20	0.3308	0.4307	0.4772	0.2095	0.2198	0.2443	0.2303	0.2112
	>10%	0.2967	0.3915	0.4314	0.1530	0.2568	0.2641	0.2472	0.1799
	>5%	0.3569	0.4307	**0.4898**	0.1694	0.2880	**0.3304**	0.2889	0.1834
Hotelling T^2 test	T10	0.1740	0.1715	0.1801	0.1492	0.2099	0.2321	0.3251	**0.3713**
	T20	0.1562	0.1553	0.1583	0.1028	0.1988	0.2985	0.2940	0.2441
	>10%	0.1736	**0.2148**	0.1828	0.0713	0.2039	0.2707	0.3596	0.1674
	>5%	0.1432	0.1952	0.1892	0.0802	0.2043	0.2420	0.2619	0.1726
AANN	T10	0.4261	0.5024	0.3452	0.2989	0.2460	0.2776	0.3217	0.4099
	T20	0.4209	0.4852	0.2363	0.1466	0.2492	0.3013	0.2324	0.2448
	>10%	0.4234	0.5903	0.1294	0.1380	0.3451	0.4055	0.1733	0.1703
	>5%	0.4495	**0.6272**	0.1370	0.1148	0.3361	**0.43**	0.1824	0.18
1-SVM	T10	0.4846	0.5439	0.5756	0.5823	0.2360	0.2893	0.3387	**0.5377**
	T20	0.4703	0.5333	0.5589	0.5510	0.2479	0.4512	0.4819	0.4901
	>10%	0.4838	0.5711	0.5832	0.4788	0.3271	0.3954	0.5030	0.3725
	>5%	0.5152	0.6204	**0.6356**	0.4277	0.3384	0.5035	0.5038	0.3277

indiquent un degré de similarité entre les méthodes basées sur l'ACP et le modèle NDF couplé avec DPM (cf. Tableau 4.3). C'est probablement parce que toutes ces méthodes réalisent un niveau important de décorrélation entre données. Le décorrélation assure la détection des données qui dévient de manière significative des données utilisées pour l'apprentissage. Mais, en présence de certains bruits corrélés dans la représentation des données d'apprentissage, le mécanisme de décorrélation peut affecter de manière négative les résultats de la classification.

Les réseaux auto-associatifs (AANN) offrent une meilleure performance que les méthodes basées sur l'ACP. Néanmoins, ils présentent une grande sensibilité aux différents aspects expérimentaux tels que les schémas de pondération et la dimensionnalité de l'espace de représentation, mais ils sont moins affectés par le nombre de neurones cachés comme il ressort du tableau 4.7. Leurs meilleurs résultats sont obtenus pour la dimension la plus grande de l'espace de représentation (>5%) en utilisant le schéma de pondération logNTF et un nombre restreint de neurones cachés ($H/I = 10\%$).

Parmi les quatre méthodes candidates que nous avons examinées pour la tâche de la classification à partir d'une seule classe, 1-SVM réalise la meilleure performance. Lorsque testée avec le

TAB. 4.7 – L'impact de la taille du réseau AANN (le nombre de neurones cachés (H) par rapport à celui de neurones dans les couches d'entrée (ou de sortie) (I)) sur les rsultats de la classification. Les résultats sont rapportés en termes des scores MAP et en utilisant la méthode de pondération logNTF. Les entrées en gras indiquent les scores les plus élevés statistiquement, alors que les entrées soulignées indiquent les scores correspondant aux conditions référentielles retenues selon les résultats globaux du réseau.

	Reuters				WebKB			
(H/I)	10%	25%	50%	75%	10%	25%	50%	75%
T10	0.5024	0.5042	0.4580	0.4877	0.2776	0.3030	0.2509	0.2962
T20	0.4852	0.4801	0.5072	0.4762	0.3013	0.2868	0.2790	0.2981
>10%	0.5903	0.5829	0.6257	0.5918	0.4055	0.3836	0.4035	0.4084
>5%	0.6272	0.6110	0.5947	**0.6335**	**0.43**	0.4183	0.3785	0.3510

TAB. 4.8 – Les scores MAP de la méthode 1-SVM pour differentes fonctions de noyau (le schéma de pondération est ANTF).

Fonction noyau	Reuters				WebKB			
$K(x, y)$	T10	T20	>10%	>5%	T10	T20	>10%	>5%
Linéaire $x \cdot y$	0.5756	0.5589	0.5832	0.6356	0.3387	0.4819	0.5030	0.5038
Polynomiale $(x \cdot y + 1)^3$	0.5690	0.5535	0.5885	0.6365	0.3840	0.4683	0.5038	0.5075
RBF $\exp(-0.1\|x - y\|^2)$	0.5755	0.5607	0.5873	0.6369	0.3434	0.4801	0.5038	0.5029
Sigmoïdale $tanh(x \cdot y)$	0.5785	0.5559	0.5781	0.6303	0.3452	0.48	0.5047	0.5026

schéma de pondération binaire, la méthode 1-SVM souffre de dégradation de performance avec l'augmentation de la dimensionnalité de l'espace de représentation, ce qui est en accord avec des études précédentes (Manevitz et Yousef, 2001). Cependant, 1-SVM présente un comportement opposé lorsque les autres schémas de pondération sont utilisés. La meilleure performance est atteinte en utilisant le schéma ANTF et la dimensionnalité la plus grande (>5 %).

En plus de la fonction noyau linéaire, 1-SVM a été testée avec des fonctions non linéaires (les fonctions polynomiales de degré d, les fonctions à base radiale (RBF) et les fonctions sigmoïdes). Les résultats du test sont reportés dans le tableau 4.8. Ils indiquent que les fonctions non linéaires ne parviennent pas à améliorer la performance de la classification, ce qui est souvent expliqué par le fait que les problèmes de classification de textes sont linéairement séparables et donc la transformation dans un espace de plus grande dimension n'est pas nécessaire.

Le point clé de la méthode 1-SVM réside dans le choix du paramètre ν qui la rend tolérante aux bruits qui pourraient être présents dans les données utilisées pour l'apprentissage. La figure

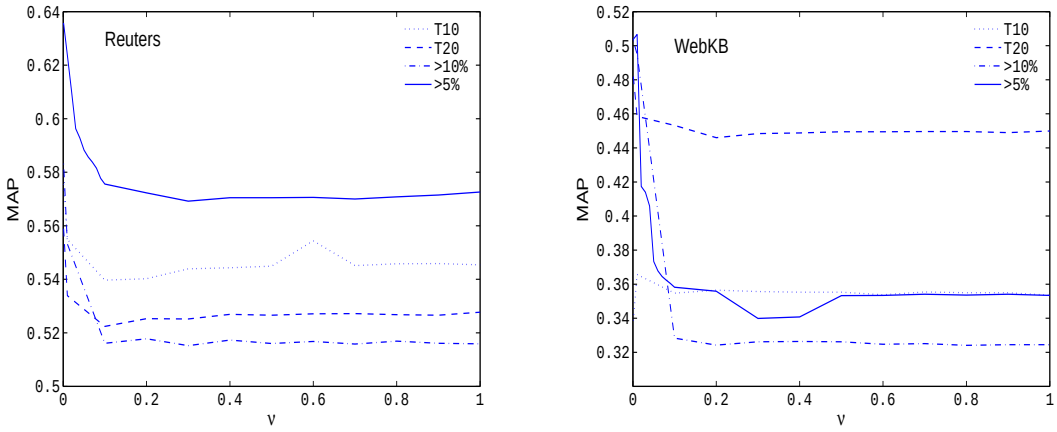


FIG. 4.12 – Les scores MAP de la méthode 1-SVM en fonction de la valeur du paramètre ν (a) la collection Reuters (b) la collection WebKB (le schéma de pondération est ANTF).

4.12 montre les courbes des scores MAP obtenus sur les corpus Reuters et WebKB en variant la valeur du paramètre ν. En augmentant ν, la description de la classe positive devient de plus en plus spécifiques aux données d'apprentissage, et par conséquent, l'identification de nouvelles données positives devient aussi de plus en plus difficile. La valeur optimale de ν a été choisie après plusieurs essais comme étant celle correspondant à la meilleure valeur des scores MAP et s'est avérée très basse à la fois sur les corpus Reuters et WebKB ($\nu \simeq 0.001$). La raison tient au fait que les collections de test standards sont généralement soigneusement formées et ainsi elles sont peu susceptibles de contenir beaucoup de données bruitées. En général, la détermination de la valeur du paramètre ν est un problème délicat, en particulier lorsqu'on n'a aucune connaissance préalable de la nature des données d'apprentissage.

Évaluation des stratégies de seuillage

Nous tournons maintenant notre attention vers l'évaluation des stratégies de seuillage utilisées par les différentes méthodes de classification. Au vu de la faible performance du modèle NDF et des méthodes basées sur l'ACP, nous ne donnons pas ici les détails de leurs résultats. Nous nous concentrons sur la performance du modèle ILoNDF et celle des méthodes AANN et 1-SVM. Le tableau 4.9 résume les valeurs des macro- et micro-moyennes des critères d'évaluation : Précision, Rappel et F_1 sur les corpus Reuters et WebKB.

Selon les résultats obtenus pour AANN, il semble que la stratégie de seuillage que nous avons decrite dans la section 4.7.2 est intéressante. L'apprentissage du réseau sur plusieurs époques d'entraînement permet de produire une estimation fiable des scores de classification des données positives. Mais le seuil estimé s'avère favoriser le rappel au détriment de la précision dans le cas des espaces de faible dimension, et au contraire, favoriser la précision au détriment du rappel dans le cas des espaces de grande dimension. Dans les deux cas, il n'y pas de différence notable entre les valeurs de F_1 obtenues selon cette stratégie de seuillage et celles obtenues selon le critère de "break-even point".

La capacité de 1-SVM à réaliser un bon compromis entre la précision et le rappel est meilleure sur le Reuters que sur le corpus WebKB. Souvent, 1-SVM attache relativement plus d'importance

TAB. 4.9 – Comparaison de performance de classification des méthodes ILoNDF, AANN, et 1-SVM sur les corpus Reuters et WebKB. Les résultats sont rapportés en utilisant le schéma de pondération logNTF pour AANN, et du schéma de pondération ANTF pour ILoNDF et 1-SVM. Les entrées en gras dénotent les scores les plus élevés réalisés par les différentes méthodes.

				Macro-moyennes				Micro-moyennes			
				P	R	F_1	BKP	P	R	F_1	BKP
Reuters	AANN		T10	0.4695	0.5474	0.4218	0.4886	0.2887	0.4716	0.3582	0.5519
			T20	0.4896	0.5221	0.4484	0.4767	0.3845	0.4605	0.4191	0.5518
			>10%	0.6850	0.4544	0.5387	0.5561	0.7382	0.5159	0.6074	0.6078
			>5%	0.7432	0.4338	0.5421	0.5935	0.8329	0.5081	0.6312	0.7011
	1-SVM		T10	0.5320	0.5880	0.5150	0.5650	0.4327	0.5644	0.4899	0.5780
			T20	0.5043	0.5815	0.5011	0.5512	0.4136	0.5619	0.4765	0.5503
			>10%	0.5807	0.5688	0.5591	0.5734	0.6278	0.6167	0.6222	0.6710
			>5%	0.6452	0.5598	0.5896	0.6086	0.7321	**0.6292**	0.6768	0.7192
	ILoNDF	$\tau = 1$	T10	0.6613	0.5175	0.5708	0.5942	0.6121	0.4672	0.5299	0.5706
			T20	0.6151	0.5217	0.5309	0.5851	0.5488	0.5020	0.5749	0.5749
			>10%	0.7254	0.5121	0.5871	0.6313	0.7524	0.5297	0.6217	0.7034
			>5%	**0.8173**	0.4708	0.59	**0.6552**	**0.8776**	0.5414	0.6696	**0.7514**
		$\tau = 0.95$	T10	0.5171	**0.6275**	0.5327	—	0.4289	0.5952	0.4985	—
			T20	0.5392	0.6183	0.5374	—	0.4424	0.5943	0.5072	—
			>10%	0.6626	0.6056	0.6123	—	0.6828	0.6341	0.6575	—
			>5%	0.7514	0.5557	**0.6291**	—	0.8257	0.6249	**0.7114**	—
WebKB	AANN		T10	0.2110	0.6242	0.2779	0.2643	0.2713	**0.6169**	0.3768	0.2853
			T20	0.2563	0.5663	0.31	0.2749	0.2473	0.4597	0.3216	0.3042
			>10%	0.3037	0.2088	0.2270	0.2914	0.3476	0.2621	0.2989	0.3257
			>5%	0.4587	0.4310	0.3746	0.3007	0.4490	0.3548	0.3964	0.3445
	1-SVM		T10	0.2839	0.6257	0.3231	0.2946	0.3114	0.5524	0.3983	0.4262
			T20	0.3150	0.5554	0.3306	0.4584	0.3372	0.4637	0.3905	0.4637
			>10%	0.3747	0.5134	0.3652	0.4686	0.3732	0.4274	0.3985	0.5
			>5%	0.4297	0.4634	**0.4091**	**0.4810**	0.4322	0.4113	0.4215	**0.5323**
	ILoNDF	$\tau = 1$	T10	0.2906	0.5898	0.3268	0.4739	0.3351	0.5202	0.4076	0.4718
			T20	0.3083	0.5394	0.3136	0.4716	0.3182	0.4234	0.3633	0.4556
			>10%	0.3966	0.4509	0.3422	0.4889	0.3489	0.3306	0.3395	0.5081
			>5%	**0.4620**	0.4102	0.3816	0.4784	**0.4540**	0.3185	0.3744	0.5242
		$\tau = 0.95$	T10	0.2439	**0.6480**	0.3063	—	0.2857	0.6048	0.3881	—
			T20	0.2589	0.6007	0.2970	—	0.2806	0.5081	0.3615	—
			>10%	0.32	0.5607	0.3359	—	0.3306	0.4839	0.3928	—
			>5%	0.3659	0.5012	0.3684	—	0.4094	0.4556	**0.4313**	—

au rappel qu'à la précision même pour la valeur optimale du paramètre ν.

La stratégie de seuillage de la section 4.6 appliquée au modèle ILoNDF est plutôt intéressante. Les résultats sont globalement bons même sans utilisation du paramètre de relaxation τ (i.e. pour $\tau = 1$). Dans ce cas, le seuil s'avère favoriser la précision au détriment du rappel, ceci en particulier dans le cas des espaces de grande dimension. Pour le problème de la classification à une classe, il est important que la méthode de classification fasse preuve d'une forte précision tout en maintenant un niveau de rappel acceptable. Ceci est plus délicat à obtenir, partiellement à cause du fait que les données positives sont souvent beaucoup plus rares que les données négatives, ainsi un gain important en rappel est toujours au détriment de la précision, et vice versa. Dans d'autres situations où le besoin d'avoir un fort rappel aurait plus d'importance que le besoin d'avoir une forte précision, le paramètre de relaxation peut être utilisé pour régler la différence entre la précision et le rappel. Une illustration est donnée dans le tableau 4.9 en fixant la valeur du paramètre τ à 0.95.

L'observation des résultats du tableau 4.9 permet de constater que les micro-moyennes des scores breakeven (dominées par la performance sur les catégories communes) sont souvent plus élevées que les macro-moyennes (dominées par la performance sur les catégories moins fréquentes). Cela signifie que la performance de toutes les méthodes étudiées tend à s'améliorer avec l'augmentation du nombre de données d'apprentissage. Néanmoins, la différence entre les micro-moyennes et les macro-moyennes n'est pas moins importante dans le cas du modèle ILoNDF que dans les cas des méthodes AANN et 1-SVM. Par conséquent, nous pouvons conclure que la performance du modèle ILoNDF est moins affectée par la quantité de données d'apprentissage, mais elle dépendait principalement de la qualité des données utilisées pour l'apprentissage.

Dans une note finale, il faut préciser que, tandis que le modèle ILoNDF est bien plus performant que les autres méthodes (cf. Tableaux 4.3, 4.4 and 4.6), sa supériorité, en particulier, sur 1-SVM, devient moins prononcée après le seuillage des scores de classification. Ceci suggère que la stratégie de seuillage devrait être encore améliorée pour profiter pleinement des capacités du modèle ILoNDF.

4.8.2 Une comparaison directe des méthodes de classification des données fortement multidimensionnelles

Nous poursuivons la présentation de nos expérimentations dans cette section en procédant à une comparaison directe de la performance du modèle ILoNDF avec celle des meilleures méthodes de classification identifiées, à savoir AANN et 1-SVM. Nous présentons également des résultats tirés des expérimentations conduites le modèle NDF dans le but de mieux comprendre les relations entre les deux modèles : NDF et ILoNDF. La comparaison est centrée sur le cas des données fortement multidimensionnelles. Nous nous limitons donc à la dimensionnalité la plus grande (>5 %) de notre gamme de conditions expérimentales. La figure 4.13 montre les courbes de Rappel-Précision des méthodes NDF, ILoNDF, AANN et 1-SVM. La figure 4.14 montre les scores MAP et F_1 sur les différentes catégories des corpus Reuters et WebKB. Le tableau 4.10 résume les scores MAP et F_1 obtenus par les différentes méthodes.

À partir des résultats obtenus, nous soulevons les constatations suivantes : Le modèle NDF n'est pas particulièrement efficace sur le corpus Reuters. Le modèle ILoNDF dépasse sensiblement la performance du modèle NDF, jusqu'à 30% en termes des scores F_1 sur ce même corpus. En

TAB. 4.10 – Les scores MAP et les scores F_1 correspondant aux points de break-even obtenus par les méthodes NDF, ILoNDF, AANN et 1-SVM. Les résultats sont rapportés en utilisant les schémas de pondération référentiels et la dimensionnalité la plus grande de l'espace de représentation (>5 %). Les entrées en gras indiquent les meilleurs scores.

	Reuters		WebKB	
	MAP	BKP	MAP	BKP
NDF	0.3512	0.3584	0.4075	0.4227
AANN	0.6272	0.5935	0.43	0.3007
1-SVM	0.6356	0.6086	0.5038	**0.4810**
ILoNDF	**0.6962**	**0.6552**	**0.5118**	0.4784

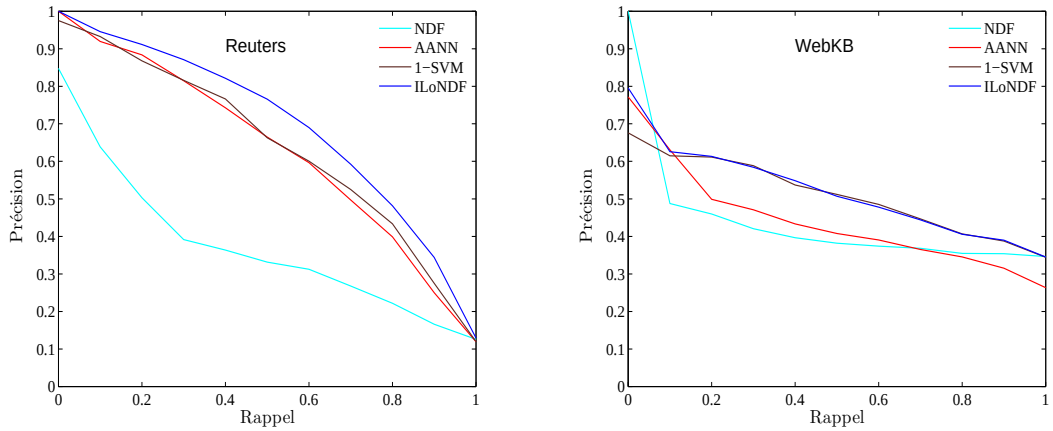


FIG. 4.13 – Les courbes Rappel-Précision des méthodes NDF, AANN, 1-SVM et ILoNDF. Les résultats sont rapportés pour les schéma de pondération référentiels et la dimensionnalité la plus grande de l'espace de représentation.

outre, ILoNDF dépasse constamment les autres méthodes, portant des améliorations d'environ 6-7 % sur les méthodes 1-SVM et d'AANN, respectivement. ILoNDF tend à être particulièrement utile dans le cas des petites catégories ambiguës où la représentation des données peut être très bruitée. Par exemple, les catégories "ship" et "crude" sont connues pour être fortement recouvrantes et ainsi partagent beaucoup de termes communs (le pourcentage de dispersion des termes sur ces deux catégories est 70.5%). De même, la catégorie "trade" est aussi en lien avec la catégorie "earn" (le pourcentage de dispersion des termes est 55.5%) mais la catégorie "earn" est moins affectée par ce phénomène en raison de sa taille importante par rapport à celle de "trade". Pour sa part, la méthode 1-SVM semble être légèrement meilleure que la méthode AANN, avec une amélioration de l'ordre de +1%.

La performance de toutes les méthodes de classification se dégrade en passant du corpus Reuters au corpus WebKB. La différence (en termes de performance) entre les modèles NDF et ILoNDF est réduite de 30-35% à 5-11%. En outre, la performance du modèle ILoNDF est toujours légèrement meilleure que celle des autres méthodes, bien que l'amélioration de la performance soit moins significative que celle obtenue sur le corpus Reuters (généralement d'environ +1-8 %) en termes des scores MAP. Notre hypothèse est que la classification des documents issus

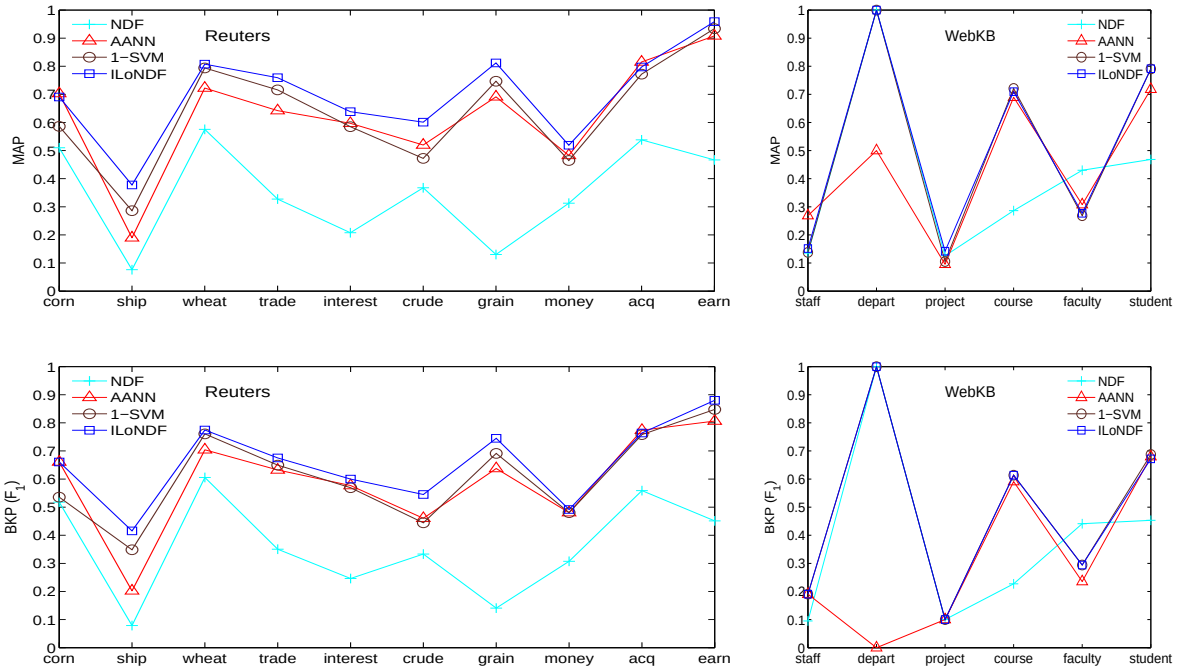


FIG. 4.14 – Une comparaison entre les méthodes de classification NDF, AANN, 1-SVM et ILoNDF sur les corpus Reuters et WebKB en utilisant la dimensionnalité la plus grande de l'espace de représentation (>5%). Les résultats sont rapportés en utilisant le schéma de pondération logNTF pour AANN, et le schéma de pondération ANTF pour ILoNDF et 1-SVM.

du web peut être plus difficile que la classification des documents de nature plus spécifique. En effet, les documents issus du web — à la différence de la plupart des collections typiquement utilisées pour l'évaluation expérimentale des méthodes de classification de textes — présentent un manque d'homogénéité et de régularité[32]. Comme précisé précédemment à l'aide du tableau 4.5, les termes extraits du corpus WebKB sont très dispersés. Le pourcentage de dispersion des termes sur les catégories est de 49.1% sur le corpus WebKB et 32.3 % sur le corpus Reuters. Ainsi, il y a beaucoup moins de termes spécifiques pour chaque catégorie contre un nombre plus important de termes non discriminatifs (bruits) dans le cas du corpus WebKB, ce qui résulte en une performance nettement inférieure des méthodes de classification y compris ILoNDF.

Pour conclure sur cette section, il est important de noter que le modèle ILoNDF a un coût de calcul comparable à celui de la méthode 1-SVM, et notablement moins important que celui de AANN. À titre indicatif, nous avons réalisé des tests sur un processeur Intel Core 2 Duo à 3.00 GHz et 2 Go de mémoire vive. En moyenne, sur les 10 catégories du corpus Reuters, la méthode SVM est la plus efficace en termes de temps d'apprentissage (temps CPU, < 2 sec/catégorie). Le modèle ILoNDF est la deuxième méthode la plus efficace ($\simeq 24$ sec/catégorie).

[32]Un aspect relatif à la classification des documents issus du web que nous n'avons pas eu l'occasion d'examiner de manière approfondie concerne l'exploitation des informations riches dans ce type de documents et de la connectivité entre documents pour améliorer la qualité de la classification. Les liens hypertextes, les balises HTML, et les méta-données sont des exemples des informations qu'on peut extraire et utiliser pour la classification de documents web, et qui ne sont pas typiquement disponibles lors de la classification traditionnelle de textes (Yang et al., 2002a).

Enfin, l'AANN est beaucoup plus lent ($\simeq$ 1302 sec/catégorie). À ce titre, il est important de préciser que notre développement actuel du modèle ILoNDF ne supporte pas l'utilisation des bibliothèques de fonctions de manipulation de matrices creuses. Dès lors, nous pensons que le recours à de telles fonctions peut nous aider à réduire considérablement le temps de calcul relatif au modèle ILoNDF. Par ailleurs, en raison des nombreux paramètres intervenant dans la méthode 1-SVM, une série d'expériences de validation est généralement effectuée pour déterminer leurs valeurs optimales. De telles optimisations sont coûteuses en temps de calcul, et augmentent ainsi de manière significative la complexité de calcul de 1-SVM. Pour sa part, ILoNDF ne requiert pas de telles optimisations et s'avère être moins sensible aux aspects expérimentaux tels que les schéma de pondération et la dimensionnalité de l'espace de représentation.

4.8.3 Synthèse des résultats expérimentaux

Nous reprenons ici, de manière synthétique, les résultats expérimentaux qui viennent d'être présentés. Nous avons étudié, dans un premier temps, le comportement des modèles NDF et ILoNDF vis-à-vis des différentes méthodes de calcul des scores de classification : DPM, V-PM et CS. Nous avons aussi testé la robustesse des modèles vis-à vis des différents aspects expérimentaux tels que les schémas de pondération et la dimensionnalité de l'espace de représentation. II ressort des résultats les points suivants :

- Le comportement des modèles NDF et ILoNDF est très différent vis-à-vis des méthodes DPM et V-PM. La méthode CS s'est avérée être une solution raisonnable aux problèmes éventuels qui peuvent se produire lors de l'utilisation des méthodes précitées ;
- La méthode de pondération ANTF est la méthode la plus sûre à la fois pour les modèles NDF et ILoNDF dans le cadre de la classification à partir d'une seule classe ;
- Les modèles NDF et ILoNDF se comportent différemment au regard de la dimensionnalité de l'espace de représentation des documents. Le modèle NDF réalise ses meilleurs résultats dans le cas des espaces de relativement faible dimension, alors que le modèle ILoNDF réalise ses meilleurs résultats dans le cas des espaces de relativement grande dimension ;
- La supériorité du modèle ILoNDF sur le modèle NDF est très claire, en particulier dans le cas des espaces de représentation de grande dimension.

Dans un deuxième temps, nous avons examiné le comportement des autres méthodes de classification sous les différents aspects expérimentaux. Les méthodes étudiées sont les résidus de l'ACP, le test T^2 de Hotelling, les réseaux de neurones auto-associatifs (AANN) et les SVM monoclasses (1-SVM). Les expérimentations ont révélé les résultats suivants :

- Les méthodes basées sur l'ACP (les résidus de l'ACP et le test T^2 de Hotelling) rendent une performance très médiocre et relativement proche de celle du modèle NDF. Il a été difficile de tirer des conclusions claires concernant le meilleur schéma de pondération ou la dimensionnalité la plus adaptée pour ces méthodes ;
- Les réseaux auto-associatifs (AANN) offrent une meilleure performance que les méthodes basées sur l'ACP. Ils présentent une grande sensibilité aux différents aspects expérimentaux. Leurs meilleurs résultats sont obtenus en utilisant un nombre restreint de neurones cachés, le schéma de pondération logNTF et la dimension la plus grande de l'espace de représentation ;
- Parmi les quatre méthodes étudiées, 1-SVM a réalisé la meilleure performance. La meilleure performance est atteinte en utilisant le schéma ANTF et la dimensionnalité la plus grande de l'espace de représentation. Les fonctions non linéaires n'apportent pas d'améliorations claires par rapport à la fonction noyau linéaire dans le cas des données textuelles, mais le

choix de la valeur du paramètre ν est très important et affecte directement la performance de la méthode.

En ce qui concerne les stratégies de seuillage adoptées pour les différentes méthodes, nous avons pu constater les difficultés suivantes :

- La stratégie de seuillage appliquée pour la méthode AANN tend à favoriser le rappel au détriment de la précision dans le cas des espaces de faible dimension, et au contraire, à favoriser la précision au détriment du rappel dans le cas des espaces de grande dimension ;
- La méthode 1-SVM donne relativement plus d'importance pour le rappel que pour la précision même pour la valeur optimale du paramètre ν ;
- La stratégie de seuillage appliquée au modèle ILoNDF s'avère favoriser la précision au détriment du rappel sans l'utilisation du paramètre de relaxation τ. Le paramètre de relaxation peut être utilisé pour régler la différence entre la précision et le rappel ;
- La performance du modèle ILoNDF est moins affectée par la quantité de données d'apprentissage que les autres méthodes.

En conclusion, nos expérimentations révèlent une meilleure performance du modèle ILoNDF que celle des méthodes actuelles, en particulier dans le cas des espaces de grande dimension. D'ailleurs, dans ce cas précis, ILoNDF offre une performance supérieure ou au moins une performance comparable à celle de la méthode la plus performante connue à ce jour, 1-SVM. À cela s'ajoute sa simplicité, son aspect fonctionnel (notamment celui de l'apprentissage en ligne et en un seul passage sur les données), et son temps de calcul très raisonnable, qui répondent aux contraintes les plus limitantes liées au traitement de flux de données.

4.9 Conclusion

Nous avons présenté dans ce chapitre un nouveau modèle d'apprentissage inspiré par la théorie de détection de nouveauté, baptisé ILoNDF ("Incremental data-driven Learning of Novelty Detector Filter"). Le principe du modèle est particulièrement intéressant pour l'analyse des flux de données du fait de son mode de fonctionnement en ligne sans répétition de l'apprentissage. Le grand avantage du modèle ILoNDF au regard du modèle original NDF est lié à sa capacité d'acquérir constamment de nouvelles connaissances relatives aux fréquences d'occurrence des variables et à leurs dépendances de co-occurrence dans les données utilisées pour l'apprentissage, ce qui le rend plus robuste au bruit qui peut être présent dans la description des données d'apprentissage. En outre, le modèle ILoNDF ne comporte aucun paramètre à régler avant ou pendant l'apprentissage, il n'y a donc aucun besoin de faire des calculs supplémentaires et coûteux en matière d'optimisation de paramètres.

Nous avons testé le modèle ILoNDF dans le cadre précis de la classification à partir d'une seule classe. Notre objectif premier était de démontrer le potentiel de notre modification de la règle d'apprentissage du modèle original NDF, en particulier dans le cas du traitement des données fortement multi-dimensionnelles. De même qu'une comparaison approfondie des résultats du modèle ILoNDF à ceux du modèle original NDF, le panel des méthodes d'apprentissage prises en compte dans la comparaison a été étendu pour y inclure des méthodes statistiques multivariés basées sur l'ACP, les réseaux de neurones auto-associatifs, et les SVM monoclasses. Bien que les méthodes précitées ne sont pas adaptées à un fonctionnement en ligne, notre objectif en les comparant au modèle ILoNDF était d'évaluer son apport du point de vue de la qualité de la

classification au regard des méthodes existantes applicables au problème de la classification à partir d'une seule classe dans son cadre le plus général. Cela nous a permis de prouver encore plus clairement la supériorité du modèle ILoNDF vis-à-vis des autres modèles d'apprentissage.

Nous verrons dans le chapitre suivant l'adaptation fine du modèle ILoNDF au cadre du filtrage d'informations en suivant trois types de directions, ayant pour but commun la mise en place d'une stratégie de filtrage orientée-utilisateur. La première direction concerne la modélisation du profil utilisateur par l'intermédiaire de deux modèles de type ILoNDF appris respectivement à partir d'exemples positifs et négatifs du besoin d'informations de l'utilisateur. Cette approche présente l'originalité de reposer sur une analyse non paramétrée du contenu du modèle ILoNDF de manière postérieure à l'apprentissage. Ceci permet de définir une méthode de seuillage basée sur la précision attendue et qui dépend directement du comportement de l'utilisateur et de son type de besoin en informations. La deuxième direction est celui de la détection et du suivi de l'évolution du besoin de l'utilisateur. La troisième direction est de proposer une stratégie de combinaison des deux types de filtrage, filtrage basé sur le contenu et filtrage collaboratif, à l'aide du modèle ILoNDF.

Chapitre 5

Personnalisation et modélisation des profils utilisateurs

La fourniture d'informations à la demande, ou plus généralement le filtrage d'informations, est encore de nos jours un grand challenge dans le cadre d'un environnement dynamique caractérisé à la fois par l'évolution du besoin de l'utilisateur et par la dynamicité du contenu des informations distribuées en flux. La pertinence des informations délivrées, leur opportunité et leur adaptation au besoin spécifique de l'utilisateur constituent des facteurs clés du succès des systèmes de filtrage d'informations. Les problématiques que nous abordons ici, sous le titre d'accès personnalisé aux informations s'organisent principalement en trois volets. D'abord, la modélisation utilisateur à l'aide du modèle d'apprentissage ILoNDF dans le but de mettre en place un système de filtrage d'informations basé sur le contenu qui soit spécifiquement adapté aux spécificités des types de besoin de l'utilisateur, et donc plus performant du point de vue utilisateur (Sections 5.1 et 5.2). Puis, le raffinement des fonctionnalités d'adaptation du modèle utilisateur en le dotant d'une capacité à détecter et à suivre l'évolution du besoin de l'utilisateur en exploitant son retour de pertinence sur les informations filtrées par le système (Section 5.3). Et, enfin, un troisième volet concerne l'amélioration d'un système de distribution ciblée de sites web par satellite en utilisant pour partie les stratégies de modélisation utilisateur précitées et en intégrant de nouvelles fonctionnalités adaptées aux exigences spécifiques du système. Ceci est réalisé dans le cadre du projet européen Sat-N-Surf, en association avec des collaborateurs scientifiques et des partenaires industriels (Section 5.4).

5.1 Le modèle ILoNDF comme modèle utilisateur

5.1.1 Motivation

La modélisation du profil de l'utilisateur est une tâche centrale qui conditionne largement l'efficacité du processus de filtrage. Cette tâche nécessite, d'une part, de représenter les centres d'intérêt de l'utilisateur dans le système, et, d'autre part, d'adapter cette représentation aux changements des centres d'intérêt de l'utilisateur au cours du temps.

Comme nous l'avons vu au chapitre 3, les principaux travaux ayant été menés en matière de filtrage basé sur le contenu s'appuient essentiellement sur des modèles issus de la recherche d'informations auxquels sont ajoutées des fonctions de seuillage. L'apprentissage du profil utilisateur se fait en utilisant le contenu textuel des documents que l'utilisateur a sélectionné comme

exemples de son besoin d'informations. Contrairement à une requête dans le contexte de la recherche d'informations, la richesse des informations contenues dans les documents fournis par l'utilisateur permet de construire une représentation plus riche de son profil. Cependant, la plupart des approches actuelles ne se focalisent que sur l'analyse du contenu des documents pour construire un profil utilisateur le plus souvent sous forme de listes de termes pondérés, analogue à celle des requêtes. Or, à notre avis, pour bénéficier de toute la richesse du contenu des documents, il est nécessaire de moduler l'analyse des documents, de tirer des conclusions sur le type de besoin d'informations de l'utilisateur (ex. précis, thématique, ou exploratoire) et d'évaluer la cohérence des décisions prises par l'utilisateur au regard de la pertinence des documents délivrés par le système. L'intégration de telles connaissances sur le besoin de l'utilisateur dans le système de filtrage l'aide sûrement à délivrer des informations plus proches des attentes réelles de l'utilisateur.

Le mécanisme de modélisation automatique du profil de l'utilisateur que nous proposons est essentiellement basé sur le modèle ILoNDF. Tout en permettant d'apprendre de manière incrémentale une représentation précise du profil et d'adapter cette représentation au changement des centres d'intérêt de l'utilisateur, le modèle ILoNDF a une capacité synthétique à mieux comprendre la nature du besoin d'informations de l'utilisateur. En termes plus concrets, le modèle ILoNDF fournit une nouvelle manière de regarder le profil utilisateur en termes de critères de précision, d'exhaustivité et de contradiction. Ce qui permet, entre autres, d'optimiser le seuil de filtrage tenant compte de l'importance relative que pourrait donner l'utilisateur à la précision et au rappel. Les résultats expérimentaux obtenus sur des corpus standard prouvent l'efficacité du modèle ILoNDF et confirment l'utilité d'adapter les résultats de filtrage en fonction de la connaissance acquise sur le type de besoin de l'utilisateur. Nous reviendrons plus en détail sur ces points dans les sections qui suivent.

5.1.2 Principe

Le principe de la détection de nouveauté est particulièrement intéressant à appliquer à la problématique du filtrage d'informations du fait de l'absence des exemples négatifs dans la majorité des applications réelles (Žižka et al., 2006; Yu et al., 2004). Les raisons en sont multiples, dont quelques exemples sont donnés ci-dessous :

- Beaucoup d'applications recueillent des exemples du besoin de l'utilisateur par l'observation de son comportement ; les documents (ou les liens) visités par l'utilisateur peuvent être choisis en tant qu'exemples positifs du besoin de l'utilisateur alors que les documents (ou les liens) non visités peuvent être choisis en tant qu'exemples négatifs. Or, comme il a été par exemple souligné dans (Schwab et al., 2000), le fait de ne pas visiter certains documents ne signifie pas forcément qu'ils ne sont pas intéressants pour l'utilisateur (ils seront vraisemblablement visités plus tard) et donc de tels documents ne devrait pas justement être considérés comme négatifs ;
- En pratique, il semblerait que les utilisateurs soient très occupés et ne trouvent pas le temps de fournir assez d'exemples de leurs besoins d'informations ; ainsi certains systèmes de filtrage sont conçus sur la base des documents existant dans les favoris de l'utilisateurs (bookmarks) ou des documents collectés sur une longue période de temps qui ne constituent que des exemples positifs du besoin de l'utilisateur (Denis et al., 2003; Žižka et al., 2006) ;
- Pour des raisons de marketing, très souvent les utilisateurs ne sont autorisés à donner que des estimations positives sur les produits, les annonces publicitaires ou les sites web prépayés pour leur inclusion dans les flux (comme c'est le cas du système de distribution

ciblée de sites web par satellite CASABLANCA que nous présenterons plus tard dans la section 5.4 de ce chapitre).

- Plusieurs études ont prouvé que l'apprentissage d'un modèle de classification à partir des exemples positifs uniquement permet d'obtenir de meilleurs résultats que dans le cas d'un fort déséquilibre entre classes (le nombre d'exemples de la classe positive est faible et bien inférieur au nombre d'exemples de la classe négative) (Raskutti et Kowalczyk, 2004; Kassab et Lamirel, 2007).

Dans le contexte du filtrage, le principe de la détection de nouveauté s'applique généralement d'une manière inversée, c.-à-d. ce sont les documents qui sont semblables à un modèle appris par l'intermédiaire d'exemples positifs du besoin de l'utilisateur qui seront choisis et présentés à l'utilisateur. Dans le cas où les exemples du besoin de l'utilisateur sont disponibles en deux types, exemples positifs (choix) et exemples négatifs (rejets), le composant destiné à modéliser le besoin de l'utilisateur est naturellement structuré sous la forme de deux modèles ILoNDF, chacun étant associé à un type d'exemples. Le rôle de chaque modèle est donc d'assurer le traitement des exemples associés à son type, de manière à en extraire les caractéristiques synthétiques en termes d'habituation et de nouveauté. L'interaction entre le système et l'utilisateur peut être vue comme une suite d'étapes. À chaque étape, le système fournit à l'utilisateur un ensemble de documents qui doivent être jugés par ce dernier. L'ensemble des documents ayant fait l'objet d'un même jugement par l'utilisateur (choix ou rejet) peut ainsi être considéré comme un nouvel ensemble de données d'apprentissage destinées à alimenter un modèle ILoNDF spécifique à la catégorie de jugement concernée. À l'issue de l'apprentissage, le contenu de chaque modèle ILoNDF sera représentatif des caractéristiques de l'ensemble des documents ayant bénéficié d'un même jugement par l'utilisateur. À partir des deux modèles ILoNDF construits, deux valeurs, S_P et S_N, peuvent être obtenues pour chaque nouveau document d en utilisant l'une des méthodes présentées dans la section 4.4, à savoir la méthode de projection directe (DPM), la synthèse élémentaire de l'apprentissage (V-PM) et la méthode combinatoire (CS). La première valeur, S_P, reflète le degré de similarité entre le document d et les exemples positifs du besoin de l'utilisateur, alors que la valeur S_N reflète le degré de similarité entre le document d et les exemples négatifs du besoin de l'utilisateur, c.-à-d., le degré de dissimilarité entre le document et le besoin de l'utilisateur. Pour calculer le score de pertinence de chaque nouveau document, une combinaison des valeurs s'effectue en utilisant la fonction suivante :

$$S = \alpha.S_P - \gamma.S_N \tag{5.1}$$

où α et γ sont des paramètres positifs contrôlant respectivement l'importance des exemples positifs et des exemples négatifs. Un document est délivré à l'utilisateur lorsque le score de pertinence qui lui correspond est supérieur à un seuil prédéterminé.

Les avantages potentiels associés à l'utilisation du modèle ILoNDF pour le filtrage d'informations, se situent à plusieurs niveaux. Ils peuvent être brièvement résumés dans les quatre points suivants :

- Il permet d'apprendre incrémentalement le profil de l'utilisateur en opérant un traitement cumulatif des exemples fournis par l'utilisateur au cours de son interaction avec le système de filtrage d'informations. L'apprentissage incrémental du profil utilisateur présente un intérêt particulier dans le cas du filtrage adaptatif pour lequel on ne dispose pas (ou très peu) d'exemples du besoin de l'utilisateur au démarrage du processus du filtrage, et la méthode d'apprentissage doit ainsi permettre de faire évoluer le profil utilisateur au fur et à mesure que sont fournis des nouveaux exemples au système de filtrage ;

- Typiquement, l'apprentissage du modèle ILoNDF ne requiert que des exemples positifs (ou négatifs) pour modéliser le besoin d'informations de l'utilisateur ; et dans ce cas, comme nous avons déjà pu le montrer dans le chapitre 4, le modèle ILoNDF offre une performance supérieure à celle des méthodes typiques de la classification à partir d'une seule classe. Lorsque le besoin de l'utilisateur est exprimé à travers des exemples positifs et négatifs, la modélisation du profil utilisateur peut aussi être réalisée simplement et efficacement en combinant les résultats d'apprentissage de deux modèles ILoNDF, chacun étant associé à un type d'exemples du besoin de l'utilisateur ;
- La capacité d'oubli du modèle ILoNDF permet de dépister le changement des centres d'intérêt de l'utilisateur au cours du temps. En effet, si l'on adopte un point de vue centré utilisateur, ce dernier doit pouvoir changer les jugements qu'il a rendus antérieurement concernant ses centres d'intérêt, ce qui exige une aptitude à pouvoir annuler l'effet des jugements périmés de la représentation du profil de l'utilisateur.
- Il permet d'évaluer le type de besoin de l'utilisateur à partir des mesures de précision, d'exhaustivité, et de contradiction de ses jugements portant sur la pertinence ou la non pertinence des documents qui lui sont fournis par le système, c.-à-d. les exemples positifs et négatifs du besoin de l'utilisateur. Ce type d'évaluation est de prime abord indispensable pour la mise en place d'une stratégie de filtrage adaptée à chaque type de besoin exprimé par l'utilisateur. L'évaluation du type de besoin de l'utilisateur se révèle également efficace pour améliorer de manière indirecte les résultats bruts du filtrage en exploitant une optimisation du seuil de filtrage basée sur la mesure F_β.

Bien que l'un ou l'autre des avantages qui viennent d'être mentionnés puisse être commun à une minorité des méthodes existantes, le modèle ILoNDF est, à notre connaissance, le seul à pouvoir rassembler l'ensemble de ces avantages qui constituent des éléments clés pour la conception d'un système de filtrage dans une perspective d'évolutivité, de dynamicité, et d'orientation utilisateur. Dans la suite de cette section, nous examinons en détail le fonctionnement de base du modèle ILoNDF comme modèle utilisateur dans le contexte du filtrage d'informations. L'évaluation et l'intégration du type de besoin de l'utilisateur seront détaillées dans la section 5.2. Nous aborderons ensuite l'utilisation du modèle ILoNDF en présence d'un changement des centres d'intérêt de l'utilisateur dans la section 5.3.

5.1.3 Évaluation

Nous rapportons ici les résultats de nombreuses expérimentations que nous avons menées pour évaluer l'application du modèle ILoNDF à la représentation du profil utilisateur. Dans ce cadre et à fins de comparaison, nous étendons le panel des méthodes d'apprentissage prises en compte dans l'évaluation en y incluant les méthodes de type Rocchio et SVM. En fait, la méthode de Rocchio est très souvent prise comme base de comparaison de la performance des autres méthodes candidates (cf. à titre d'exemples (Ault et Yang, 2001), (Debole et Sebastiani, , 2005) et (Dumais et al., 1998)). Elle s'est également révélée très performante à condition de bien la paramétrer (cf. Section 3.3.2). Les machines à vecteurs supports (SVM) ont de plus été retenues pour leurs performances reconnues pour les tâches de filtrage et de classification (Dumais et al., 1998).

Les expérimentations sont organisées en deux parties : Tout d'abord, nous évaluons l'impact du choix des paramètres sur la performance des différentes méthodes et nous déterminons les valeurs optimales de ces paramètres pour en tirer les meilleures performances. D'autres expérimentations sont ensuite menées pour comparer directement l'efficacité du modèle ILoNDF

à celle des autres méthodes (Rocchio et SVM). Les expérimentations sont réalisées sur deux sous-ensembles du corpus Reuters-21578 : 1) *Reuters10* correspondant aux 10 catégories les plus fréquentes ; et 2) *Reuters90* correspondant aux 90 catégories les plus fréquentes. Le choix de ces sous-ensembles a été fait de façon à couvrir le plus de caractéristiques possibles des documents textuels soumis à des fins de filtrage ou de classification (cf. Annexe A). Chaque catégorie du corpus Reuters a été utilisée pour simuler un besoin spécifique à un utilisateur : les documents d'apprentissage associés à chaque catégorie ont été utilisés en tant qu'exemples positifs du besoin de l'utilisateur. Généralement, les documents associés aux autres catégories peuvent être utilisés en tant qu'exemples négatifs du besoin de l'utilisateur. Mais, en raison de leur nombre relativement plus important et des fins d'optimisation de performance, nous avons opéré un choix plus sélectif des exemples négatifs en appliquant la méthode de "*Query Zoning*" (cf. Section 3.3.2).

Importance du réglage des paramètres

L'exploitation conjointe des exemples positifs et négatifs du besoin de l'utilisateur, quelle soit à travers le modèle ILoNDF (cf. Eq. 5.1) ou la méthode de Rocchio (cf. Eq. 3.15), fait intervenir deux paramètres, α et γ, ajustant respectivement les contributions relatives des exemples positifs et négatifs. Il est clair que si l'on divise ces paramètres par un même facteur il n'y aura pas d'impact sur la performance des méthodes précitées (cela induit seulement un décalage de la valeur du seuil de filtrage). On peut donc fixer la valeur d'un paramètre, disons α, à 1 et étudier l'influence d'un seul paramètre, $\eta = \gamma/\alpha$, sur la performance de ces méthodes.

Dans la littérature, les valeurs standard définies pour les paramètres de Rocchio sont telles que α=16 et γ=4 ($\eta = 0.25$) ; Cependant, Singhal et al. (1997) ont trouvé que le choix de α=γ peut être adéquat dans le cas où les exemples négatifs sont sélectionnés selon la méthode de "*Query Zoning*" (QZ) que nous utilisons dans nos expérimentations. Mais est-ce que, effectivement, ce sont des valeurs optimales ? et est-ce qu'elles le seront aussi pour le modèle ILoNDF ? Pour explorer la sensibilité de la méthode Rocchio et du modèle ILoNDF à ces paramètres, nous avons testé le fonctionnement de Rocchio et ILoNDF pour différentes valeurs de η. Les résultats obtenus sont respectivement présentés sur les figures 5.1 et 5.2.

D'après les résultats qu'illustre la figure 5.1, la valeur optimale de η en termes de performance moyenne de toutes les catégories est de 0.5 sur le corpus *Reuters10* et de 0.25 sur le corpus *Reuters90*. Les valeurs optimales trouvées dans nos expériences sont donc différentes de celle suggérée par Singhal et al. (1997), mais il est vrai que le choix d'une valeur supérieure à 1 pour le paramètre η n'apporte pas de gain de performance lors de l'utilisation de la méthode QZ. De manière générale, l'analyse des résultats de Rocchio sur nos deux corpus de test, *Reuters10* et *Reuters90*, permet de constater qu'en augmentant la valeur de η, la performance s'améliore progressivement jusqu'à atteindre sa valeur maximale et diminue ensuite en dessous de cette valeur. Ce comportement est attendu et s'explique par le fait que plus η augmente plus de termes non pertinents sont exclus de la représentation du profil utilisateur mais à partir d'une certaine valeur du paramètre η, des termes pertinents peuvent aussi être éliminés. Par ailleurs, l'examen des courbes individuelles tracées pour chacune des catégories fait ressortir deux points importants à cet égard méritant d'être soulignés : 1) la valeur optimale de η pour certaines catégories peut être différente de celle évaluée en termes de performance moyenne de toutes les catégories. 2) l'exploitation des exemples négatifs, dans le cas de certaines catégories, souvent celles peu fréquentes, n'apporte aucune amélioration mais au contraire affecte négativement la performance.

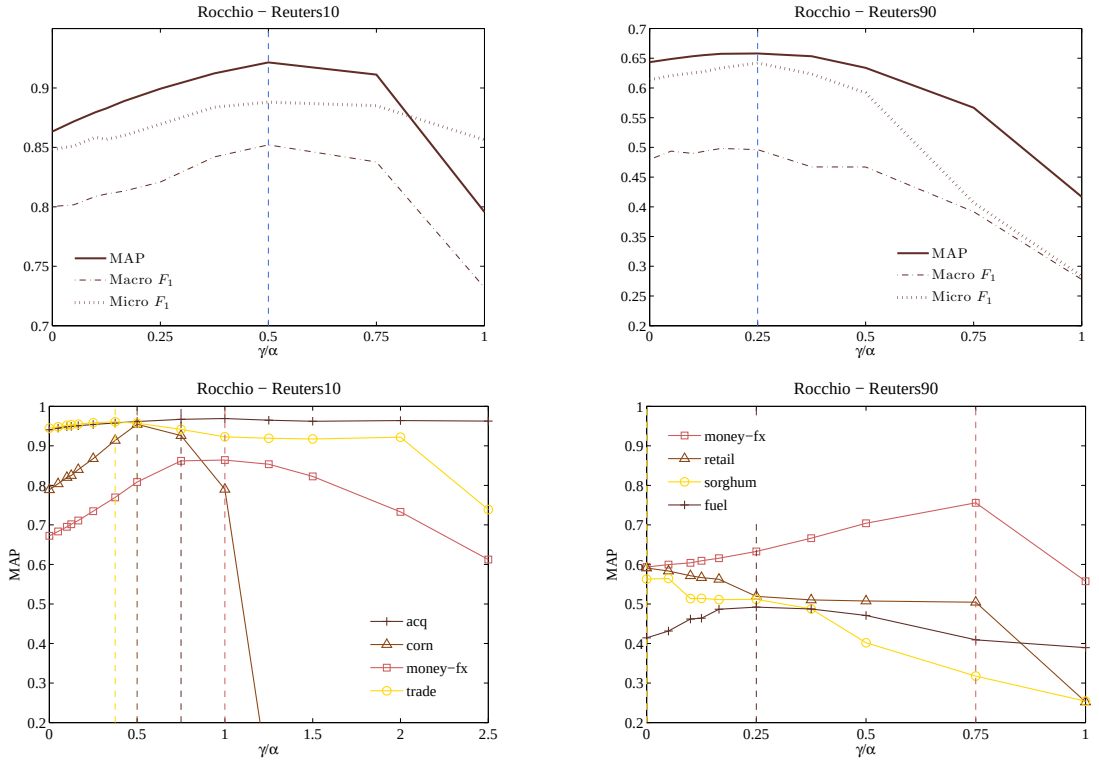


FIG. 5.1 – Influence des paramètres α et γ de la méthode Rocchio en termes de performance moyenne de toutes les catégories du corpus Reuters10 et du corpus Reuters90 (en haut) et quelques-unes des courbes MAP obtenues pour quelques catégories de ces corpus (en bas). Les catégories représentées et les nombres de documents (apprentissage/test) associés sont : acq (1650/719), corn(181/56), money-fx (538/179) et trade (369/117) sur le corpus *Reuters10* ; et money-fx (538/179), retail (23/2), sorghum (24/10) et fuel (13/10) sur le corpus *Reuters90*.

Comme pour la méthode de Rocchio, nous avons étudié le fonctionnement du modèle ILoNDF en variant la valeur du paramètre $\eta = \gamma/\alpha$. L'examen des résultats présentés dans la figure 5.2 conduit à des observations semblables à celles que nous venons de faire à propos de la méthode de Rocchio. Les valeurs optimales évaluées pour le modèle ILoNDF en termes de performance moyenne sont également de 0.5 sur le corpus *Reuters10* et de 0.25 sur le corpus *Reuters90*, cependant, il est possible de noter parfois une différence entre ILoNDF et Rocchio au niveau des valeurs optimales du paramètre η obtenues séparément pour les catégories des corpus étudiés (voir, par exemple, le cas de la catégorie "money-fx" sur les figures 5.1 et 5.2). À ce sujet, il est important de souligner le fait que cette concordance relativement forte entre Rocchio et ILoNDF au regard du choix de la valeur optimale du paramètre η n'est constatée que lors de l'utilisation de la méthode QZ. En fait, nous avons trouvé que l'utilisation de la méthode QZ conduit à un intervalle plus petit de valeurs optimales du paramètre η pour la méthode Rocchio. Par exemple, lors de l'utilisation de la totalité des exemples négatifs, les valeurs optimales de η ont été trouvées dans l'intervalle $[0, 25]$ (au lieu de $[0, 1]$) sur le corpus *Reuters10*, et dans l'intervalle $[0, 20]$ (au lieu de $[0, 1]$) sur le corpus *Reuters90*. La figure 5.3 montre la différence entre l'utilisation ou non de la méthode QZ sur deux catégories du corpus *Reuters10*. Cette figure souligne deux points : 1) La valeur optimale du paramètre η peut être beaucoup plus grande (cas de la catégorie "money-

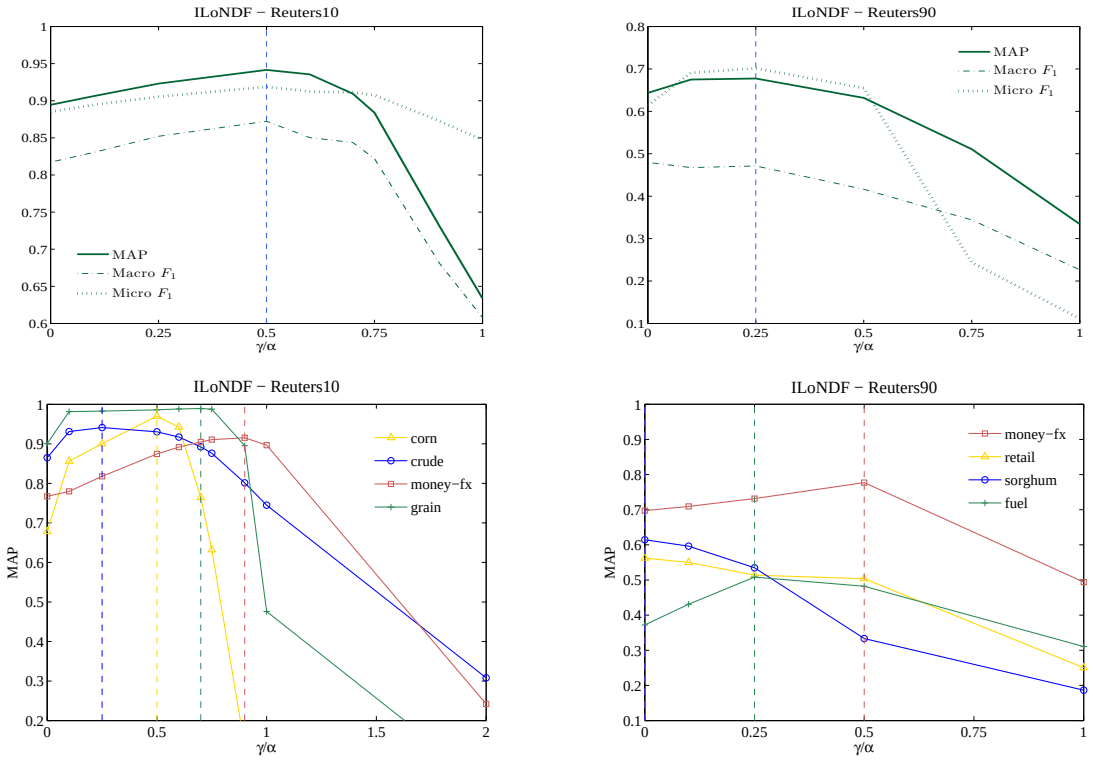


FIG. 5.2 – Influence des paramètres α et γ du modèle ILoNDF en termes de performance moyenne de toutes les catégories du corpus Reuters10 et du corpus Reuters90 (en haut) et quelques-unes des courbes MAP obtenues pour quelques catégories de ces corpus (en bas). Les catégories représentées et les nombres de documents (apprentissage/test) associés sont : corn (181/56), crude (389/189), money-fx (538/179) et grain (433/149) sur le corpus *Reuters10* ; et money-fx (538/179), retail (23/2), sorghum (24/10) et fuel (13/10) sur le corpus *Reuters90*.

fx") ou plus petite (cas de la catégorie "trade") de celle obtenue lors de l'utilisation de la méthode QZ. 2) Contrairement à ce que l'on peut tirer des travaux antérieurs comme (Schapire et al., 1998), l'amélioration de la performance grâce à l'application de la méthode QZ n'est substantielle que pour une même valeur du paramètre η (par exemple, pour η=1 dans le cas de la catégorie "money-fx"). En ce qui concerne le modèle ILoNDF, le choix de la valeur optimale du paramètre η ne semble pas dépendre de l'application de la méthode QZ (cf. Figure 5.3). Les valeurs optimales obtenues sur les corpus *Reuters10* et *Reuters90* sont généralement comprises entre 0 et 1.

Du fait que la valeur optimale obtenue en termes de performance moyenne est différente de celle obtenue séparément pour chacune des catégories, nous pouvons conclure que la pratique de spécifier une seule valeur pour toutes les catégories peut constituer une solution satisfaisante mais non optimale. Dans cette perspective, une amélioration de la performance du modèle ILoNDF pourrait être obtenue en exploitant une approche par validation croisée comme celle proposée par Moschitti (2003) pour la méthode de Rocchio. L'idée est d'estimer la valeur optimale individuellement pour chaque catégorie sur un ensemble de validation en augmentant progressivement la valeur du paramètre η par un facteur ϵ jusqu'à atteindre la valeur qui maximise un critère de performance ou jusqu'à une valeur maximale (η_{max}). Lorsqu'elle est appliquée à la méthode

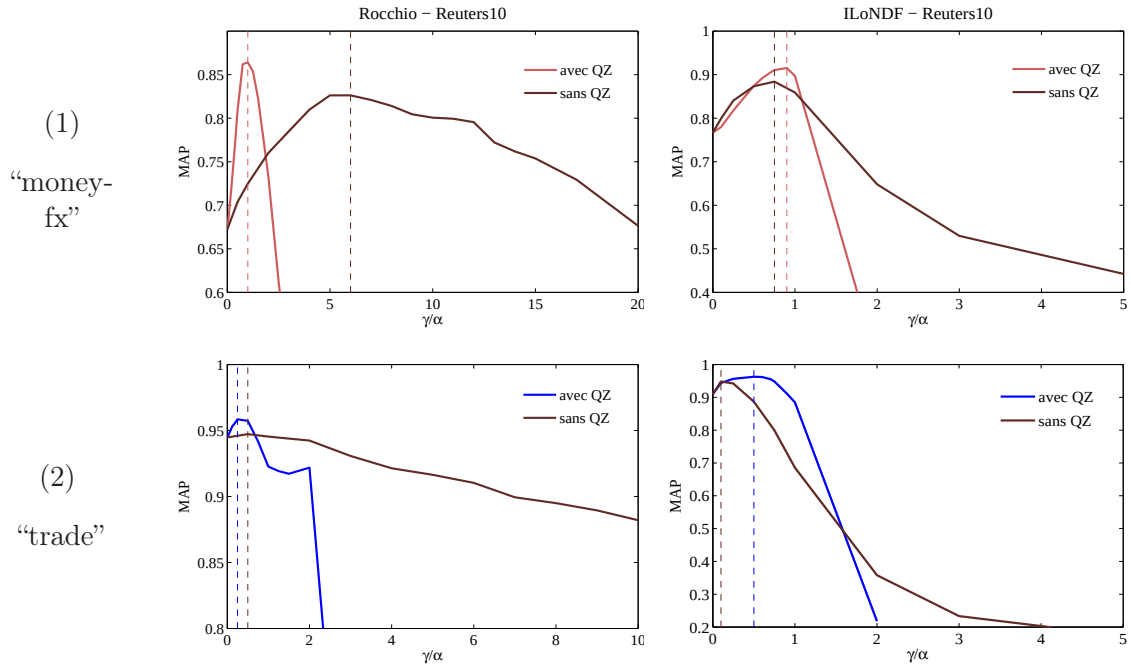


FIG. 5.3 – Illustration de l'effet de l'utilisation ou non de la méthode "Query Zoning" (QZ) sur le choix de la valeur optimale du paramètre η de la méthode Rocchio (gauche) et du modèle ILoNDF (droite). Deux exemples sont présentés pour les catégories "money-fx" et "trade" du corpus *Reuters10*.

Rocchio, le principal défaut de cette approche tient au fait que l'estimation du paramètre s'est avérée très coûteuse en temps de calcul. La raison en est que la valeur optimale de η peut être comprise dans un intervalle allant de 0 à une valeur relativement forte η_{max} (η_{max} est mise à 30 dans (Moschitti, 2003)). Or, comme nous venons de le voir, l'utilisation de la méthode QZ conduit à un intervalle plus petit de valeurs optimales de η, ce qui rend le choix de la valeur optimale du paramètre η par validation croisée moins coûteux à la fois pour Rocchio et ILoNDF.

Pour évaluer le degré auquel la performance pourrait être améliorée par un choix judicieux du paramètre η, nous avons testé l'optimisation de la valeur du paramètre η sur un ensemble de validation dans l'intervalle [0,1] avec un pas ϵ=0.25. La figure 5.4 (1) montre le gain de performance obtenu par optimisation locale (par catégorie) du paramètre η pour la méthode Rocchio sur les corpus *Reuters10* et *Reuters90*. Le gain de performance en termes de MAP est de +0.83% sur le corpus *Reuters10* et de +3.31% sur le corpus *Reuters90*. De plus, la différence entre macro- et micro-moyennes des scores F_1 qui est due à la faible performance de la méthode sur les petites catégories du corpus *Reuters90* est moins prononcée pour une valeur optimale du paramètre η trouvée localement que globalement.

La figure 5.4 (2) montre le gain de performance obtenu par optimisation locale (par catégorie) du paramètre η sur les corpus *Reuters10* et *Reuters90* dans le cas du modèle ILoNDF. Le gain de performance en termes de MAP est de +0.42% sur le corpus *Reuters10* et de +2.31% sur le corpus *Reuters90*. Le gain de performance en termes de macro- et micro-moyennes des scores F_1 est important sur le corpus *Reuters10* mais moins apparent sur le corpus *Reuters90*. La perte

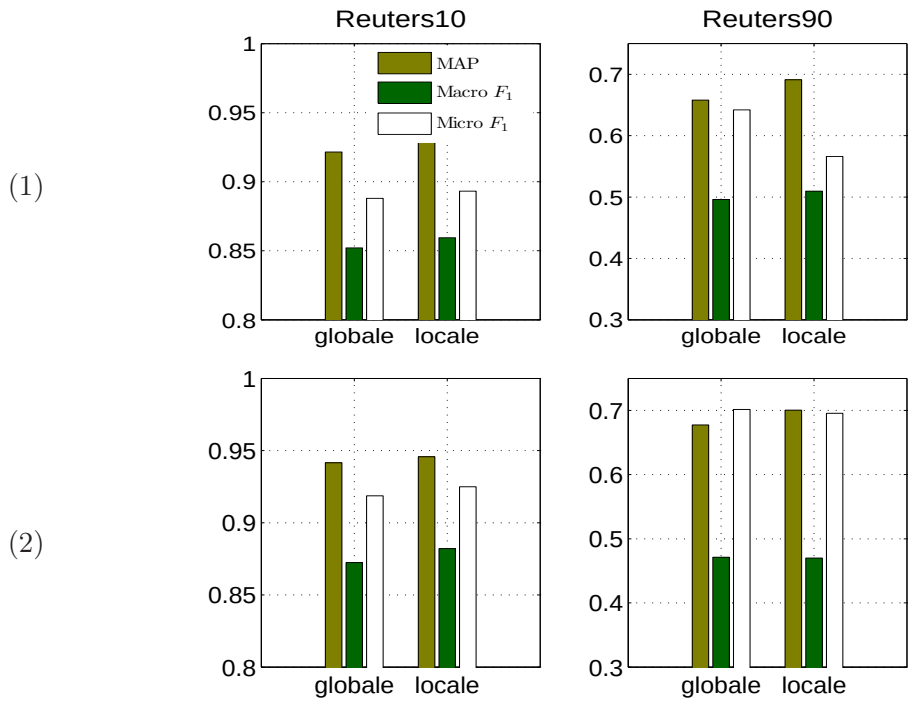


FIG. 5.4 – L'effet du réglage du paramètre η (1) de la méthode Rocchio et (2) du modèle ILoNDF : Comparaison entre l'optimisation globale du paramètre en termes de performance moyenne sur toutes les catégories et l'optimisation locale du paramètre en termes de performance individuelle sur chacune des catégories.

de performance du modèle ILoNDF en termes de macro-moyennes des scores F_1 sur le corpus *Reuters90* est liée à l'utilisation de la méthode CS pour le calcul des scores de pertinence des documents. Comme il ressort du tableau 5.1, V-PM fournit de meilleures performances en termes de MAP et macro-moyennes de F_1 sur le corpus *Reuters90*. Cela signifie que V-PM pourrait être plus précise que DPM et CS dans le cas des petites catégories, et que nous aurions dû prendre en compte le nombre de données disponibles pour faire l'apprentissage lors de la mise en œuvre de notre méthode de combinaison.

Nous nous tournons maintenant vers l'analyse de l'impact des paramètres de la méthode SVM. Pour nos expérimentations, nous utilisons l'implémentation SVM^{light}[33] développée par Joachims (1999). Nous retenons le noyau linéaire avec les valeurs par défaut de tous les paramètres sauf deux. Le premier est le paramètre de régularisation C qui ajuste le compromis entre maximisation de la marge et minimisation de l'erreur d'apprentissage (cf. Section 1.4.5). Le second paramètre J est un facteur de coût qui permet de considérer deux sortes d'erreurs séparément, les erreurs sur les exemples positifs d'apprentissage et les erreurs sur les exemples négatifs d'apprentissage, l'objectif étant de réduire la sensibilité de la méthode au déséquilibre entre ces deux types d'exemples. Pour ce dernier paramètre, nous prenons systématiquement la valeur proposée par Morik et al. (1999) qui correspond au rapport entre le nombre d'exemples négatifs et le nombre d'exemples positifs. Lorsque nous utilisons ce paramètre, nous le signalons en précisant la méthode SVM par le nom CS-SVM (Cost-Sensitive SVM).

[33] http ://svmlight.joachims.org/

TAB. 5.1 – Comparaison des méthodes de calcul des scores de pertinence des documents par le modèle ILoNDF sur les corpus *Reuters10* et *Reuters90*.

			Macro			Micro		
		MAP	Précision	Rappel	F_1	Précision	Rappel	F_1
Reuters10	DPM	0.9432	0.8919	0.8814	0.8856	0.9262	0.9228	0.9245
	V-PM	0.9335	0.8767	0.8562	0.8648	0.92	0.8999	0.9098
	CS	0.9415	0.9072	0.8423	0.8724	0.9392	0.8992	0.9187
Reuters90	DPM	0.6546	0.5644	0.5115	0.4588	0.5751	0.7802	0.6621
	V-PM	0.6813	0.5918	0.5739	0.5110	0.5950	0.7906	0.6790
	CS	0.6775	0.5940	0.4928	0.4712	0.6479	0.7649	0.7016

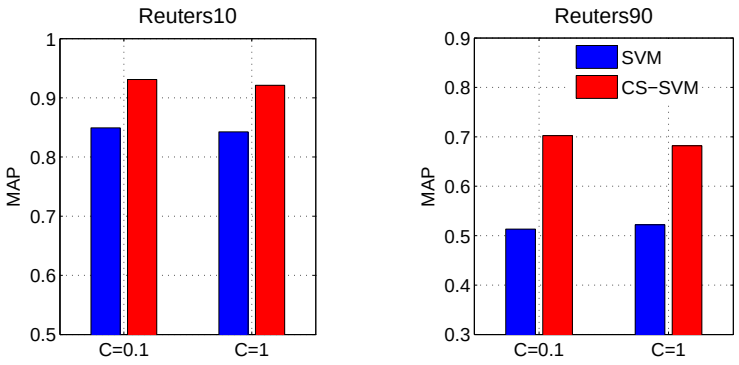


FIG. 5.5 – L'effet de la prise en compte du facteur du coût sur la performance de la méthode SVM. La performance est rapportée en termes de MAP avec (CS-SVM) et sans (SVM) l'utilisation du facteur de coût et pour deux valeurs différentes du paramètre C.

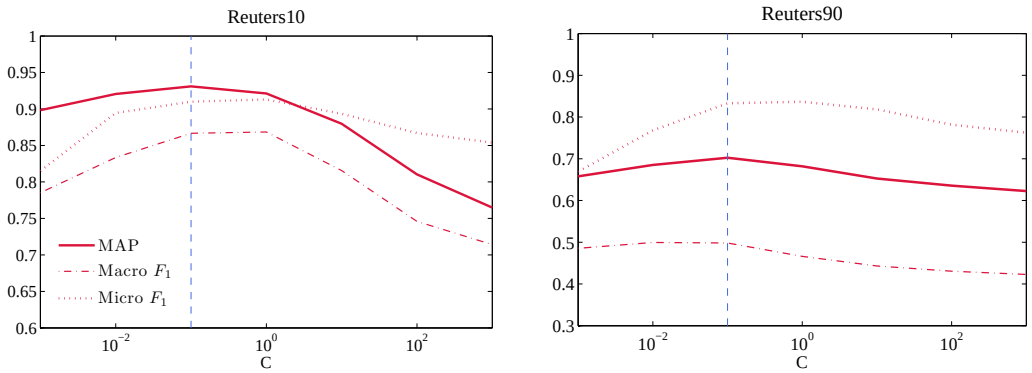


FIG. 5.6 – L'influence du choix du paramètre C sur la performance de la méthode CS-SVM.

TAB. 5.2 – Comparaison des moyennes des scores MAP, Précision, Rappel et F_1 obtenus pour le modèle ILoNDF et les méthodes Rocchio et SVM.

		Macro			Micro		
	MAP	Précision	Rappel	F_1	Précision	Rappel	F_1
Reuters10							
Rocchio (Opt.locale)	0.9298	0.8767	0.8463	0.8595	0.9208	0.8671	0.8931
(Opt.globale)	0.9215	0.8810	0.8290	0.8521	0.9160	0.8617	0.8880
ILoNDF (Opt.locale)	**0.9457**	0.9129	0.8564	**0.8821**	0.9419	0.9086	**0.9250**
(Opt.globale)	0.9415	0.9072	0.8423	0.8724	0.9392	0.8992	0.9187
SVM	0.8424	0.7665	0.5077	0.5839	0.9501	0.7903	0.8629
CS-SVM	0.9310	0.8079	0.9414	0.8669	0.8686	0.9556	0.9100
Reuters90							
Rocchio (Opt.locale)	0.6911	0.5978	0.5903	0.5096	0.4389	0.7975	0.5662
(Opt.globale)	0.6580	0.5836	0.5538	0.4962	0.5558	0.7598	0.6420
ILoNDF (Opt.locale)	0.7006	0.5906	0.4930	0.4701	0.6366	0.7665	0.6955
(Opt.globale)	0.6775	0.5940	0.4928	0.4712	0.6479	0.7649	0.7016
SVM	0.5220	0.3239	0.1715	0.2102	0.9214	0.6178	0.7397
CS-SVM	**0.7024**	0.5810	0.4847	**0.4982**	0.7888	0.8821	**0.8328**

Tout d'abord, nous illustrons à travers la figure 5.5 l'importance de la prise en compte du facteur de coût pour une meilleure performance de la méthode SVM. Sur cette figure, on peut voir que la performance de CS-SVM est bien supérieure à celle de SVM sur nos deux corpus de test, *Reuters10* et *Reuters90*. Cette supériorité de CS-SVM est spécialement accentuée sur le corpus *Reuters90* en raison du déséquilibre relativement important existant entre les catégories de ce corpus. C'est pour cette raison que nous utilisons CS-SVM pour la suite de nos expérimentations.

Comme deuxième étape, nous avons cherché à optimiser le paramètre C dans l'intervalle $[10^{-3}, 10^{+3}]$. La figure 5.6 trace l'évolution de la performance de la méthode SVM en fonction du paramètre C. Les résultats indiquent que la performance de SVM dépend sensiblement de la valeur du paramètre C, en particulier dans le cas du corpus *Reuters10*. La valeur optimale du paramètre C pour les deux corpus de test, *Reuters10* et *Reuters90*, est de 0.1. Une observation intéressante au sujet de la valeur optimale du paramètre C est que celle-ci, une fois trouvée ou estimée, semble être commune à toutes les catégories individuelles des corpus. Donc, contrairement à la méthode Rocchio et au modèle ILoNDF, l'optimisation locale de la valeur du paramètre C n'apporterait généralement pas de gain de performance.

Évaluation comparative de la qualité de l'apprentissage

Maintenant que nous avons trouvé le meilleur paramétrage, il est possible d'établir une comparaison directe entre les différentes méthodes étudiées : Rocchio, ILoNDF, et SVM. Les résultats en termes de performance moyenne sont présentés dans le tableau 5.2. La figure 5.7 montre les scores MAP sur les différentes catégories des corpus *Reuters10* et *Reuters90*.

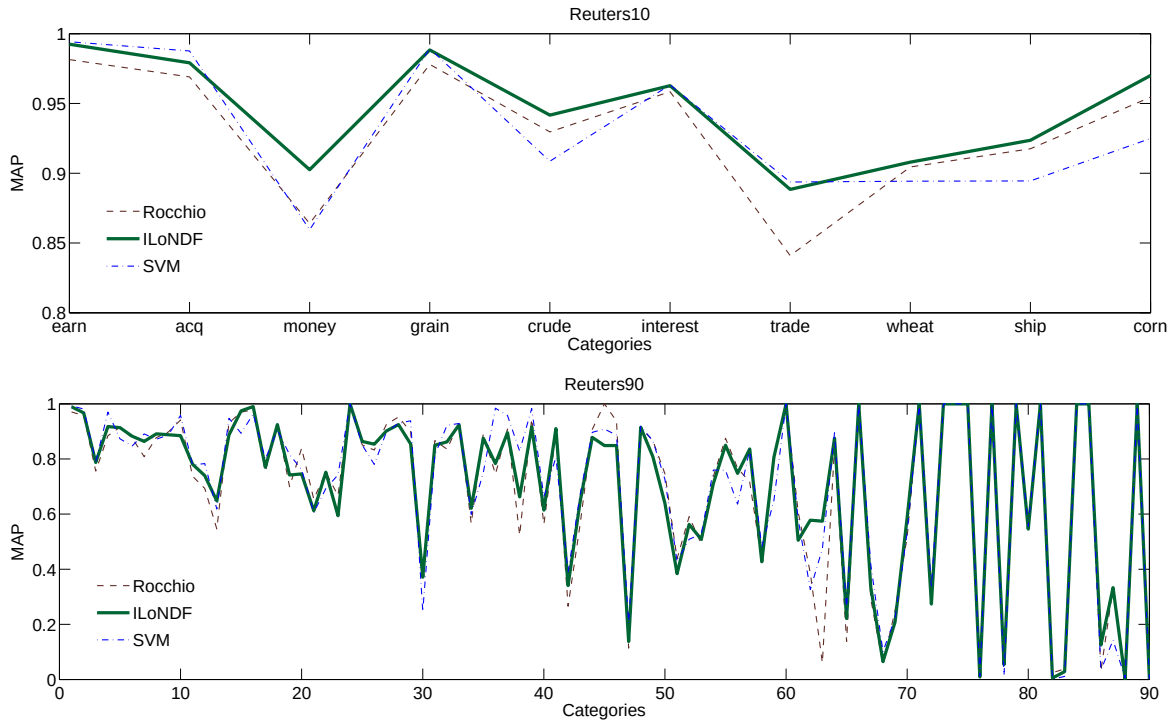


FIG. 5.7 – Comparaison de performance des méthodes Rocchio, ILoNDF, et SVM sur les différentes catégories des corpus *Reuters10* et *Reuters90*. Les catégories sont classées par ordre décroissant de leur taille.

Les résultats montrent qu'il n'existe pas de différence significative entre les valeurs moyennes obtenues pour les différentes méthodes. Le modèle ILoNDF est meilleur que les autres méthodes sur le corpus *Reuters10*, mais il est moins bon que SVM sur le corpus *Reuters90*. Comme nous l'avons démontré précédemment, cette perte de performance tient en partie à la méthode CS adoptée pour le calcul des scores de classification des documents appartenant aux petites catégories, mais aussi à la qualité de représentation des documents du corpus *Reuters90*. En fait, notre sélection d'un nombre égal de termes par catégorie risque d'introduire beaucoup de bruit dans la représentation des documents appartenant aux grandes catégories (termes d'indexation non discriminatifs), cf. Annexe A. Il est donc préférable de sélectionner un nombre de termes proportionnel au nombre de documents associés à chaque catégorie lors d'un fort déséquilibre entre catégories. Notons aussi que dans le cas précis du filtrage, il est préférable de se focaliser sur la performance des méthodes sur les catégories de petite ou moyenne taille, du fait que le nombre d'exemples fournis par l'utilisateur ne dépasse pas un nombre bien limité de documents.

Pour conclure

Au terme de cette évaluation, nous avons pu mettre en évidence deux points essentiels relatifs au fonctionnement du modèle ILoNDF :

- La performance du modèle ILoNDF est particulièrement sensible au choix de la valeur du paramètre η. Néanmoins, la valeur optimale peut facilement être obtenue par validation croisée et est généralement comprise dans l'intervalle $[0, 1]$;
- Le modèle ILoNDF peut réaliser une bonne performance comparable à celle des meilleures

méthodes (SVM), avec cependant l'avantage d'un apprentissage incrémental et la possibilité de la modélisation du profil utilisateur à partir d'exemples positifs uniquement.

Dans les deux sections suivantes, nous aborderons des capacités propres au modèle ILoNDF qui suscitent davantage son potentiel d'utilisation comme modèle utilisateur.

5.2 Analyse synthétique du besoin de l'utilisateur

L'analyse fine et la compréhension approfondie du contenu des documents fournis par l'utilisateur en tant qu'exemples positifs et/ou négatifs de son besoin spécifique en informations est un facteur clé de la réussite de tout système intelligent de filtrage d'informations. Ce type d'analyse peut aider à apporter des indications purement objectives sur les attentes réelles de l'utilisateur, qui se révèlent, à leur tour, utiles pour la mise en place d'un système de filtrage adapté aux caractères spécifiques de l'utilisateur et aux différents types de ses besoins en informations. Cependant, la majorité des systèmes actuels sont plutôt conçus dans le seul but d'assurer un fonctionnement optimal en termes de critères d'évaluation de la pertinence du point du vue système, tels que la précision et le rappel (cf. Section 1.5.1). Bien qu'une telle fonctionnalité constitue une des caractéristiques principales que devraient satisfaire tous les systèmes de filtrage pour répondre efficacement aux besoins des utilisateurs, d'autres caractéristiques importantes devraient également être recherchées pour une meilleure satisfaction des utilisateurs. Le point de vue adopté dans notre travail est que le type de besoin de l'utilisateur peut aller d'un besoin très précis — lorsque l'utilisateur connaît parfaitement ce qu'il peut attendre du système — à un besoin très vague ou exploratoire — lorsque l'utilisateur ne possède pas un besoin clairement défini —. Le type de besoin peut aussi être qualifié de contradictoire, ou encore d'ambigu, lorsque la similarité entre les exemples positifs et négatifs que choisit l'utilisateur pour décrire son besoin est forte. Par conséquent, un système de filtrage devrait pouvoir caractériser les différents types de besoin d'informations de l'utilisateur et les considérer comme faisant partie intégrante du processus de filtrage.

Un des aspects les plus importants du modèle ILoNDF est sa capacité à capturer les relations de co-occurrence existant entre les termes des documents utilisés pour faire l'apprentissage (cf. Section 4.3). Pour des fins de clarté d'illustration, nous prendrons ici un exemple simple. Supposons que nous avons deux documents, d_1 et d_2, fournis comme exemples positifs du besoin d'un utilisateur et représentés dans un espace à 5 termes :

	t_1	t_2	t_3	t_4	t_5
d_1	1	0	1	0	0
d_2	0	1	1	0	0

Après l'apprentissage du modèle ILoNDF sur d_1, la fonction de transfert est :

$$\Phi_1 = \begin{pmatrix} 0.5 & 0 & -0.5 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ -0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

La valeur négative dans la matrice ci-dessus indique une dépendance de l'occurrence des termes t_1 et t_3. En poursuivant maintenant l'apprentissage du modèle ILoNDF sur le document d_2, on obtient :

$$\Phi_2 = \begin{pmatrix} 1.46 & 0.15 & -0.38 & 0 & 0 \\ 0.15 & 1.38 & -0.46 & 0 & 0 \\ -0.38 & -0.46 & 1.15 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

De même, la valeur de l'élément Φ_{23} (et également celle de Φ_{32} du fait de la symétrie) a diminué de 0 à -0.46, ce qui montre la dépendance d'occurrence entre les termes t_2 et t_3.

Maintenant, si l'on considère conjointement les matrices ci-dessus, on peut constater qu'après l'apprentissage sur d_2, la valeur de l'élément Φ_{13} a augmenté de -0.5 à -0.38, faisant moins dépendants les termes t_1 et t_3 puisqu'ils ne sont pas apparus ensemble dans d_2. En outre, la valeur de l'élément Φ_{12} a augmenté de 0 à 0.15, ce qui indique l'indépendance entre les termes t_1 et t_2 puiqu'ils sont apparus séparément avec t_3 mais jamais ensemble. Ainsi, on peut directement affirmer suite à l'examen de la matrice Φ_2 que le terme t_3 est apparu avec les termes t_1 et t_2, tandis que ces deux derniers sont indépendants dans les documents utilisés pour l'apprentissage, d_1 et d_2.

Dans les sous-sections suivantes, nous montrons comment évaluer le type de besoin de l'utilisateur à l'aide des relations de co-occurrence fournies par le modèle ILoNDF. En premier lieu, nous mettrons à profit ces relations pour définir des mesures de dépendance/indépendance entre les termes présents dans les documents utilisés pour l'apprentissage (c.-à-d. les exemples positifs ou négatifs du besoin de l'utilisateur). Ces mesures nous permettent ensuite d'identifier les termes corrélés à la représentation du besoin de l'utilisateur et ceux qui ne le sont pas. Sur la base de ces connaissances, nous proposons trois critères purement objectifs pour caractériser le type de besoin de l'utilisateur en termes de précision, diversité et contradiction. Enfin, une stratégie de seuillage basée sur la précision attendue par l'utilisateur du système et qui dépend directement du type de besoin de l'utilisateur est mise en place.

5.2.1 Analyse de la pertinence des termes

En vertu de l'analyse de l'apprentissage du modèle ILoNDF à l'aide de l'exemple précédent, le degré de dépendance d'un terme i par rapport à l'ensemble des termes utilisés dans la représentation des documents, $\mathcal{T}$, peut être défini de la manière suivante :

$$Dep(i/T) =\mid \{\Phi_{ij};\ \ \Phi_{ij} < 0, j \in T, i \neq j\} \mid$$

De cette manière, le degré de dépendance représente le nombre de termes dans l'ensemble T qui ont une relation de dépendance avec i. D'une manière analogue, le degré d'indépendance s'exprime par :

$$InDep(i/T) =\mid \{\Phi_{ij};\ \ \Phi_{ij} > 0, j \in T, i \neq j\} \mid$$

Nous pouvons maintenant diviser les termes appris $T_L \subset T$ en deux sous-ensembles disjoints en partant du principe qu'un terme peut être qualifié de pertinent si son degré de dépendance aux autres termes est supérieur à son degré d'indépendance. Autrement, il peut être qualifié de non pertinent ou non signifiant, comme un bruit. Les deux sous-ensembles de termes en résultant peuvent alors être formalisés comme suit :

- Termes pertinents
$$T_R = \{t \in T_L \mid Dep(t/T_L) > InDep(t/T_L)\}$$
- Terms non pertinents
$$T_N = \{t \in T_L \mid InDep(t/T_L) \geq Dep(t/T_L)\}$$

Puisque la présence des termes non pertinents est intrinsèquement liée à l'efficacité du processus de prétraitement des documents d'apprentissage et à celle du processus de sélection de termes, nous avons fait le choix d'ignorer ces termes lors du développement de notre méthode d'analyse du type de besoin de l'utilisateur. Notre but est ainsi d'analyser seulement les termes pertinents. En fait, les termes pertinents ne sont pas également informatifs vis-à-vis du besoin de l'utilisateur, ainsi, nous différencions deux types de termes pertinents :

- Termes fortement pertinents
$$T_{SR} = \{t \in T_R \mid H_t > Max(H_{t'}), t' \in T_N\}$$
- Termes faiblement pertinents
$$T_{WR} = \{t \in T_R \mid H_t \leq Max(H_{t'}), t' \in T_N\}$$

Une telle différentiation est basée cette fois sur la proportion d'habituation des termes pertinents et non pertinents, qui est très bénéfique pour l'estimation de l'importance des termes dans la représentation du besoin de l'utilisateur. Parmi les termes pertinents, ceux dont la proportion d'habituation est du même ordre que les termes non pertinents sont identifiés en tant que termes faiblement pertinents, et ceux qui restent sont identifiés en tant que termes fortement pertinents.

Pour faire suite à l'exemple de la section 5.2, il est possible de constater que t_1 et t_2 sont des termes non pertinents tandis que t_3 est un terme pertinent selon les définitions données ci-dessus. Les proportions d'habituation des termes t_1 et t_2 sont respectivement 0.24 et 0.27, alors que la proportion d'habituation du terme t_3 est 0.35 ; il peut donc être considéré comme fortement pertinent.

5.2.2 Spécification du type de besoin de l'utilisateur

Nous proposons ici trois différents critères pour l'évaluation de la nature du besoin de l'utilisateur. Les deux premiers critères peuvent être calculés à partir des exemples positifs uniquement, ou plus précisément, en analysant l'apprentissage du modèle ILoNDF réalisé à partir des exemples positifs du besoin de l'utilisateur. La troisième critère requiert à la fois des exemples positifs et négatifs et de ce fait l'apprentissage de deux modèle ILoNDF.

Exhaustivité

L'exhaustivité E est définie comme le rapport du nombre de termes pertinents appris par le modèle ILoNDF associé au traitement des exemples positifs sur le nombre de termes utilisés pour la représentation des documents d'apprentissage, soit :

$$E = \frac{\mid T_R^+ \mid}{\mid T \mid} \tag{5.2}$$

Ce critère est utile pour reflèter la diversité des sujets thématiques que recouvre le besoin de l'utilisateur. En fait, l'exhaustivité totale signifie que l'utilisateur est intéressé par tous les sujets disponibles et son besoin en informations peut être qualifié d'exploratoire ou imprécis.

Spécificité

La spécificité S est définie comme le rapport du nombre de termes fortement pertinents sur le nombre de tous les termes pertinents appris par le modèle associé au traitement des exemples positifs, soit :

$$S = \frac{\mid T_{SR}^{+} \mid}{\mid T_{R}^{+} \mid} \tag{5.3}$$

Ce critère donne un bon indicateur pour évaluer à quel point l'utilisateur a été précis dans la description de son besoin en informations. En effet, lorsque beaucoup de termes fortement pertinents sont identifiés à partir de la description du besoin de l'utilisateur contre très peu de termes faiblement pertinents, on peut conclure que le besoin de l'utilisateur est précis (ou précisément exprimé) et que l'utilisateur désire obtenir des informations qui s'adaptent parfaitement à son besoin, et vice versa.

À ce stade, il est important de noter que l'exhaustivité et la spécificité du besoin de l'utilisateur 1) sont indépendantes du nombre d'exemples que fournit l'utilisateur pour décrire son besoin en informations, et 2) ne sont pas directement liées l'une à l'autre : un niveau élevé de spécificité ne correspond pas forcément à un niveau élevé d'exhaustivité et l'opposé est aussi vrai.

Contradiction

Le besoin de l'utilisateur peut être qualifié comme contradictoire si beaucoup de termes pertinents ont été à la fois appris par les modèles associées aux exemples positifs et négatifs. Ce qui revient à dire qu'une forte corrélation existe entre les exemples positifs et négatifs que choisit l'utilisateur pour décrire son besoin en informations. Le taux de contradiction peut ainsi être donné par :

$$C = \frac{\mid T_{R}^{+} \cap T_{R}^{-} \mid}{\mid T_{R}^{+} \mid + \mid T_{R}^{-} \mid - \mid T_{R}^{+} \cap T_{R}^{-} \mid} \tag{5.4}$$

Évidemment, le filtrage des documents non pertinents sera d'autant plus difficile que le besoin de l'utilisateur sera contradictoire. Néanmoins, la contradiction dans le comportement de l'utilisateur signifie implicitement qu'il a été très sélectif dans la description de ce dont il a besoin et donc il est raisonnable de s'attendre à ce que le système soit également très sélectif dans sa réponse.

Précision attendue par l'utilisateur

Dans le but d'intégrer notre analyse du type de besoin de l'utilisateur dans le processus du filtrage d'informations, nous avons combiné les critères décrits ci-dessus en une mesure globale, appelée "*Précision attendue*" (ou "*Expected Precision*" (EP)), définissant l'importance que l'utilisateur souhaite accorder à la précision sur le rappel. Elle est donnée par :

$$EP = (1 - \rho)(\frac{(1 - E) + S}{2}) + \rho\, C \tag{5.5}$$

où le paramètre $\rho = \gamma/\alpha$ représente le taux relatif aux paramètres γ et α qui spécifient respectivement l'importance des exemples négatifs et positifs dans Eq. 5.1. Dans le cas où seuls des exemples positifs sont disponibles, il ressort que $\rho = 0$.

5.2.3 Adaptation de la fonction de décision

Le maintien d'un haut niveau à la fois de précision et de rappel est indispensable à la réussite de tout système de filtrage d'informations. Cependant, un tel équilibre entre précision et rappel n'est pas évident lorsque la description du besoin de l'utilisateur est très imprécise ou même très précise, indépendamment de l'efficacité du système lui-même. Parmi les critères d'évaluation de l'efficacité du filtrage (cf. Section 1.5.1), la mesure F_β est principalement conçue pour incorporer le fait que les utilisateurs peuvent avoir différentes préférences en matière de précision et de rappel. En d'autres termes, cette mesure permet de mesurer l'efficacité du filtrage par rapport à un utilisateur qui attache au rappel β fois plus d'importance qu'à la précision. La valeur de β peut être directement spécifiée par l'utilisateur ou automatiquement déterminée par le système.

À notre connaissance, toutes les études utilisant cette mesure supposent que la valeur du paramètre β est connue (fournie par l'utilisateur) ; elles utilisent ainsi la mesure F_β avec une valeur fixe de β. Les valeurs de β les plus couramment utilisées sont 1 (la précision et le rappel sont d'égale importance) et 0.5 (la précision est deux fois plus privilégiée par rapport au rappel) (Robertson et Soboroff, 2002). Néanmoins, dans la pratique, les utilisateurs peuvent ne pas savoir comment spécifier leur besoin en termes de précision et de rappel par une valeur numérique exacte. Par conséquent, il serait très intéressant si le système lui-même pouvait automatiquement ajuster la valeur de β au regard d'un besoin spécifique de l'utilisateur. Notre idée est donc d'intégrer une telle fonctionnalité dans le processus du filtrage d'informations.

Sur la base de notre analyse du besoin de l'utilisateur, la valeur de la précision attendue par l'utilisateur peut être interprétée en tant qu'étant équivalente au paramètre α dans Eq. 1.13, et ainsi :

$$\beta = (\frac{1}{EP} - 1)^{0.5} \tag{5.6}$$

Notre approche vise à déterminer un bon compromis entre la précision et le rappel en fonction du type de besoin de l'utilisateur : Si la précision attendue par l'utilisateur est forte, il est rationnel de supposer que l'utilisateur veut avoir des informations correspondant parfaitement à son besoin ; et les résultats du filtrage doivent être régulés en vue de maximiser la précision. Par contre, si la précision attendue par l'utilisateur est faible, il est rationnel de supposer que l'utilisateur peut être intéressé par toutes sortes d'informations correspondant plus ou moins à son besoin ; et donc les résultats du filtrage doivent être régulés en vue de maximiser le rappel. La régulation des résultats de filtrage est réalisée par l'optimisation de la fonction de seuillage au sens de la mesure F_β sur un ensemble de validation (cf. Section 3.3.3).

5.2.4 Évaluation

Pour clarifier l'utilité de notre analyse du besoin de l'utilisateur en général, et en particulier lors de l'ajustement du seuil de filtrage, de nombreuses expérimentations ont été menées sur deux sous-ensembles du corpus Reuters-21578 : *Reuters10* et *Reuters90*. L'analyse du besoin de l'utilisateur a été réalisée avec et sans l'utilisation des exemples négatifs et, pour des fins de comparaison, le seuil de filtrage a été en premier lieu déterminé en optimisant la mesure F_1 ($\beta = 1$) et ensuite, en optimisant la mesure F_β (β est décrit par l'équation 5.6).

TAB. 5.3 – Analyse des termes appris par le modèle ILoNDF et la valeur du paramètre β de la mesure F_β en utilisant des exemples positifs uniquement dans le cas du corpus *Reuters10*

Catégories	T_{SR}	T_{WR}	T_N	T_L	EP	β
acq	17	42	217	276	0.57	0.88
corn	4	36	191	231	0.50	1
crude	5	47	250	302	0.48	1.04
earn	1	51	209	261	0.44	1.13
grain	4	55	221	280	0.46	1.09
interest	5	58	131	194	0.46	1.09
money-fx	4	81	137	222	0.41	1.2
ship	3	21	237	261	0.53	0.94
trade	12	64	184	260	0.48	1.04
wheat	4	44	194	242	0.48	1.04

TAB. 5.4 – Analyse des termes appris par le modle ILoNDF et la valeur du paramètre β de la mesure F_β en utilisant des exemples positifs uniquement dans le cas des 10 catégories les plus fréquentes du corpus *Reuters90*

Catégories	T_{SR}	T_{WR}	T_N	T_L	EP	β
acq	13	23	1030	1066	0.68	0.69
corn	1	19	779	799	0.52	0.96
crude	3	38	931	972	0.53	0.94
earn	1	40	893	934	0.51	0.99
grain	4	29	1084	1117	0.56	0.89
interest	0	70	580	650	0.49	1.02
money-fx	3	69	847	919	0.51	0.98
ship	3	16	708	727	0.58	0.86
trade	4	58	873	935	0.52	0.95
wheat	2	28	805	835	0.53	0.94

Exploitation des exemples positifs uniquement

Les résultats de l'analyse des termes appris par le modèle ILoNDF sur les corpus *Reuters10* et *Reuters90*, sont respectivement résumés dans les tableaux 5.3 et 5.4. Dans ces tableaux sont également indiquées la précision attendue et la valeur du paramètre β calculées pour chacune des 10 catégories plus populaires des corpus *Reuters10* et *Reuters90*. Les tableaux 5.5 et 5.6 présentent respectivement les résultats obtenus concernant la qualité du filtrage lorsque la valeur du paramètre β est mise à 1 et lorsqu'elle est calculée sur la base de la précision attendue par l'utilisateur (Eq. 5.6).

Dans le cas du corpus *Reuters10*, l'analyse des résultats présentés dans le tableau 5.3 permet de constater que toutes les catégories du corpus sont assez précises. Les valeurs de la précision attendue associées aux catégories varient de 0.41 à 0.57 et les valeurs du paramètre β correspondantes sont respectivement 1.2 et 0.88. Ces valeurs sont très proches de la valeur usuelle du paramètre ($\beta = 1$). Cela est normalement prévu pour la plupart des catégories dans les corpus

TAB. 5.5 – Résultats obtenus sur le corpus *Reuters10* lors de l'optimisation du seuil de filtrage en fonction des mesures F_1 et F_β en exploitant des exemples positifs uniquement.

	Optimisation de F_1			Optimisation de F_β		
Catégories	Précision	Rappel	F_1	Précision	Rappel	F_β
acq	0.8708	0.9421	0.9050	0.9186	0.8927	0.9072
corn	0.7647	0.6940	0.7290	0.7647	0.6964	0.7288
crude	0.8859	0.6984	0.7811	0.8859	0.6984	0.7773
earn	0.9590	0.9055	0.9315	0.9453	0.9149	0.9281
grain	0.9063	0.9732	0.9385	0.9063	0.9732	0.9415
interest	0.7521	0.6718	0.7097	0.746	0.7176	0.7303
money-fx	0.7551	0.6201	0.6810	0.7152	0.6313	0.6633
ship	0.8061	0.8876	0.8449	0.8061	0.8876	0.8424
trade	0.8203	0.8974	0.8571	0.8203	0.8974	0.8588
wheat	0.8028	0.8028	0.8028	0.8028	0.8028	0.8028
Mean 10C	0.8323	0.8095	0.8181	0.8311	0.8112	0.8180

standard mais n'est pas certainement le cas dans un contexte réel d'interaction directe avec un utilisateur dont le besoin peut varier de très précis à très imprécis. Tout de même, la légère déviation de la valeur usuelle de β apporte des perfectionnements apparents dans la précision des résultats du filtrage pour quelques unes des catégories du corpus. La catégorie la plus précise "acq" en est un exemple très clair, pour laquelle la différence entre précision et rappel lors de l'optimisation du seuil de filtrage en fonction de la mesure F_1 était à peu près de 5% en faveur du rappel, alors que la différence a tourné à l'avantage de la précision lors de l'optimisation du seuil de filtrage en fonction de la mesure F_β. En revanche, dans le cas des catégories les moins précises, telles "money-fx", "earn" et "interest", l'optimisation du seuil de filtrage en fonction de la mesure F_β tend à réguler les résultats en faveur du rappel. Cela n'implique pas forcément que le rappel soit plus élevé que la précision mais qu'un bon compromis soit trouvé en fonction de la valeur du paramètre β assignée à chaque catégorie représentative d'un besoin d'un utilisateur. En moyenne, la précision et le rappel sont très légèrement influencés par les différentes mesures d'optimisation.

Dans le cas du corpus *Reuters90*, la description des 10 catégories les plus fréquentes semble être plus précise que dans le cas du corpus *Reuters10*. Les valeurs de la précision attendue varient de 0.49 à 0.68. La raison en est simplement que l'exhaustivité de ces catégories est beaucoup plus faible dans le cas du corpus *Reuters90* au vu du nombre plus élevé de catégories, et par conséquent, du nombre plus élevé de termes choisis pour la représentation des documents. À cet égard, il est évident que ces 10 catégories auront tendance à être plus spécifiques lorsqu'elles coexistent avec les autres 80 catégories. Notre stratégie de seuillage semble également très intéressante sur le corpus *Reuters90*. Comme il ressort du tableau 5.6, un meilleur compromis entre précision et rappel a été atteint après l'optimisation du seuil en fonction de la mesure F_β, tel est notamment le cas des catégories "acq", "crude", "ship" et "trade". En moyenne, notre stratégie apporte, sur les 10 catégories les plus fréquentes, une amélioration de la précision d'environ 3% au détriment du

TAB. 5.6 – Résultats obtenus sur le corpus *Reuters90* lors de l'optimisation du seuil de filtrage en fonction des mesures F_1 et F_β en exploitant des exemples positifs uniquement.

	Optimisation de F_1			Optimisation de F_β		
Catégories	Précision	Rappel	F_1	Précision	Rappel	F_β
acq	0.8839	0.8691	0.8764	0.9153	0.7981	0.8736
corn	0.7593	0.7321	0.7455	0.7593	0.7321	0.7405
crude	0.7845	0.7513	0.7676	0.9016	0.5820	0.7168
earn	0.9543	0.8670	0.9086	0.9543	0.8670	0.9091
grain	0.8194	0.7919	0.8055	0.8194	0.7919	0.8070
interest	0.7545	0.6336	0.6888	0.7545	0.6336	0.6875
money-fx	0.6593	0.6704	0.6648	0.6593	0.6704	0.6647
ship	0.8105	0.8652	0.8370	0.8675	0.8090	0.8417
trade	0.7090	0.8120	0.7570	0.8173	0.7265	0.7713
wheat	0.9273	0.7183	0.8095	0.9273	0.7183	0.8155
Mean 10C	0.8062	0.7711	0.7861	0.8376	0.7329	0.7833
Mean 90C	0.6214	0.5659	0.5214	0.6299	0.5490	0.6178

rappel. Une amélioration moins significative peut être observée sur toutes les 90 catégories du corpus, néanmoins, la valeur de F_β (0.6178) est environ 10% plus élevée que celle de F_1 (0.5214), ceci prouve l'utilité de la stratégie aussi bien pour les petites catégories que les grandes du point de vue utilisateur.

Il est également intéressant de noter que le nombre de termes pertinents ($T_{SR} + T_{WR}$) dans chacun des deux corpus *Reuters10* et *Reuters90* est proche du nombre de termes choisi par la statistique de χ^2 pour chacune des catégories (50 termes). Ceci confirme encore davantage la fiabilité de la stratégie que nous avons adoptée pour analyser les termes appris par le modèle ILoNDF (cf. Section 5.2.1).

Exploitation des exemples positifs et négatifs

Dans toutes les expérimentations que nous effectuerons par la suite avec des exemples positifs et négatifs, nous avons conservé les valeurs des paramètres ajustées précédemment sur le corpus Reuters : $\eta = 0.5$ pour *Reuters10* et $\eta = 0.25$ pour *Reuters90*. Le tableau 5.7 montre les valeurs du paramètre β calculées en prenant en compte les taux de contradiction constatés sur les différentes catégories des corpus *Reuters10* et *Reuters90*. Il est possible de voir que les taux de contradiction reflètent directement le degré de proximité sémantique entre les différentes catégories. Par exemple, dans le cas du corpus *Reuters10*, le taux le plus élevé de contradiction est assigné à la catégorie "grain" qui est connue comme étant une super-catégorie de deux autres : "corn" et "wheat". D'autre part, les taux les plus faibles de contradiction sont assignés aux catégories "acq" et "earn" qui sont bien séparées des autres catégories. En conséquence, la précision attendue est élevée pour les catégories plutôt contradictoires, reflétant le fait que les utilisateurs — dont les besoins sont représentés par ces catégories — sont intéressés par des informations

TAB. 5.7 – Les valeurs du paramètre β calculées en prenant en compte les taux de contradiction lors du calcul de la précision attendue associée à chaque catégorie des corpus *Reuters10* et *Reuters90* — Cas d'exploitation d'exemples positifs et négatifs pour l'apprentissage.

	Reuters10			Reuters90		
Catégories	C	EP	β	C	EP	β
acq	0.38	0.47	1.06	0.29	0.58	0.85
corn	0.69	0.59	0.83	0.45	0.5	0.99
crude	0.49	0.49	1.03	0.54	0.53	0.93
earn	0.31	0.38	1.29	0.4	0.48	1.04
grain	0.88	0.67	0.71	0.73	0.60	0.82
interest	0.73	0.59	0.83	0.71	0.54	0.92
money-fx	0.69	0.55	0.90	0.71	0.56	0.89
ship	0.5	0.52	0.97	0.37	0.52	0.95
trade	0.66	0.57	0.87	0.56	0.53	0.94
wheat	0.69	0.58	0.85	0.5	0.52	0.96

répondant précisément à leurs besoins. L'optimisation des seuils en fonction de la mesure F_β aide le système à être plus précis et plus adapté aux différents types de besoins des utilisateurs. Cela dit, l'amélioration des résultats est moins remarquable qu'elle ne l'était dans le cas où l'analyse du besoin de l'utilisateur a été effectuée à partir d'exemples positifs uniquement (cf. Tableaux 5.5 et 5.8). C'est simplement parce que les résultats sont presque parfaits et un bon compromis a déjà été établi entre précision et rappel. Cependant, la catégorie "interest" représente exceptionnellement une nette amélioration de la précision qui a augmenté de 0.76 à 0.87, contre une légère diminution du rappel de 0.69 à 0.66.

La discussion précédente s'applique également sur les résultats obtenus sur le corpus *Reuters90*. Cependant, un point qui mérite d'être souligné concerne la diminution des taux de contradiction de certaines catégories du corpus *Reuters90* par rapport à ceux que nous avons observés sur le corpus *Reuters10*. L'explication pourrait être la suivante : La similarité sémantique entre les catégories est relativement moins forte dans le cas du corpus *Reuters10* que dans le cas du corpus *Reuters90*. Ainsi, on pourrait s'attendre, dans le cas du corpus *Reuters90*, à un taux plus élevé de contradiction et à une demande plus sélective d'informations de la part de l'utilisateur. Or, un tel raisonnement ne serait vrai que si tous les documents d'apprentissage associés aux autres catégories étaient utilisés en tant qu'exemples négatifs. En fait, dans la méthode de Query Zoning que nous utilisons, la sélection d'exemples négatifs se limite aux documents les plus proches des exemples positifs dont le nombre est égal au nombre d'exemples positifs. À cet égard, et en raison du nombre plus élevé de documents semblables dans le cas du corpus *Reuters90*, il est fortement probable que les exemples négatifs sélectionnés soient plus similaires l'un à l'autre qu'ils ne le sont dans le cas du corpus *Reuters10*, et ainsi, ils ne représentent qu'une partie des documents semblables aux exemples positifs. En conséquence, le nombre de termes en commun entre les exemples positifs et négatifs, c.-à-d. le taux de contradiction, est relativement plus faible dans le cas du corpus *Reuters90*.

TAB. 5.8 – Résultats obtenus sur le corpus *Reuters10* lors de l'optimisation du seuil de filtrage en fonction des mesures F_1 et F_β en exploitant des exemples positifs et négatifs

	Optimisation de F_1			Optimisation de F_β		
Catégories	Précision	Rappel	F_1	Précision	Rappel	F_β
acq	0.9374	0.9308	0.9341	0.935	0.935	0.935
corn	0.8814	0.9286	0.9043	0.8814	0.9286	0.9
crude	0.8785	0.8413	0.8595	0.8785	0.8413	0.859
earn	0.9696	0.9357	0.9524	0.9681	0.9461	0.9543
grain	0.9167	0.9597	0.9377	0.9338	0.9463	0.9379
interest	0.7583	0.6947	0.7251	0.87	0.6641	0.7727
money-fx	0.7949	0.8659	0.8289	0.7949	0.8659	0.8253
ship	0.8732	0.6966	0.7750	0.8732	0.6966	0.7777
trade	0.9352	0.8632	0.8978	0.9352	0.8632	0.9026
wheat	0.8219	0.8451	0.8333	0.8219	0.8451	0.8314
Mean	0.8767	0.8562	0.8648	0.8892	0.8532	0.8696

TAB. 5.9 – Résultats obtenus sur le corpus *Reuters90* lors de l'optimisation du seuil de filtrage en fonction des mesures F_1 et F_β en exploitant des exemples positifs et négatifs.

	Optimisation de F_1			Optimisation de F_β		
Catégories	Précision	Rappel	F_1	Précision	Rappel	F_β
acq	0.9241	0.8983	0.9110	0.9297	0.8844	0.9101
corn	0.9756	0.7143	0.8247	0.9756	0.7143	0.8256
crude	0.8402	0.7513	0.7933	0.8402	0.7513	0.7963
earn	0.96	0.8874	0.9223	0.9469	0.9215	0.9335
grain	0.8176	0.8121	0.8148	0.8276	0.8054	0.8185
interest	0.7961	0.626	0.7009	0.7961	0.626	0.7083
money-fx	0.6769	0.7374	0.7059	0.75	0.58	0.66
ship	0.8409	0.8315	0.8362	0.8409	0.8315	0.8364
trade	0.8396	0.7607	0.7982	0.8396	0.7607	0.8007
wheat	0.7917	0.8028	0.7972	0.7917	0.8028	0.797
Mean 10C	0.8463	0.7822	0.8104	0.8538	0.7678	0.8086
Mean 90C	0.5918	0.5739	0.511	0.6079	0.5676	0.5182

Pour conclure

Nous avons présenté une nouvelle manière de regarder le profil utilisateur en termes de critères de précision, d'exhaustivité et de contradiction. L'approche que nous avons proposée repose sur une analyse synthétique du contenu du modèle ILoNDF. Cette analyse se déroule naturellement de manière postérieure à l'apprentissage. Cela nous a permis de définir une méthode de seuillage basée sur la précision attendue qui dépend directement du comportement de l'utilisateur et des caractéristiques intrinsèques de son besoin d'informations. Les résultats des expérimentations que nous avons obtenus sont très encourageants et mettent en évidence le potentiel de l'intégration de l'utilisateur comme une composante très importante dans la stratégie d'évaluation d'un système de filtrage d'informations. Pourtant, il convient à ce propos de faire remarquer que ces résultats ne sont que des indications de l'intérêt de notre approche dans un objectif d'orientation vers la conception des systèmes de filtrage centrés utilisateur. En fait, il nous semble indéniable que le meilleur scénario d'évaluation d'un système de filtrage orienté utilisateur est de le faire tester par un grand panel d'utilisateurs Bêta-testeurs, mais la difficulté de réaliser un tel scénario nous a restreint à l'évaluation empirique des résultats.

5.3 Adaptation à la dérive du besoin de l'utilisateur

5.3.1 Motivation

Nous étudions ici le problème de la dérive de concept dans le cadre de la modélisation du besoin de l'utilisateur quand ce dernier peut changer avec le temps. Par exemple, un utilisateur peut être intéressé par un ensemble de sujets, C_1, pour une période de temps donnée, mais pourrait ne pas l'être pour une autre période de temps, pendant laquelle les centres d'intérêt de l'utilisateur se tournent vers d'autres sujets, C_2. Ainsi, les centres d'intérêt de l'utilisateur peuvent évoluer significativement tout au long de son interaction avec le système de filtrage. Au fil de l'évolution, des nouveaux sujets d'intérêt peuvent apparaître et d'autres disparaître. Vu ce contexte, deux points importants sont à mettre en avant en ce qui concerne la modélisation du profil utilisateur. D'une part, si les nouveaux sujets d'intérêt de l'utilisateur n'y sont pas considérés, l'utilisateur peut manquer des documents importants et pertinents à son besoin en informations car le système ne sera pas en mesure de les délivrer. D'autre part, si les sujets périmés demeurent représentés à travers le profil utilisateur, la représentation du profil sera de moins en moins précise et le système peut délivrer de plus en plus des informations non pertinentes à l'utilisateur. Donc, le système de filtrage devrait pouvoir modifier ses connaissances et s'adapter rapidement à l'évolution du besoin de l'utilisateur dans le temps.

Dans cette perspective, la modélisation du profil utilisateur a pour principal objectif de représenter les centres d'intérêt de l'utilisateur et de faire émerger leur variation au cours du temps. Le principe de fonctionnement est fondamentalement analogue à celui du filtrage adaptatif : les documents sont traités un par un et des jugements de pertinence binaires sont fournis par l'utilisateur pour tout document filtré[34]. Ainsi, les jugements de l'utilisateur peuvent être considérés comme des exemples relativement indépendants du besoin de l'utilisateur correspondant aux différents intervalles de temps.

[34] Il faut noter que, dans certains cas, le jugement de l'utilisateur peut ne porter que sur une partie des documents filtrés par le système, mais cela n'aura pour conséquence qu'un nombre plus restreint d'exemples d'apprentissage, et donc, une certaine perte de performance.

Peu de travaux se sont intéressés au problème de l'évolution du besoin de l'utilisateur sous l'angle de la dérive de concepts. Ces travaux abordent le problème en utilisant l'une des approches vues précédemment dans la section 2.3. L'approche la plus utilisée est de faire oublier les anciens documents jugés par l'utilisateur à l'aide de fenêtres temporelles, de taille fixe ou adaptative (Mitchell et al., 1994; Klinkenberg et Renz, 1998). Dans le cas d'une dérive du besoin de l'utilisateur, il est généralement supposé que les nouveaux documents sont plus représentatifs du besoin courant de l'utilisateur que les anciens. À part des difficultés liées à l'ajustement de la taille de la fenêtre (cf. Section 2.3.3), un tel type d'approche ne pourrait être retenue que pour l'adaptation du profil en cas de dérive progressive du besoin de l'utilisateur. En cas d'une dérive brusque, cette approche ne permet pas de répondre efficacement au besoin de l'utilisateur du fait que le profil doit être mis à zéro ce qui rend incontournable une chute importante de performance et un délai relativement plus long pour la récupération de la performance (Wang et al., 2003; Valizadegan et Tan, 2007).

5.3.2 Méthode

L'apprentissage en ligne du modèle ILoNDF appuyé par sa capacité d'oubli devrait lui permettre de répondre mieux à la fois aux dérives brusques et progressives du besoin de l'utilisateur. Or, l'analyse expérimentale du comportement du modèle ILoNDF nous a fait percevoir que le délai nécessaire à la récupération de la performance pourrait être très long, en particulier, dans le cas d'une dérive brusque du besoin de l'utilisateur. Cela nous a amené à penser qu'il est théoriquement envisageable d'adapter le mode d'apprentissage du modèle ILoNDF de sorte à accélérer l'adaptation du profil utilisateur aux nouveaux centres d'intérêt de l'utilisateur en cas de détection d'une dérive de ceux-ci. Grâce à la capacité d'oubli du modèle ILoNDF, il suffit de renforcer l'apprentissage de nouvelles données pour faire oublier les plus anciennes données qui deviennent périmées. Pour ce faire, nous devons introduire un facteur de biais δ dans la règle d'apprentissage du modèle ILoNDF (Eq. 4.16) permettant de retrouver l'équation suivante :

$$\Phi_k = I + \Phi_{k-1} - \delta \frac{\tilde{x}_k \tilde{x}_k^T}{\|\tilde{x}_k\|^2} \tag{5.7}$$

avec

$$\delta = \begin{cases} \lambda \, min(diag(\Phi_{k-1})) & \text{si une dérive est détectée au niveau de la donnée } x_k \\ 1 & \text{sinon} \end{cases}$$

La détection de la présence d'une dérive est donc indispensable à une adaptation rapide du profil au changement du besoin de l'utilisateur. Pour ce faire, nous employons une méthode statistique simple, analogue à celle utilisée dans beaucoup des travaux de la littérature comme (Klinkenberg et Renz, 1998) et (Fung et al., 2004) pour ajuster la taille d'une fenêtre glissante sur les derniers documents pris en compte par le modèle courant. L'idée consiste à présenter au système de filtrage les documents d'apprentissage en lots successifs, et à surveiller la variation de la valeur d'un indicateur de performance, tels que la précision et le rappel, dans le temps : À chaque pas de temps, la valeur moyenne de l'indicateur, $\overline{m}$, est calculée sur les m derniers lots. La valeur de l'indicateur calculée sur le lot courant, s_{l_c}, est comparée à un intervalle de confiance, qui correspond à la moyenne plus ou moins n fois l'écart-type évalué sur les m derniers lots. Selon le résultat, nous traitons deux cas :

- Si $s_{l_c} > \overline{m} + n \; \sigma_m$, aucune dérive n'est intervenue dans le besoin de l'utilisateur, et dans ce cas, la règle classique d'apprentissage du modèle ILoNDF est appliquée ($\delta = 1$) ;

- Si $s_{l_c} \leq \overline{m}+n\,\sigma_m$, une dérive du besoin de l'utilisateur est prévue, et la règle d'apprentissage biaisé du modèle ILoNDF doit être mise à profit pour accélérer le suivi de cette dérive.

Dans ce qui suit, nous évaluons d'abord le comportement du modèle ILoNDF en présence d'une dérive du besoin de l'utilisateur sans et avec le biais de l'apprentissage sur tous les documents correspondant aux nouveaux centres d'intérêt de l'utilisateur. Ensuite, nous évaluons l'intérêt de ne procéder à un biais d'apprentissage qu'en cas de détection de perte significative de performance.

5.3.3 Évaluation

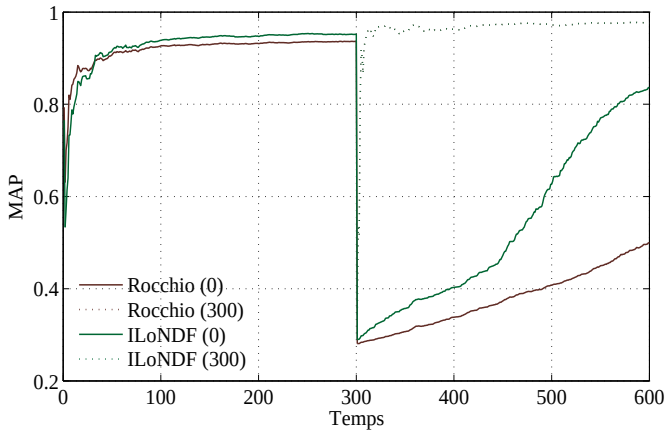
Analyse préliminaire de l'impact de la dérive sur la performance du modèle ILoNDF

Nous commençons par une analyse expérimentale du comportement du modèle ILoNDF pour évaluer, en premier lieu, à quel point la capacité d'oubli peut l'aider à s'adapter au changement des intérêts de l'utilisateur, et pour montrer, en second lieu, le potentiel de renforcer cette capacité en présence d'une dérive du besoin de l'utilisateur. Pour cela, nous nous appuyons sur les résultats des expérimentations préliminaires menées sur le sous-ensemble du corpus Reuters-21578, *Reuters10*. Les expérimentations ont été réalisées comme suit. La présence d'une dérive du besoin de l'utilisateur au cours du temps a été simulée en utilisant les 300 premiers documents d'apprentissage de deux catégories du corpus *Reuters10*. La dérive est donc provoquée par le passage d'une des catégories C_1 à l'autre C_2. Après l'apprentissage de chaque document par le modèle ILoNDF, la performance du modèle en termes de MAP est évaluée sur un ensemble indépendant de documents (les documents de test du corpus). Notons que nous n'utilisons ici que des exemples positifs et nous utilisons la métrique de MAP dans le but d'évaluer la performance indépendamment du choix des paramètres et du réglage du seuil de filtrage.

La figure 5.8 illustre le comportement du modèle ILoNDF, ainsi que de la méthode de Rocchio, en présence d'une dérive du besoin de l'utilisateur. Pour chaque méthode, le profil utilisateur est appris à partir de tous les documents d'apprentissage disponibles à un moment donné t (ILoNDF (0) et Rocchio (0)). Pour des fins de comparaison, nous avons aussi tracé sur la figure 5.8 la performance des méthodes si un nouveau modèle est appris à partir du moment où commence la dérive (ILoNDF (300) et Rocchio (300)). La figure 5.8 illustre deux cas possibles. Dans le premier cas, la dérive du besoin de l'utilisateur est simulée par deux catégories bien distinctes : "acq" et "earn". Dans le deuxième cas, les deux catégories sont relativement moins distinctes : "interest" et "money-fx". Quoi qu'il en soit, dans les deux cas, la perte de performance due à la dérive est énormément prononcée pour ILoNDF et Rocchio. Pourtant, ILoNDF semble meilleure que Rocchio dans le cas des catégories bien distinctes. ILoNDF parvient, dans ce dernier cas, à annihiler graduellement l'effet des documents précédemment appris (acq) mais il faut bien longtemps avant qu'il arrive à un niveau de performance équivalent à celui obtenu en cas d'absence de dérive (ou si un nouveau modèle est appris dès la présence de la dérive). Dans cet exemple précis des catégoires "acq" et "earn", nous avons trouvé que le modèle ILoNDF n'arrive au niveau idéal obtenu par un modèle ILoNDF appris à partir des 300 premiers documents de la catégorie "earn" qu'après l'apprentissage sur 2150 documents issus de cette catégorie ("earn"). Dans le cas des catégories "interest" et "money-fx", l'annulation de l'effet des documents de la catégorie "interest" est beaucoup moins évidente à gérer par le modèle ILoNDF à cause de la forte similarité entre catégories. Mais c'est aussi en raison de cette similarité que la performance obtenue par le modèle au début du passage entre les deux catégories est bien meilleure que celle obtenue par un nouveau modèle appris uniquement à partir des premiers documents de la catégorie "money-fx".

(1)

C_1= “acq” et C_2= “earn”



(2)

C_1= “interest” et C_2= “money-fx”

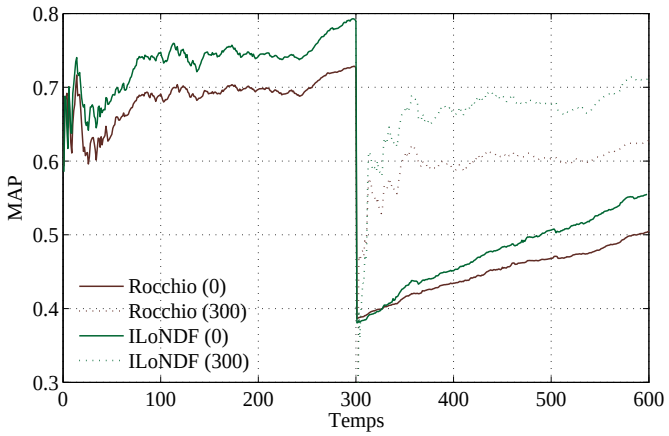
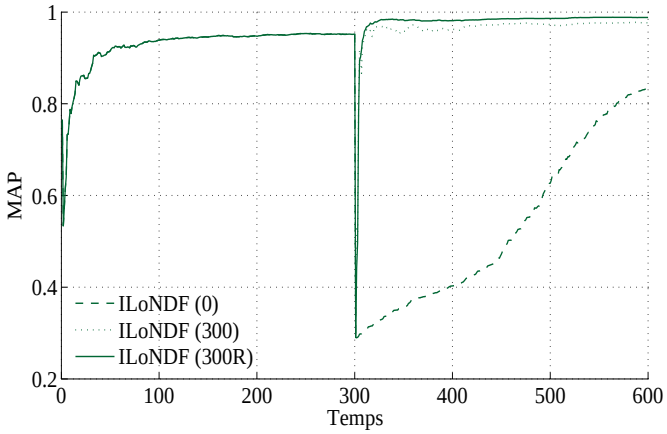


Fig. 5.8 – Comportement de Rocchio et ILoNDF en présence d’une dérive brusque du besoin de l’utilisateur. À partir du moment t=300 (après l’apprentissage sur 300 documents), le besoin de l’utilisateur change de C_1 à C_2. Le comportement “idéal” des méthodes si un nouveau modèle est appris à partir du moment où commence la dérive est illustré au moyen des lignes pointillées.

Maintenant, nous pouvons nous tourner vers l’évaluation du potentiel à renforcer la capacité d’oubli du modèle ILoNDF en présence d’une dérive du besoin de l’utilisateur. Pour ce faire, nous commençons la construction du profil utilisateur à partir des 300 premiers documents d’une catégorie C_1 en appliquant la règle classique d’apprentissage du modèle ILoNDF, puis nous poursuivons la construction du modèle à partir des 300 premiers documents d’une autre catégorie C_2 en appliquant la règle d’apprentissage biaisé du modèle ILoNDF. La figure 5.9 illustre les résultats pour les deux cas précédemment mentionnés. Dans le cas des catégories “acq” et “earn”, le potentiel est énorme. Après l’apprentissage à partir de très peu de documents de la catégorie “earn” (5 documents), la performance du modèle devient équivalente, voire supérieure, à celle obtenue par un nouveau modèle appris à partir des documents de la catégorie “earn” uniquement. Or, dans le cas des catégories “interest” et “money-fx”, le gain de performance est moins important. Au début de la dérive (de $t = 301$ à $t = 362$), la performance augmente dû au renforcement de l’apprentissage des variables caractéristiques de la catégorie “money-fx” au regard de celles caractéristiques de la catégorie “interest”, mais au bout d’un certain temps on observe une perturbation de la performance. Cette perturbation peut être le résultat du

(1)

C_1= "acq" et C_2= "earn"



(2)

C_1= "interest" et C_2= "money-fx"

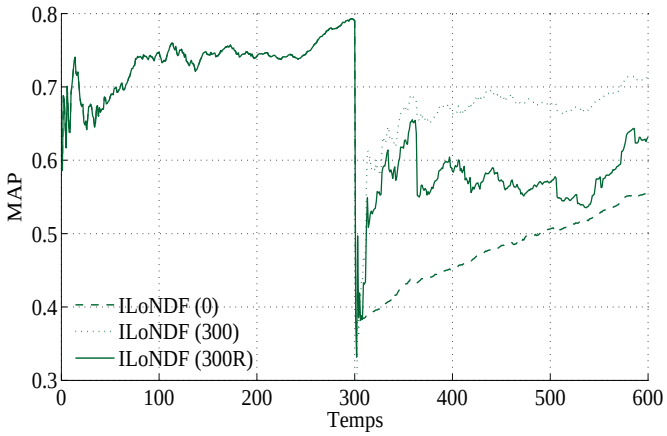


FIG. 5.9 – Comportement du modèle ILoNDF en présence d'une dérive brusque du besoin de l'utilisateur. À partir du moment $t = 300$, le mode d'apprentissage biaisé du modèle ILoNDF est appliqué : ILoNDF (300R).

renforcement de l'apprentissage des variables communes aux deux catégories "interest" et "money-fx". Cela propose que le biais de l'apprentissage ne doit être appliqué sur tous les documents de la catégories "money-fx" mais seulement lors de la détection d'une perte importante de performance.

Détection et suivi de la dérive du besoin de l'utilisateur par le modèle ILoNDF

Ici, nous traitons le problème de la dérive du besoin de l'utilisateur en deux démarches successives : d'abord, la détection d'une dérive du besoin de l'utilisateur ; et ensuite, l'adaptation du profil utilisateur construit à l'aide du modèle ILoNDF de manière à suivre l'évolution du besoin de l'utilisateur. Pour les expérimentations décrites dans cette section, les documents sont répartis en 60 lots de taille égale, contenant chacun 20 documents. Nous utilisons la mesure F_1 comme indicateur de performance pour la détection de dérive du besoin de l'utilisateur. Le nombre de lots pris en compte pour le calcul de la moyenne est mis à $m = 20$ et la constante n à 5. Comme dans la section précédente, nous présentons les résultats obtenus pour deux cas possibles :

1. Cas des catégories bien distinctes : C_1= "acq" et C_2= "earn".

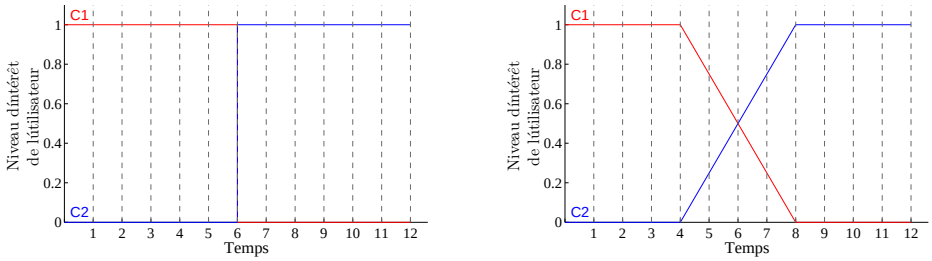


FIG. 5.10 – Deux types de dérive du besoin de l'utilisateur. (a) Dérive brusque : Le besoin de l'utilisateur change à partir du moment $t = 6$ de la catégorie C_1 à la catégorie C_2 ($\Delta t = 0$) (b) Dérive progressive : L'intérêt porté par l'utilisateur sur C_1 (C_2) diminue (augmente) progressivement à partir du moment $t = 4$ jusqu'à $t = 8$ ($\Delta t = 4$).

2. Cas des catégories relativement moins distinctes : C_1= "interest" et C_2= "money-fx".

Également, deux scénarios de dérive du besoin de l'utilisateur ont été simulés :

- Dérive brusque du besoin de l'utilisateur (Figure 5.10 (a)) : Les 30 premiers lots sont formés à partir de 300 documents issus d'une des catégories C_1 du corpus *Reuters10* ; ces documents sont considérés en tant qu'exemples positifs, et de 300 documents choisis aléatoirement des autres catégories en tant qu'exemples négatifs. Les 30 lots suivants sont formés à partir de 300 documents issus d'une autre catégorie C_2 du corpus *Reuters10* en tant qu'exemples positifs, et de 300 documents choisis aléatoirement des autres catégories en tant qu'exemples négatifs du besoin de l'utilisateur.
- Dérive progressive du besoin de l'utilisateur (Figure 5.10 (b)) : Les 20 premiers lots sont formés à partir des 200 premiers documents d'une des catégories C_1 en tant qu'exemples positifs, et de 200 documents choisis aléatoirement des autres catégories en tant qu'exemples négatifs. Les exemples positifs dans les 20 lots suivants sont simulés par 200 documents issus de deux catégories C_1 et C_2 en diminuant (augmentant) progressivement la proportion des documents issus de C_1 (C_2) dans les lots formés. Les exemples négatifs dans ces lots sont simulés par 200 documents choisis aléatoirement des autres catégories. Les 20 derniers lots sont formés à partir de 200 documents issus de la catégorie C_2 en tant qu'exemples positifs, et de 200 documents choisis aléatoirement dans les autres catégories en tant qu'exemples négatifs du besoin de l'utilisateur.

Résultats pour une dérive brusque du besoin de l'utilisateur

La figure 5.11 compare le comportement du modèle ILoNDF pour deux modes d'apprentissage : à gauche, nous montrons les résultats lors de l'application directe de la règle classique d'apprentissage du modèle ILoNDF (Eq. 4.16), à droite, nous vérifions la présence d'une dérive avant l'apprentissage du modèle sur le lot courant, et comme nous l'avons déjà précisé, la règle d'apprentissage biaisé du modèle ILoNDF (Eq. 5.7) est appliquée lors de la présence d'une dérive du besoin de l'utilisateur. La figure 5.11 montre la variation de performance en termes de MAP, précision, recall et F_1 (moyenne de quatre essais), mais la détection de la présence d'une dérive est basée uniquement sur la mesure F_1.

D'après les résultats obtenus, on peut voir qu'au moment du passage de la catégorie C_1 à

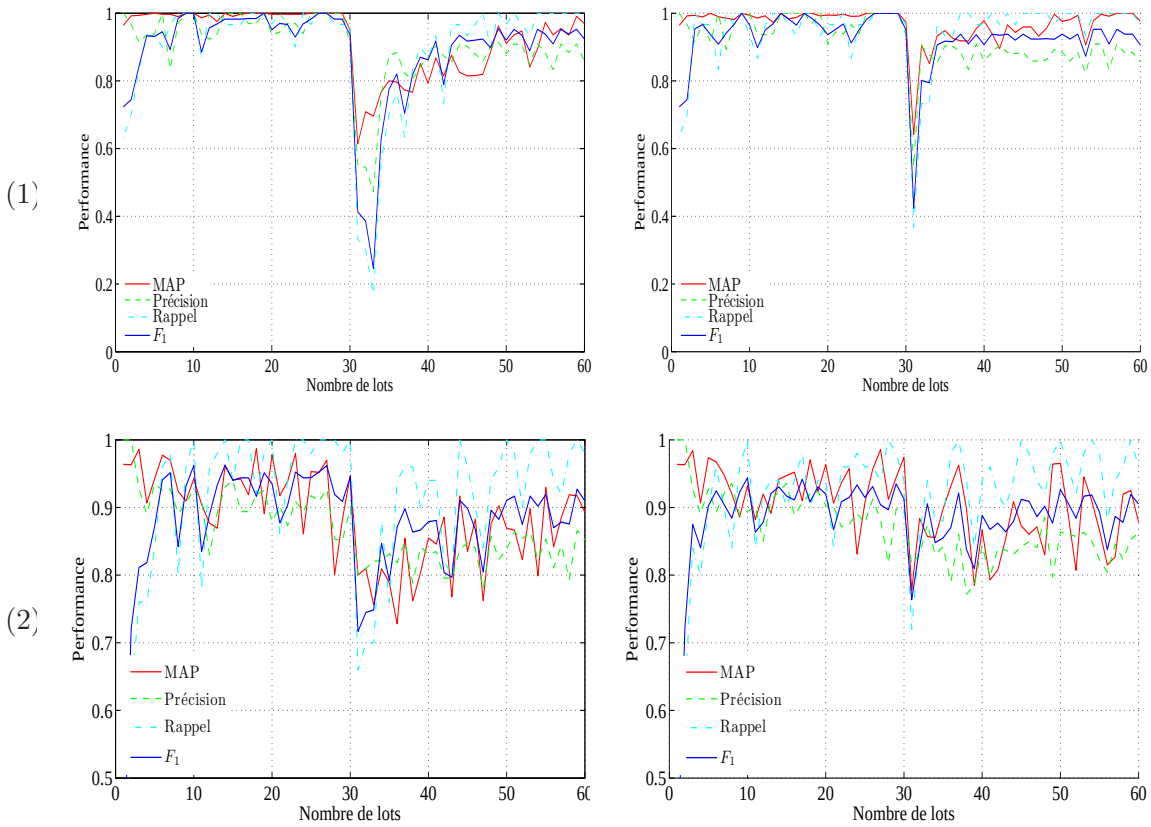


FIG. 5.11 – Comportement du modèle ILoNDF en présence d'une dérive brusque du besoin de l'utilisateur. (1) Cas des catégories C_1 = "acq" et C_2 = "earn" (2) Cas des catégories C_1 = "interest' et C_2 = "money-fx". À gauche, apprentissage classique du modèle ILoNDF. À droite, apprentissage biaisé lors de la détection d'une dérive du besoin de l'utilsateur.

la catégorie C_2 (à partir du lot #31), la performance diminue immédiatement et de façon spectaculaire. La différence entre les deux modes d'apprentissage du modèle ILoNDF réside dans la période de perte de performance et le délai nécessaire à la récupération de la performance. En fait, dans le cas de l'apprentissage classique du modèle ILoNDF, la période de perte de performance est relativement longue et la récupération de la performance est lente. Alors que la période de perte de performance est beaucoup moins longue et la vitesse de récupération de la performance est significativement plus rapide dans le cas de l'application de l'apprentissage biaisé du modèle ILoNDF chaque fois qu'une perte de performance se produit.

Sur la figure 5.11, on peut aussi voir que la perte de performance au moment où s'est produite la dérive brusque est plus forte dans le cas des catégories bien distinctes ("acq" et "earn") que dans le cas des catégories moins distinctes ("interest" et "money-fx"). C'est en raison de la similarité relativement plus forte entre les catégories "interest" et "money-fx" qui permet de tirer profit de l'apprentissage a priori de certaines des variables qui y sont communes.

Enfin, il convient d'insister sur le fait qu'une augmentation significative du taux de perte de performance n'est pas nécessairement associée à une dérive effective du besoin de l'utilisateur.

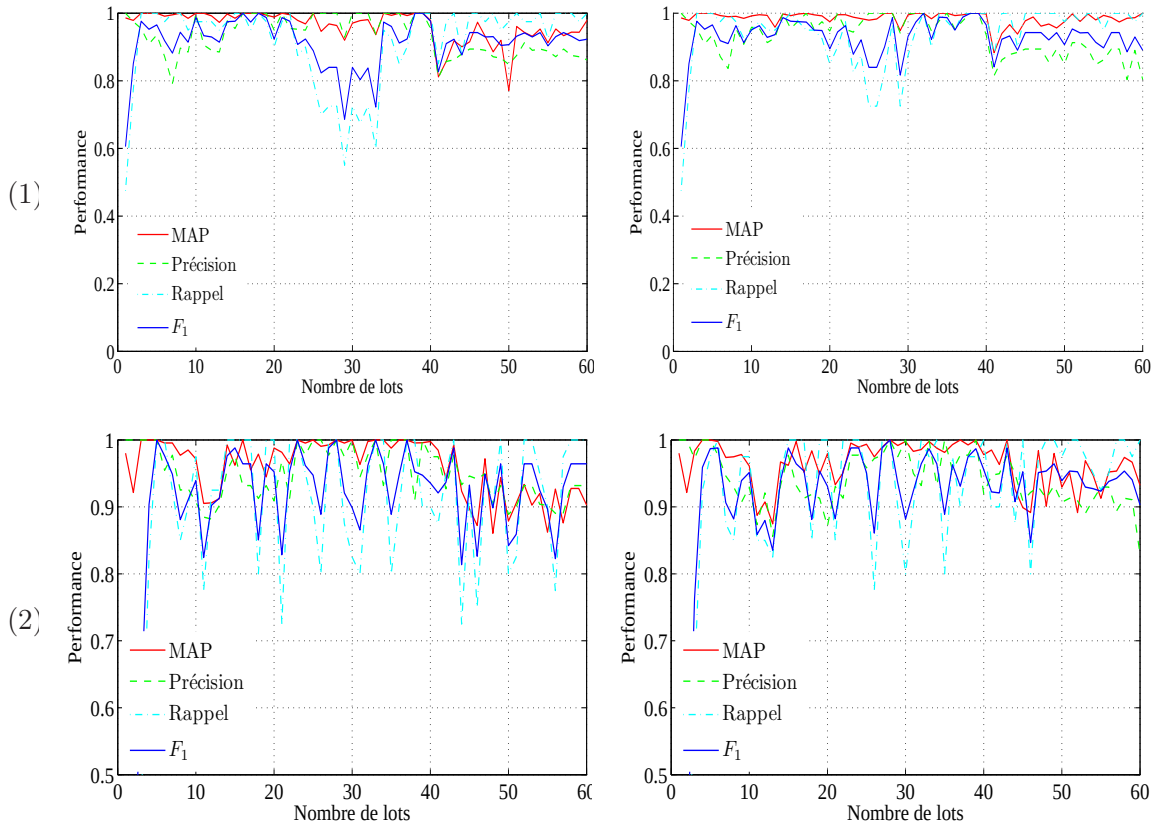


FIG. 5.12 – Comportement du modèle ILoNDF en présence d'une dérive progressive du besoin de l'utilisateur. (1) Cas des catégories C_1 = "acq" et C_2 = "earn" (2) Cas des catégories C_1 = "interest' et C_2 = "money-fx". À gauche, apprentissage classique du modèle ILoNDF. À droite, apprentissage biaisé lors de la détection d'une dérive du besoin de l'utilsateur.

Elle peut être causée par un léger changement des intérêts de l'utilisateur ou encore par des perturbations de performance lors du démarrage de l'apprentissage du profil utilisateur. Nous ne faisons, volontairement, aucune distinction entre la détection d'une dérive effective ou seulement potentielle. La raison tient en deux points : 1) Il ne suffit pas de renforcer l'apprentissage du modèle ILoNDF au moment où s'est produite une dérive mais aussi lors de toute perturbation de performance jusqu'à atteindre une certaine stabilisation. 2) Il semble utile de renforcer l'apprentissage du modèle pendant la phase de démarrage comme le montre la figure 5.11.

Résultats pour une dérive progressive du besoin de l'utilisateur

La figure 5.12 montre la variation de performance du modèle ILoNDF en présence d'une dérive progressive du besoin de l'utilisateur. Comme pour le cas de dérive brusque, nous présentons la moyenne des résultats de quatre essais pour les deux modes d'apprentissage du modèle ILoNDF. Les résultats révèlent que la différence en termes de performance entre ces modes d'apprentissage est moins visible dans le cas de dérive progressive que dans le cas de dérive brusque du besoin de l'utilisateur. C'est parce que la perte de performance est relativement moins importante pour le mode classique d'apprentissage du modèle ILoNDF dans le cas de dérive progressive du besoin

de l'utilisateur. Pendant que le besoin de l'utilisateur passe de C_1 à C_2, la perte de performance semble être maximale autour du point d'équilibre de dominance entre les deux catégories ($\alpha = 0.5$). De même, l'effet négatif de la dérive sur la performance du modèle est plus important dans le cas des catégories distinctes ("acq" et "earn") que dans le cas des catégories relativement moins distinctes ("interest" et "money-fx"). Dans les deux cas, l'application du mode d'apprentissage biaisé du modèle ILoNDF lors de la détection d'une perte significative de performance rend presque nul l'effet de la dérive sur la performance du modèle, et peut globalement accroître la performance avant et après le passage de la dérive du besoin de l'utilisateur.

5.4 Le système CASABLANCA : Nouvelles fonctionnalités

Un cadre principal d'application envisagé pour notre travail de thèse est celui de l'amélioration du système CASABLANCA initié par la société SES ASTRA[35], ceci dans le cadre du projet ESA SAT-N-SURF mené en partenariat avec le centre de recherche public Henri Tudor (CRP-HT)[36]. Le système CASABLANCA représente originellement un service de diffusion ciblée de sites web par satellite. Dans sa version de base, celui-ci ne comprend cependant aucune fonction de personnalisation élaborée et ne fait pas intervenir de traitement d'informations nouvelles en provenance du web, alors que ces deux points s'avèrent primordiaux à la fois pour la pertinence des sites web qu'il est susceptible de fournir aux utilisateurs, et pour son évolutivité. Dans le cadre du projet Sat-N-Surf, il a donc été choisi de développer un nouveau prototype en faisant intervenir les travaux de recherche issus de notre propre thèse et ceux d'une autre thèse menée en parallèle dans l'équipe MAIA au LORIA (Castagnos et Boyer, 2006). Ces travaux ont permis la mise en place de l'architecture définitive du système CASABLANCA permettant de combiner les avantages du filtrage dirigé par le contenu (CORTEX) avec ceux du filtrage collaboratif (MAIA).

Par la suite, nous donnons tout d'abord une description générale du système CASABLANCA et de l'architecture que nous avons proposé de mettre en place afin de lui intégrer des nouvelles fonctionnalités permettant d'effectuer un filtrage personnalisé d'informations. Ensuite, nous expliquons les différentes techniques utilisées lors de chacune des phases du processus de filtrage.

5.4.1 Contexte

La transmission satellite de CASABLANCA couvre actuellement la quasi-totalité de l'Europe ; elle est divisée en trois canaux (ou webcasters) correspondant aux trois zones géographiques : Europe, Allemagne et Italie. La transmission comporte des centaines des sites web, des forums de discussion (news groups) et des listes de diffusion (mailing-lists) sur l'Internet. Ce service est entièrement gratuit parce que financé par la publicité apparente sur les sites diffusés. En 2005, au moment du projet Sat-N-Surf, près de 150,000 personnes ont déjà téléchargé le logiciel client du système CASABLANCA dans toute l'Europe. La politique de diffusion est multilingue, c'est-à-dire les sites web peuvent être en plusieurs langues. Actuellement, le système supporte cinq langues différentes : anglais, allemand, italien, français et polonais, mais pourrait ultérieurement être étendu à d'autres langues. Le système CASABLANCA est structuré selon une architecture client-serveur. Du côté client, les sites web sont organisés dans diverses catégories thématiques et l'utilisateur peut voter pour les sites web qu'il souhaite recevoir en cochant des cases associées aux sites web et peut également suggérer l'inclusion de nouveaux sites web

[35] http ://www.ses-astra.com/landingPage/en/index.php
[36] http ://www.tudor.lu/

FIG. 5.13 – Interface de vote et de suggestion du système CASABLANCA - côté client.

dans le bouquet satellitaire (cf. Figure 5.13). Pour des raisons de marketing, le système n'offre pas à l'utilisateur la possibilité de spécifier explicitement les sites web qu'il ne veut pas, vu que ce type de vote peut entraîner des évaluations négatives des sites web dont l'inclusion est payée par des particuliers ou des professionnels. Les votes de l'utilisateur sont envoyés au serveur via une connexion Internet standard. Si l'utilisateur ne dispose pas d'une telle connexion, il ne peut pas voter et les sites web seront stockés aléatoirement dans la mémoire cache du terminal de l'utilisateur. Du côté du serveur, les votes provenant de l'ensemble de la communauté des utilisateurs sont agrégés et utilisés pour déterminer les sites web les plus populaires qui seront figurés dans la transmission.

L'apport de notre contribution dans le cadre du projet Sat-N-Surf tient essentiellement à l'intégration de nouvelles fonctionnalités permettant au système CASABLANCA de ne fournir que des informations conformes aux attentes des utilisateurs, tout en respectant les contraintes techniques inhérentes au système. Ces fonctionnalités incluent :

- La mise en place d'un processus d'indexation multilingue automatisé des sites web (côté serveur) ;
- L'implémentation du modèle ILoNDF comme modèle utilisateur (côté client) ;
- La gestion de la nouveauté dans la transmission (côté serveur) ;
- La mise en place d'une stratégie de combinaison mettant en jeu notre mode de filtrage et un filtrage collaboratif par l'intermédiaire du modèle ILoNDF (côté client).

Dans les sous-sections qui suivent, nous reprenons plus en détail chacune des fonctionnalités ci-dessus afin de mieux appréhender leur rôle et leur apport au système CASABLANCA.

5.4.2 Indexation conceptuelle indépendante de la langue des sites web

Travaux connexes

Depuis le début des années 70, l'aspect multilingue des ressources documentaires a été largement étudié, en particulier dans le domaine de la recherche d'informations (Salton, 1970, 1973; Sheridan et Ballerini, 1996; Carbonell et al., 1997). Principalement, il existe deux familles d'ap-

proches permettant l'indexation multilingue des documents :

- Les toutes premières approches sont basées sur l'utilisation de vocabulaires contrôlés (ou de thésaurus multilingues). Chaque concept de ces vocabulaires est décrit par un ou plusieurs termes synonymes avec des termes équivalents dans les différentes langues à gérer. Ces concepts sont alors employés pour l'indexation des documents indépendamment de leurs langues et les requêtes des utilisateurs sont exprimées en utilisant des termes issus du vocabulaire considéré pour l'indexation. Une telle approche s'est avérée particulièrement réussie lorsque les utilisateurs sont familiers avec le vocabulaire et tant que le nombre de concepts demeure raisonnable (Salton, 1970, 1973; Oard, 1997). Or, dans le cas de grandes collections de documents issus de sources hétérogènes, comme Internet, l'utilisation d'une telle approche ne serait plus avantageuse en raison 1) du coût élevé que nécessitait la préparation des vocabulaires contrôlés spécifiques aux collections et 2) de la difficulté que l'utilisateur peut rencontrer dans le choix des termes de sa requête à partir d'un vocabulaire de taille très importante (Oard et Marchionini, 1996). Ces restrictions ont motivé la recherche d'autres alternatives.
- Les approches d'indexation libre basées sur la connaissance. Selon le type de connaissance utilisée, ces approches peuvent être divisées en approches basées sur les dictionnaires et approches basées sur les corpus.
  - Les approches basées sur les dictionnaires se fondent sur l'utilisation de dictionnaires bilingues comme intermédiaires entre les documents et les requêtes de l'utilisateur, traduisant les requêtes dans les différentes langues dans lesquelles sont écrits les documents[37]. Un défaut majeur de cette approche tient à la possibilité de trouver plusieurs traductions possibles pour le même mot dans les dictionnaires. La présence d'une telle ambiguïté peut influencer très négativement la performance du système. Un autre défaut concerne le fait que les dictionnaires n'intègrent pas forcément tous les mots possibles pour assurer une traduction complète et correcte des requêtes de l'utilisateur.
  - Les approches basées sur l'analyse des corpus pour en extraire automatiquement les informations requises pour établir un modèle de traduction spécifique au domaine des documents. Certaines approches s'appuient sur des corpus parallèles, où chaque document existe en plusieurs exemplaires, l'original et ses traductions (Littman et al., 1997). D'autres approches plus répandues s'appuient sur des corpus comparables, où les documents traitent du même sujet sans pour autant avoir de correspondance un-à-un (Sheridan et Ballerini, 1996). Les approches basées sur les corpus sont plus performantes que celles basées sur les dictionnaires (Carbonell et al., 1997), mais il est souvent difficile de trouver des corpus parallèles ou comparables, adéquats pour les applications réalistes.

Pour notre part, nous avons choisi d'adopter une approche à vocabulaire contrôlé basée sur l'utilisation de l'annuaire ouvert DMOZ (Open Directory Project)[38] comme source des catégories multilingues. Cette approche a l'avantage par rapport aux autres approches d'être facilement extensible à toutes les langues. De plus, la taille restreinte du vocabulaire répond aux exigences du système CASABLANCA en temps de calcul, de dynamicité et de taille des vecteurs représentatifs des sites web. En fait, l'aspect multilingue n'est pas pris en considération dans la version antérieure du système CASABLANCA. Une approche d'indexation libre considérant tous les mots contenus dans les sites web, excepté les mots vides, a été utilisée quelle que soit la langue des

[37] Il est également possible de traduire tous les documents dans les langues potentielles des requêtes mais il serait plus réaliste de traduire seulement les requêtes du fait qu'elles sont sont souvent beaucoup plus courtes que les documents.

[38] www.dmoz.org

TAB. 5.10 – Statistiques sur le vacabulaire contrôlé construit automatiquement à partir des trois premiers niveaux de la hiérarchie de DMOZ.

Nombre de concepts	406
Nombre de termes en anglais associés aux concepts	6043
Nombre de termes en allemand associés aux concepts	2874
Nombre de termes en français associés aux concepts	1495
Nombre de termes en italien associés aux concepts	1550
Nombre de termes en polonais associés aux concepts	651

sites web à indexer. Le problème majeur de cette approche est que les vecteurs qui en résultent sont de dimension très importante.

Méthode

L'annuaire ouvert DMOZ fournit actuellement la plus grande hiérarchie de catégories sur le web. Il est développé et maintenu par une vaste communauté mondiale d'éditeurs bénévoles. Il comporte 15 catégories principales et environ 600,000 sous-catégories dont il existe deux types : Les sous-catégories directes qui sont effectivement classées sous la catégorie courante et celles indirectes qui sont aussi associées à la catégorie courante mais classées sous une autre catégorie de la hiérarchie (leur nom est suivi par @). Les catégories sont accessibles non seulement en anglais, mais aussi dans d'autres langues (environ 75 langues). À partir des trois premiers niveaux de cette hiérarchie, un vocabulaire contrôlé est organisé sous forme d'une liste de concepts. Chaque concept regroupe une des catégories du 2ème niveau — cette catégorie peut être utilisée pour représenter le concept —, et toutes ses sous-catégories directes du 3ème niveau, avec leurs catégories équivalentes dans les quatre langues supportées par le système, si disponibles. Nous ne considérons que les 11 catégories gérées par le système CASABLANCA. Le tableau 5.10 fournit des statistiques sur le vocabulaire en résultant. Pour des fins de pondération et de classification, nous associons également à chaque concept le nom de la super-catégorie du 1er niveau de la hiérarchie. La figure 5.14 représente un exemple de concept dans le vocabulaire.

Les concepts du vocabulaire contrôlé ainsi construits peuvent avoir beaucoup de termes en commun, ce qui aura pour effet de réduire le pouvoir de discrimination des concepts. C'est la raison pour laquelle, il est indispensable de procéder à une pondération des termes associés à chacun des concepts de sorte à diminuer l'importance des termes communs avec d'autres concepts. Pour ce faire, nous avons envisagé deux types de pondération. La première est faite au niveau des concepts, alors que l'autre est faite au niveau des super-catégories du premier niveau de la hiérarchie DMOZ. Ces deux pondérations peuvent être formulées comme suit. Soit $C = \{c_1, \ldots, c_n\}$ l'ensemble des concepts du vocabulaire, et soit $S = \{s_1, \ldots, s_m\}$ l'ensemble des super-catégories du premier niveau de la hiérarchie DMOZ.

- La pondération au niveau des concepts est donnée par :

$$W_{t_i}^{c_j} = \frac{1}{|\{t_i | t_i \in c_l, c_l \neq c_j\}|} \tag{5.8}$$

  Ainsi, le poids d'un terme t_i associé à un concept c_j est inversement proportionnel à sa fréquence d'apparition dans les autres concepts.

```
<INDEX_TERM>
  <SUPER_ENGLISH_CATEGORY> Computers </SUPER_ENGLISH_CATEGORY>
  <ENGLISH> artificial intelligence </ENGLISH>
  <ENGLISH> vision </ENGLISH>
  <ENGLISH> machine learning </ENGLISH>
  <ENGLISH> qualitative physics </ENGLISH>
  <ENGLISH> genetic programming </ENGLISH>
  <ENGLISH> agents </ENGLISH>
  <ENGLISH> belief networks </ENGLISH>
  <ENGLISH> programming languages </ENGLISH>
  . .
  <ENGLISH> neural networks </ENGLISH>
  <ENGLISH> games </ENGLISH>
  <ENGLISH> support vector machines </ENGLISH>
  <SUPER_FRENCH_CATEGORY> Informatique </SUPER_FRENCH_CATEGORY>
  <FRENCH> intelligence artificielle </FRENCH>
  <FRENCH> agents </FRENCH>
  <FRENCH> connexionisme </FRENCH>
  <FRENCH> vie artificielle </FRENCH>
  . .
  <SUPER_GERMAN_CATEGORY> Wissenschaft </SUPER_GERMAN_CATEGORY>
  <GERMAN> künstliche intelligenz </GERMAN>
  <GERMAN> neuronale netze </GERMAN>
  <GERMAN> zeitschriften </GERMAN>
  . .
  <SUPER_ITALIAN_CATEGORY> Scienza </SUPER_ITALIAN_CATEGORY>
  <ITALIAN> intelligenza artificiale </ITALIAN>
</INDEX_TERM>
```

FIG. 5.14 – Un exemple de concept dans le vocabulaire contrôlé construit à partir de la hiérarchie de DMOZ.

- La pondération au niveau des super-catégories est donnée par :

$$W_{t_i}^{c_j}|_{c_j \sqsubseteq s_k} = \frac{|\{t_i | t_i \in c_l, c_l \sqsubseteq s_k\}|}{|\{t_i | t_i \in c_l, c_l \sqsubseteq s_k\}| + |\{t_i | t_i \in c_l, \neg(c_l \sqsubseteq s_k)\}|} \tag{5.9}$$

  Notons qu'ici seuls les termes communs entre concepts relatifs aux différentes catégories du premier niveau de la hiérarchie DMOZ sont considérés comme problématiques.

Peu importe le type de pondération, le vecteur représentatif d'un site web est constitué en comptant les occurrences des termes associés aux concepts du vocabulaire qui apparaissent dans le site web. La dimension du vecteur correspond donc au nombre de concepts du vocabulaire et le poids accordé à chaque composante du vecteur désigne l'importance d'un concept correspondant dans le site web. La formule de calcul de ce poids est la suivante :

$$\mathrm{CF}_{c_j} = \sum_{i=1}^{r} TF(t_i) * W_{t_i}^{c_j} \tag{5.10}$$

où $TF(t_i)$ représente la fréquence d'occurrence du terme t_i associé au concept c_j dans le site web ; et $W_{t_i}^{c_j}$ représente le poids associé au terme t_i selon l'une des méthodes de pondération précitées. Les poids finaux des concepts sont calculés selon le schéma TF-IDF avec la normalisation cosinus. La formule exacte est :

$$V_{c_j} = \frac{\mathrm{CF}_{c_j} . \log \frac{N}{n_t}}{\sqrt{\sum_{sitesweb} (\mathrm{CF}_{c_j} . \log \frac{N}{n_i})^2}} \tag{5.11}$$

où N représente le nombre de sites web contenus dans le bouquet de transmission ; et n_{c_j} représente le nombre de sites web contenant au moins un des termes associés au concept c_j.

Évaluation

Nous présentons ici une série d'expérimentations dont le but est de répondre aux questions suivantes : 1) Dans quelle mesure notre approche d'indexation est-elle efficace par rapport à l'approche d'indexation libre utilisée précédemment par le système CASABLANCA ? 2) Dans quelle mesure la performance est-elle affectée par le multilinguisme ? 3) Dans quelle mesure le nombre moins important de termes en d'autres langues que l'anglais dans le vocabulaire contrôlé affecte-il la qualité de l'indexation ? Les expérimentations ont été menées sur un corpus de sites web collecté à partir de l'annuaire de Google[39]. Ce corpus comporte 11 catégories, et 1004 pages web au total dans les différentes langues supportées par le système CASABLANCA (cf. Annexe A).

En premier lieu, nous nous concentrons sur la performance des approches d'indexation dans le cas monolingue. Nous avons donc mené des expérimentations sur la partie de sites web du corpus en anglais qui serviront de base de comparaison avec le cas multilingue. Le tableau 5.11 présente les résultats obtenus selon les deux approches d'indexation en vocabulaire contrôlé et d'indexation libre en vocabulaire non contrôlé. Pour l'approche d'indexation contrôlée, les résultats montrent que la méthode de pondération au niveau des concepts n'est pas judicieuse et donne une qualité bien inférieure à celle obtenue sans pondération. La raison tient simplement au fait que la diminution de l'importance des termes qui apparaissent dans d'autres concepts relatifs à des sujets plus ou moins similaires peut sous-pondérer les termes propres à ces sujets, ce qui rend la qualité de leur représentation très mauvaise. La pondération au niveau des super-catégories semble une bonne solution au problème lié à la première approche de pondération et donne une qualité d'indexation très proche de celle obtenue par l'approche d'indexation libre, alors que la dimension de l'espace de représentation des pages web (406) est 27 fois moins important que celle correspondant à l'approche d'indexation libre (11137). Une tentative que nous avons aussi testée est de diviser les termes composés dans les concepts construits à partir des catégories de DMOZ en termes simples et de les ajouter à la description des concepts correspondants. Les résultats montrent qu'une telle démarche peut nuire à la qualité d'indexation.

Maintenant, pour évaluer l'aspect multilingue du vocabulaire contrôlé construit à partir de la hiérarchie DMOZ, nous utilisons le corpus dans sa totalité et nous comparons les résultats à ceux obtenus en considérant uniquement les pages web écrites en anglais (cf. Tableau 5.12). D'après les résultats, l'approche d'indexation contrôlée a réalisé une efficacité équivalente à 74% de celle obtenue dans le cas monolingue. Ces résultats sont très encourageants et il est certain qu'ils auraient été encore meilleurs si la hiérarchie de DMOZ était aussi développée pour les autres langues que pour l'anglais.

5.4.3 Modélisation du profil utilisateur

Comme précédemment mentionné, l'utilisateur peut exprimer son besoin en information en cochant des cases associées à des sites web via le portail du système (cf. Figure 5.13). Ce type de vote fournit des exemples positifs du besoin de l'utilisateur[40]. Ces exemples peuvent être utilisés pour alimenter un modèle ILoNDF et classer les sites web diffusés par satellite en fonction de leur pertinence vis-à-vis du besoin de l'utilisateur. L'évaluation de cette composante du système

[39] http ://www.google.fr/dirhp ?hl=fr

[40] Pour s'assurer du vote régulier des utilisateurs, la politique adoptée dans le système CASABLANCA consiste à ne considérer que les votes soumis durant les 15 derniers jours

TAB. 5.11 – Comparaison des méthodes d'indexation libre et d'indexation contrôlée dans le cas monolingue (anglais).

Méthode d'indexation	Nombre de termes	MAP
Indexation libre	11137	**0.9275**
Indexation contrôlée sans pondération	406	0.5489
Indexation contrôlée avec pondération au niveau des concepts	406	0.4356
Indexation contrôlée avec pondération au niveau des super-catégories	406	**0.8567**
Indexation contrôlée avec pondération au niveau des super-catégories et les termes composés decomposés en termes simples	406	0.8047

TAB. 5.12 – Comparison monolingue vs. multilingue dans le cas de l'approche d'indexation contrôlée basée sur la hiérarchie de DMOZ.

Monolingue (anglais)	0.8567
Multilingue (anglais, français, allemand, italien, polonais)	0.6422

a été largement étudiée dans les sections précédentes (à voir, particulièrement, Section 5.2.4).

5.4.4 Gestion de l'intégration de nouveautés

Dans la version originale du système CASABLANCA, le contenu du bouquet change toutes les semaines. Les sites web inclus dans le bouquet de transmission satellite sont déterminés manuellement sur la base du nombre de votes que les utilisateurs donnent pour exprimer leurs besoins spécifiques. Notre apport à ce sujet concerne l'automatisation de ce processus de mise à jour du bouquet de transmission et l'intégration de nouveauté dans le résultat du filtrage côté client en fonction du type de besoin de l'utilisateur[41].

Du côté serveur, la mise à jour du bouquet satellitaire est effectuée en fonction de préférences de l'ensemble des utilisateurs. Ce processus fait intervenir deux étapes clés :

- Profilage de communautés : Un algorithme de clustering est développé par Castagnos et Boyer (2006) pour la constitution de communautés d'utilisateurs d'intérêt commun sur la base de la proximité des votes des utilisateurs. Chaque communauté est donc liée à un ensemble d'utilisateurs et à un ensemble de sites web qui correspondent à leurs besoins communs. Ces sites web sont utilisés comme données d'apprentissage pour construire une représentation thématique basée sur le contenu pour chaque communauté d'utilisateurs (*profil communauté*) par l'intermédiaire du modèle ILoNDF.

[41] Notons que la gestion de nouveauté ne peut pas être menée par une approche de type filtrage collaboratif du fait qu'aucun vote n'est disponible pour les nouveaux documents.

- Mise en correspondance et sélection de sites web : D'une part, une mise en correspondance est établie entre le modèle ILoNDF associé à chaque communauté et les vecteurs des sites web inclus dans le bouquet de transmission courant (en utilisant une des méthodes DPM, V-PM et CS, cf. Section 4.4). D'autre part, une mise en correspondance est établie entre le modèle ILoNDF associé à chaque communauté et les vecteurs des sites web suggérés par les utilisateurs. Une moyenne des scores obtenus pour chacun des sites web, pondérée par le nombre d'utilisateurs associés à chaque communauté, est calculée et les sites web sont classés par ordre décroissant de leur score moyen. Les t premiers sites web sont inclus dans le bouquet de transmission suivant, t étant la taille maximale du bouquet.

Un flag de nouveauté est apporté à chaque nouveau site web dans le bouquet et y demeure pendant un mois afin d'attirer l'attention des utilisateurs sur la nouveauté. La gestion du cache du terminal de l'utilisateur est définie comme suit : Les sites web pour lesquels a explicitement voté l'utilisateur sont mis en priorité dans le cache de l'utilisateur. Si le cache de l'utilisateur n'est pas plein après l'ajout des sites web votés, nous considérons deux listes de sites web triés par leur ordre de pertinence : la liste des sites web déjà existants E et la liste des nouveaux sites web N. Une stratégie de fusion de ces deux listes est mise en place de sorte à maintenir un certain ratio de nouveauté k. Ce ratio est choisi en fonction du type de besoin de l'utilisateur (cf. Section 5.2.2). Plus précisément, le ratio de nouveauté est fixé à un niveau équivalent à la précision attendue par l'utilisateur (cf. Eq. 5.5) :

$$k = \frac{(1 - E) + S}{2} \quad ; \quad (\rho = 0) \tag{5.12}$$

Donc, nous supposons que plus le besoin de l'utilisateur est large, plus il est possible qu'il s'intéresse à la nouveauté.

5.4.5 Combinaison de filtrage par contenu et de filtrage collaboratif

Lors du développement de notre stratégie de combinaison entre filtrage par contenu et filtrage collaboratif, nous nous sommes appuyés sur deux critères : le nombre de votes fournis par l'utilisateur et le type de besoin de l'utilisateur. En effet, d'une part, le filtrage par contenu produit des résultats significativement meilleurs lorsque peu de votes sont disponibles. Le filtrage collaboratif produit des résultats plus stables lorsque assez de votes sont fournis par les utilisateurs. Cela suggère de favoriser le filtrage par contenu dans la phase de démarrage du système et de réguler l'équilibre entre les deux types de filtrage en fonction de l'augmentation du nombre de votes avec le temps. D'autre part, si le besoin de l'utilisateur est très précis, il est rationnel de supposer que l'utilisateur s'attend à une conformité aussi parfaite à son besoin spécifique et qu'il est donc moins intéressé par l'avis d'autres utilisateurs.

La stratégie de combinaison que nous avons proposée repose sur l'utilisation d'une moyenne pondérée des scores de pertinence des sites web obtenus par les deux types de filtrage. Le score final d'un site web est défini par :

$$RW(S) = \frac{\alpha\, RW_{CBF}(S) + \beta\, RW_{CF}(S)}{\alpha + \beta} \tag{5.13}$$

où α représente le degré d'importance attaché au type de filtrage par contenu ; $RW_{CBF}(S)$ est le score de pertinence du site web S obtenu selon le filtrage par contenu ; β représente le degré d'importance attaché au type de filtrage collaboratif ; $RW_{CF}(S)$ est le score de pertinence du

site web S obtenu selon le filtrage collaboratif[42].

Le degré d'importance attaché à chaque type de filtrage est déterminé par le modèle ILoNDF comme suit :

$$\alpha = (1 - EP) + (1 - \frac{v}{t}) \quad ; \quad \beta = 2 - \alpha \tag{5.14}$$

où v représente le nombre de votes de l'utilisateur et t le nombre total de sites web diffusés.

Avant de conclure cette section, il est important de préciser que l'évaluation de l'ensemble des techniques intégrées au système CASABLANCA a été conduite par le centre de recherche public Henri Tudor (CRP-HT) au Luxembourg. Pendant deux mois, 80 volontaires identifiés de toute l'Europe ont participé aux bêta-tests de la nouvelle version du système CASABLANCA. Tous les utilisateurs pilotes ont trouvé une nette amélioration du processus de filtrage, avec presque aucun impact sur la robustesse et la vitesse du système.

5.5 Conclusion

Dans ce chapitre, nous avons mis en pratique l'application du modèle ILoNDF au traitement du flux de données documentaires, et ceci dans le contexte du filtrage d'informations. Outre l'utilisation du modèle ILoNDF comme profil utilisateur, notre travail s'est orienté vers l'intégration de l'utilisateur comme une composante très importante dans la stratégie d'évaluation d'un système de filtrage. Le point de vue adopté est que le type de besoin de l'utilisateur peut aller d'un besoin très précis à un besoin très vague, voire contradictoire. À partir d'une analyse de l'apprentissage du modèle ILoNDF, nous avons développé trois critères purement objectifs pour caractériser le type de besoin de l'utilisateur en termes de spécificité, exhautivité et contradiction. Ces critères nous ont permis ensuite de définir une méthode de seuillage basée sur la précision attendue par l'utilisateur du système de filtrage et qui dépend directement du comportement de l'utilisateur et de son type de besoin en informations. Les résultats que nous avons obtenus sur le corpus Reuters ont montré l'intérêt de nos approches dans un objectif d'orientation vers la conception des systèmes de filtrage centrés utilisateur.

Ensuite, pour prendre en compte l'aspect évolutif du besoin de l'utilisateur, nous avons étendu le mode d'apprentissage du modèle ILoNDF pour renforcer sa capacité d'oubli. L'évaluation du modèle sous l'angle d'une dérive, progressive ou brusque, du besoin de l'utilisateur a mis en évidence les bonnes capacités d'apprentissage du modèle dans le cas des flux non stationnaires. Enfin, nous avons présenté notre contribution à la mise en place du système de distribution ciblée de sites web par satellite CASABLANCA.

[42]Dans le cadre du projet, l'approche utilisée pour le filtrage collaboratif ne produit comme recommandations qu'une liste non-ordonnée de sites web. C'est pour cette raison que nous affectons les scores obtenus par le filtrage basé sur le contenu aux sites web recommandés par le filtrage collaboratif et des scores de zéro aux sites web non recommandés par ce même type de filtrage. Ainsi, dans l'équation 5.13, $RW_{CF}(S)$=$RW_{CBF}(S)$ si S est recommandé par le filtrage collaboratif, sinon $RW_{CF}(S)$=0.

Chapitre 6

Méthodologie d'analyse temporelle de flux de données

La dynamique des flux de données constitue une source riche d'informations, rendant indispensable la mise en place de mécanismes de veille qui aident à en tirer parti et à déclencher des alertes automatiques pertinentes au sujet des changements qui pourraient intervenir dans les flux de données. Ce chapitre présente une méthodologie générale permettant l'analyse et le suivi de l'évolution temporelle d'un flux de données en termes d'émergence et de disparition des informations contenues dans le flux. Le principe de base de cette méthodologie consiste à découper le flux de données en fenêtres temporelles sur lesquelles on applique des techniques de clustering statique à des niveaux d'abstraction multiples (Section 6.2). Parmi les différentes méthodes de clustering existantes, nous retenons la famille des méthodes compétitives qui présentent la particularité d'entretenir des relations de voisinages prédéfinies ou adaptatives entre les clusters construits (cf. Sections 2.2.2 et 2.2.3). Le "Neural Gas" (NG) est adopté comme méthode de clustering pour l'ensemble des expérimentations que nous proposons dans ce chapitre. La mise en évidence des relations existant entre les clusters est donc réalisée à l'aide d'un apprentissage hebbien compétitif (CHL). Ensuite, un algorithme de regroupement des clusters travaillant à des niveaux d'abstraction multiples est mis en œuvre pour offrir plusieurs niveaux de granularité dans le processus d'analyse des clusters et leur relations. Fondamentalement, deux niveaux d'abstraction — micro-abstraction et macro-abstraction — sont constitués et puis organisés hiérarchiquement avec le niveau de base formé par le réseau de neurones (NG), cf. Figure 6.1. Le modèle en résultant permet, d'une part, de mettre en évidence les différents types de relations entre clusters dans une fenêtre temporelle et, d'autre part, d'évaluer l'importance de changements potentiels entre deux fenêtres temporelles d'un flux de données. En effet, ce modèle, qui est statique par construction, permet de prendre en compte l'aspect temporel en comparant les clusters construits à partir de deux fenêtres temporelles tout en considérant leur appartenance aux différents niveaux d'abstraction (Section 6.4). Le schéma de la figure 6.1 résume les grandes étapes relatives à l'élaboration de notre méthodologie d'analyse et de suivi de l'évolution d'un flux de données.

Mais avant tout, il nous a fallu résoudre de nombreux problèmes liés au dimensionnement du modèle de clustering et à l'évaluation de la validité des clusters construits. En effet, la détermination du nombre de clusters est un aspect crucial dont l'impact est très fort sur la qualité de clustering. Dans la littérature, la démarche générale suivie pour la détermination du nombre de clusters consiste à répéter l'algorithme de clustering en faisant varier le nombre de clusters dans

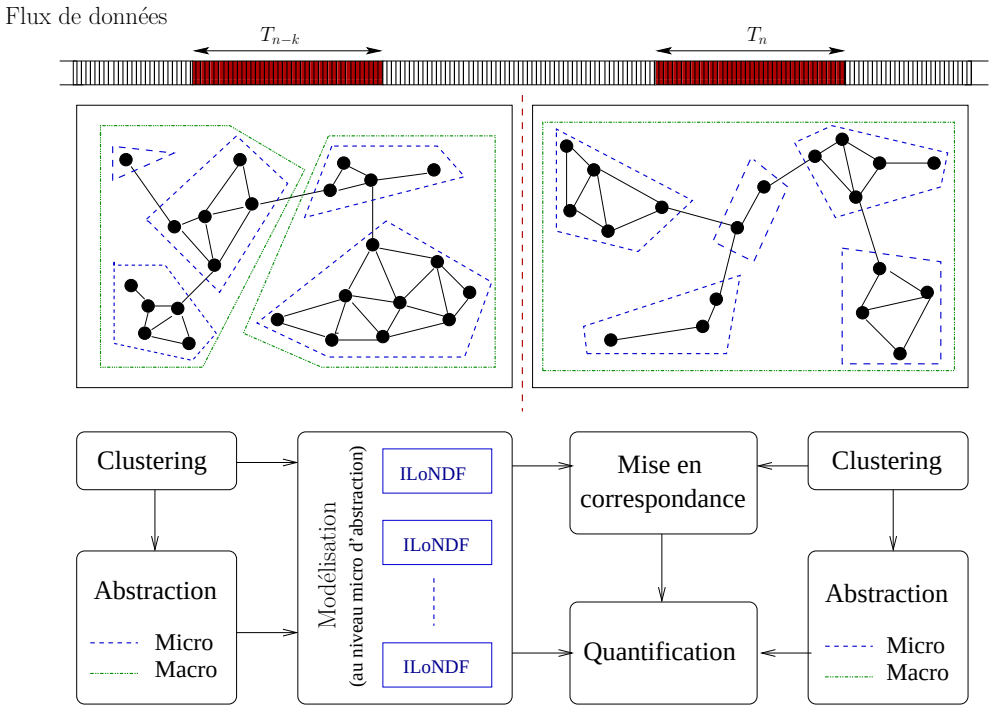


FIG. 6.1 – Les étapes clés pour la mise en œuvre d'une méthodologie d'analyse et de suivi de l'évolution des flux de données au fil du temps. Des modèles de clustering à des niveaux d'abstraction multiples (niveau de base + micro-abstraction + macro-abstraction) sont construits à partir des données issues de deux fenêtres temporelles T_{n-k} et T_n. Une mise en correspondance entre clusters issus de ces fenêtres est établie pour ensuite rendre possible la quantification des changements potentiels survenant dans le temps.

une certaine gamme de valeurs, et en cherchant à optimiser un critère de qualité de partition, tel que les critères de validité internes décrits dans la section 1.5.2. Comme nous allons le montrer plus loin, ces critères qui sont fondamentalement basés sur les mesures d'inerties intra-classe et inter-classes, évoluent de façon monotone avec l'augmentation du nombre de clusters dans le cas des données fortement multidimensionnelles. Pour cette raison, nous avons mis en place de nouveaux indices de validité des méthodes de clustering. Contrairement aux critères basés sur la distance, ces indices qui s'appuient sur la corrélation des variables inhérentes aux données associées à chacun des clusters, présentent l'avantage de bien se comporter dans le contexte des données fortement multidimensionnelles et éparses. De plus, ils permettent de contrôler la granularité du résultat de clustering, d'interpréter adéquatement le contenu sémantique des clusters et de détecter des clusters potentiellement non valides. Les indices de validité et leur évaluation seront détaillés dans la section suivante.

6.1 Nouveaux indices de validité de méthodes de clustering

L'ensemble des mesures que nous proposons ici se fonde tout particulièrement sur l'idée que la plupart des données fortement multidimensionnelles, quoiqu'elles soient très dispersées, présentent généralement un certain degré de corrélation entre leurs variables dans un espace de moindre dimension. Ces corrélations peuvent être exploitées pour générer des clusters interpré-

TAB. 6.1 – Notations et définitions de base

Notation	Définition
$\overline{C}$	L'ensemble des clusters issus de l'application d'une méthode de clustering à un ensemble de données ;
C	L'ensemble des clusters non vides découverts par la méthode de clustering ($C \subseteq \overline{C}$) ;
$\widetilde{C}$	L'ensemble des clusters non valides (définis par Eq. 6.6) découverts par la méthode de clustering ($\widetilde{C} \subseteq \overline{C}$) ;
$F(c)$	L'ensemble des variables présentes dans la représentation du contenu des données membres du cluster c ;
$CF(c)$	L'ensemble des variables noyaux (définies par Eq. 6.2) identifiées pour un cluster c ;
$C(f, f')\|d$	La valeur de corrélation entre deux variables f, f' intervenant dans la représentation d'une donnée d ;
w_f	Le poids d'importance attaché à une variable f dans la représentation d'une donnée d (calculé selon le schéma TF-IDF normalisé, cf. Section 3.1.2).

tables et significatifs du point de vue utilisateur.

Par la suite, nous verrons comment un ensemble de variables corrélées à la description de chacun des clusters découverts par une méthode de clustering peut être identifié et comment la similarité entre données/clusters peut être évaluée à l'aide des variables corrélées à leur description. Mais avant d'aborder ces points, quelques notations et définitions qui vont nous servir sont introduites dans le tableau 6.1.

6.1.1 Identification des variables de type noyau

La première étape de notre approche est basée sur l'identification automatique d'un ensemble de variables corrélées à la description de chacun des clusters issus de l'application d'une méthode de clustering à un ensemble de données. L'identification est réalisée sur la base de l'analyse des relations de co-occurrence entre les variables présentes dans la représentation des données membres d'un cluster c, ainsi que la fréquence d'occurrence de ces variables dans le cluster c. À cet effet, un score de corrélation CS est calculé pour chaque variable f dans chaque cluster selon la formule suivante :

$$CS(f) = \sum_{d \in c} \sum_{f', f \neq f'} C(f, f')|_d \tag{6.1}$$

TAB. 6.2 – Un jeu de six données représentées par six variables et classées en deux classes c_1 et c_2.

	c_1			c_2		
	d_1	d_2	d_3	d_4	d_5	d_6
f_1	1	1	1	0	0	0
f_2	0	0	0	1	1	1
f_3	1	0	1	0	1	1
f_4	0	1	0	1	0	0
f_5	0	1	1	0	1	0
f_6	1	0	0	0	0	0

avec

$$C(f,f')|_{d_n} = \begin{cases} w_f + w_{f'} & \text{if} \quad w_f \neq 0, w_{f'} \neq 0 \\ -2 \times w_f & \text{if} \quad w_f \neq 0, w_{f'} = 0, \sum_{d=d_1}^{d_{n-1}} C(f,f')|_d \neq 0 \\ -4 \times w_{f'} & \text{if} \quad w_f = 0, w_{f'} \neq 0, \sum_{d=d_1}^{d_{n-1}} C(f,f')|_d \neq 0 \\ 0 & \text{sinon} \end{cases}$$

De ce fait, si une variable f apparaît avec une autre variable f' dans une des données membres du cluster, le score de corrélation entre f et f' augmente proportionnellement à leurs poids dans la représentation de la donnée en question. Ensuite, si la variable f apparaît sans la variable f' dans une autre donnée membre du cluster, le score de corrélation diminue, et de même, si f' apparaît sans f. Dans ce dernier cas, une pénalité additionnelle est aussi appliquée au score de corrélation de la variable f pour refléter son absence de la représentation de certaines données membres du cluster c. Ainsi, il ressort comme résultat que les variables qui apparaissent ensemble dans les données membres du cluster et dont le poids est relativement fort auront un score de corrélation positif et suffisamment grand, tandis que les variables qui ont des relations de co-occurrence moins fréquentes avec les autres variables intervenant dans la représentation des données membres du cluster auront un score de corrélation négatif.

Sur la base de l'analyse ci-dessus, les variables qui sont corrélées à la description des données membre d'un cluster c, que nous appelons *"variables noyaux"*, sont définies par l'expression suivante :

$$CF(c) = \{f \mid f \in d,\ d \in c,\ CS(f) > 0\} \tag{6.2}$$

Exemple

Pour plus de clarté, considérons un exemple simple dans lequel nous avons six données $\{d_1, d_2, \ldots, d_6\}$ représentées par six variables $\{f_1, f_2, \ldots, f_6\}$, cf. Tableau 6.2.

Si l'on calcule les scores de corrélation des variables intervenant dans la représentation des données assignées au premier cluster, on obtient :

$$CS(f_1) = (2+2) + (-2+2+2-2) + (2-2+2) = 6$$

$$CS(f_3) = (2+2) + (-4) + (2+2-2) = 2$$

$$CS(f_4) = (2+2) + (-4-4) = -4$$

$$CS(f_5) = (2+2) + (2+2-2) = 6$$

$$CS(f_6) = (2+2) + (-4) + (-4-4) = -8$$

Donc, selon Eq. 6.2, les variables corrélées à la description du premier cluster c_1 sont :

$$CF(c_1) = \{f_1, f_3, f_5\}$$

Et, d'une manière analogue, l'ensemble des variables corrélées à la description du second cluster c_2 est :

$$CF(c_2) = \{f_2, f_3\}$$

Maintenant, en s'appuyant sur notre approche d'identification des variables corrélées aux clusters, nous définissons deux mesures de validité, que nous appelons le coefficient de corrélation intra-cluster et le coefficient d'isolation inter-clusters, qui seront utilisées ensuite pour l'évaluation des résultats globaux de clustering.

6.1.2 Coefficient de corrélation intra-cluster

Cette mesure est définie comme le rapport entre les poids des variables noyaux $CF(c)$ et les poids de la totalité de l'ensemble des variables intervenant dans la représentation des données membres du cluster $F(c)$. Donc, le coefficient de corrélation intra-cluster est calculé sur tous les clusters et est défini comme :

$$CC = \frac{1}{|C|} \sum_{c \in C} \frac{\sum_{d \in c, f \in CF(c)} w_f}{\sum_{d \in c, f' \in F(c)} w_{f'}} \tag{6.3}$$

Ce coefficient de corrélation permet de quantifier la qualité globale des clusters en termes d'homogénéité : Plus le nombre de variables noyaux communes aux données des clusters est important, plus les clusters sont homogènes. Les valeurs de CC sont comprises dans l'intervalle $[0, 1]$. D'une part, la limite inférieure de la valeur de CC (0) est atteinte lorsqu'aucune variable n'est pas corrélée à la description des clusters ; ce qui relève de l'incohérence totale du contenu des clusters. D'autre part, la limite supérieure de la valeur de CC (1) est atteinte lorsque toutes les données membres de chaque cluster partagent exactement les mêmes variables, mais également lorsque les clusters sont des singletons. Par consquent, un bon compromis entre ces deux limites devrait être vérifié selon le cas spécifique relatif à l'application du clustering.

6.1.3 Coefficient d'isolation inter-clusters

Nous définissons le taux de recouvrement d'un cluster c avec les autres clusters comme étant le rapport entre les poids des variables corrélées au cluster c et les poids de ces mêmes variables dans les autres clusters. De ce fait, le coefficient d'isolation inter-clusters IC est défini comme la moyenne des taux de recouvrement entre clusters, ce qui peut être exprimé par :

$$IC = \frac{1}{|C|} \sum_{c \in C} \frac{1}{|CF(c)|} \cdot \frac{\sum\limits_{d \in c, f \in CF(c)} w_f}{\sum\limits_{c' \in C} \sum\limits_{d \in c', f' \in CF(c) \cap CF(c')} w_{f'}} \tag{6.4}$$

À première vue, on pourrait croire que plus proche cette valeur est de 1 plus la solution de clustering peut être qualifiée d'optimale. Néanmoins, dans le cadre des données fortement multidimensionnelles et vu le fait que seules les variables noyaux des clusters sont considérées dans l'analyse du recouvrement entre les clusters, de très fortes valeurs de IC peuvent être atteintes, en particulier lorsqu'aucun ou peu de variables noyaux sont identifiées dans les clusters. En conséquence, pour que la solution de clustering puisse être optimale, une forte valeur en matière de IC doit être conciliée avec une forte valeur en matière de CC. Cela indique que les coefficients CC et IC devraient être considérés ensemble, et non de manière indépendante lors du choix de la solution optimale de clustering.

6.1.4 Coefficient global de validité (OqC)

En adoptant une pratique semblable à celle utilisée dans la mesure F_β de van Rijsbergen (1979), nous utilisons une combinaison des coefficients de corrélation intra-cluster et d'isolation inter-clusters reposant sur le calcul de la moyenne harmonique pondérée des dits coefficients. La combinaison résulte en un seul indice, appelé *"coefficient global de validité"* (OqC), et est donnée par :

$$OqC(\beta) = 2 \times \frac{(\beta^2 + 1)\, CC \times IC}{\beta^2\, CC + IC} \times \frac{|C|}{|\overline{C}|} \tag{6.5}$$

où β est un nombre réel spécifiant l'importance relative du coefficient de corrélation CC au détriment du coefficient d'isolation IC. Une pénalité de $(|C|/|\overline{C}|)$ est imposée pour éviter la présence de clusters vides dans la solution de clustering à retenir. En outre, après la multiplication par un facteur 2, la valeur de OqC est approximativement comprise entre 0 et 1.

Dans Eq. 6.5, le paramètre β est introduit pour prendre en considération le fait que certaines applications accordent plus d'importance à la faculté de former des clusters sans recouvrement, alors que d'autres applications favorisent la faculté de former des clusters bien homogènes que bien séparés. Par conséquent, la valeur du paramètre β doit être fixée selon les besoins spécifiques de l'application du clustering. Généralement, une valeur de 1 accorde une importance égale aux deux coefficients CC et IC, et donc, pourrait être un choix judicieux pour les applications d'exploitation et d'analyse de données. Mais, dans le cas d'autres applications, telles que les résumés automatiques et la compression de données, où l'on exige que les membres des clusters soient très semblables, la valeur de $OqC(\beta)$ devrait être régulée de sorte à mettre en avant CC plutôt que IC (c.-à-d. $\beta < 1$). Par contre, la valeur de $OqC(\beta)$ devrait être régulée de sorte à mettre en avant IC dans le cas d'applications où l'on exige que les clusters soient bien séparés, comme l'étiquetage automatique de collections de données non étiquetées.

La sélection du modèle optimal de clustering est faite en calculant la valeur de l'indice $OqC(\beta)$ pour différentes solutions proposées par une méthode de clustering qui correspondent aux différents nombres de clusters k et en en retenant celle qui maximise la valeur de l'indice $OqC(\beta)$:

$$k_{oc} = \underset{k}{argmax}\{OqC(\beta)|_k\}$$

Interprétation des clusters

Sur la base de notre approche d'identification des variables de type noyau, nous pouvons détecter les clusters non valides, i.e. aberrants ou incohérents, qui pourraient être produits par une méthode de clustering. Ce sont les clusters qui n'ont aucune variable noyau et n'ont donc pas une description spécifique de leur contenu. Ils sont donnés par :

$$\widetilde{C} = \{c \in C \mid CF(c) = \emptyset\} \tag{6.6}$$

La détection de ce type de clusters est très utile pour beaucoup d'applications de clustering. Par exemple, les clusters incohérents dans le cas de la compression de données ne sont pas assez représentatifs de leurs membres pour pouvoir se substituer à eux rigoureusement. Dans un tel cas, les clusters incohérents doivent être traités à part pour subdiviser leur contenu de manière plus fine, ou bien chaque membre de ces clusters doit être considéré comme un cluster singleton.

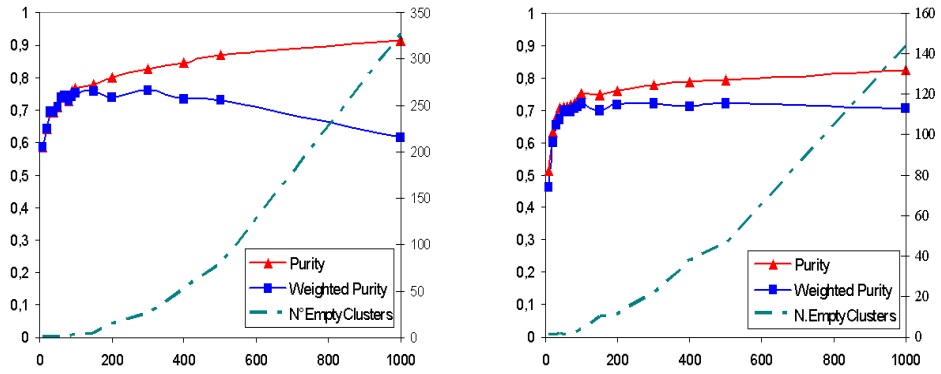
À ce sujet, il est important de noter que la présence de tels clusters est prise en compte dans le calcul des coefficients CC et IC du fait que ce calcul est fait par rapport aux clusters non vides et non pas uniquement par rapport aux clusters valides ($C \setminus \widetilde{C}$).

6.1.5 Évaluation

Dans cette section, nous présentons une évaluation expérimentale qui visait à comparer nos indices de validité à ceux fondés sur la distance qui ont été décrits dans la section 1.5.2 et donc à mettre en évidence l'avantage de nos indices pour la sélection du modèle optimal de clustering dans le cadre des données fortement multidimensionnelles. Nos expérimentations ont été menées sur le corpus *Reuters10* et les différents sous-ensembles issus de la partition temporelle du corpus Reuters-21578 (cf. Annexe A). La transformation des documents du corpus pour les mettre sous la forme vectorielle est réalisée selon des étapes standard de prétraitement : tokenisation, suppression des mots vides et lemmatisation. Les termes d'indexation sont sélectionnés dans le cas non supervisé en fonction des valeurs moyennes de TF-IDF, cf. Section 3.1.3. Le clustering est réalisé par un réseau de type NG (Neural Gas, cf. Section 2.2.3) en augmentant progressivement le nombre de neurones utilisés de n_{min} à n_{max}, et en calculant à chaque fois les valeurs des différents indices étudiés.

L'évaluation des indices de validité est essentiellement basée sur l'indice de pureté des clusters qui permet de vérifier l'homogénéité des clusters produits, c'est à dire de voir si les membres des clusters sont en effet des données appartenant à la même catégorie du corpus de test (cas des clusters homogènes) ou appartenant aux différentes catégories du corpus (cas des clusters hétérogènes). Pour cette évaluation, nous introduisons un facteur de pénalité à la formule originale de pureté (cf. Eq. 1.14) pour prendre en compte la présence des clusters vides dans une solution de clustering et nous appelons la formule qui en résulte *"indice de pureté pondéré"* :

$$WPurity = Purity \times \frac{|C|}{|\overline{C}|} \tag{6.7}$$



(a) (b)

FIG. 6.2 – L'indice de pureté vs. l'indice de pureté pondéré. (a) R-JUN (1068 documents) (b) R-MAR (8087 documents). Cette figure montre l'influence du nombre de clusters vides sur les indices de pureté.

La figure 6.2 montre la différence entre l'indice de pureté de base et l'indice de pureté pondéré. L'évaluation repose sur trois principaux enjeux :

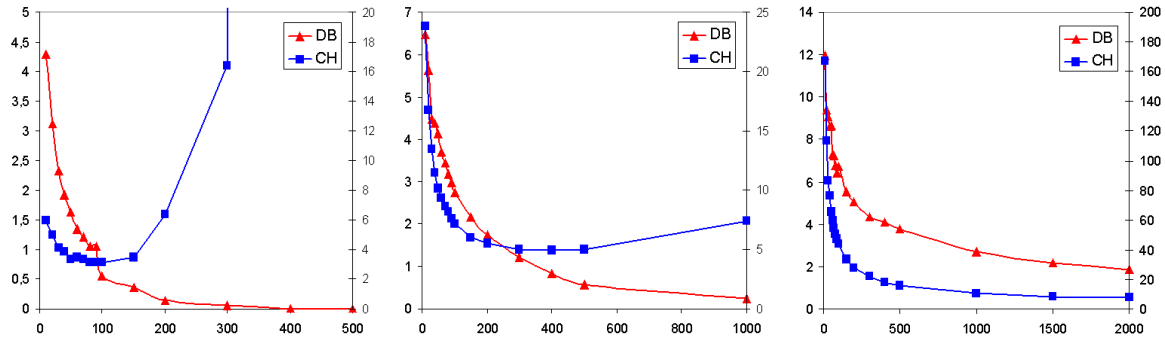
- Le nombre de clusters déterminé en fonction d'un indice de validité devrait être approximativement égal au nombre de clusters déterminé en fonction de la valeur maximale de l'indice de pureté pondéré (ou il devrait se situer dans l'intervalle correspondant aux valeurs maximales de l'indice de pureté pondéré, lorsqu'il n'y a pas une seule valeur maximale clairement définie sur la courbe de pureté) ;
- Le nombre de clusters déterminé en fonction d'un indice de validité devrait être égal ou supérieur au nombre de catégories du corpus en question ;
- Le nombre de clusters invalides ou incohérents devrait être aussi faible que possible.

Présentation et analyse des résultats

La figure 6.3 montre la variation des indices de validité DB et CH en fonction du nombre de clusters produits par la méthode de clustering NG. Rappelons d'abord que le nombre de clusters qui représente la meilleure solution de clustering est celui qui correspond à la valeur minimale de l'indice DB ou la valeur maximale de l'indice CH. À cet égard, il ressort de la figure 6.3 que ces deux indices se comportent de manière inverse dans le cas des données fortement multidimensionnelles : L'indice DB rapporte un nombre très important de clusters ($\approx$ équivalent au nombre de données), alors que l'indice CH rapporte un nombre très faible de clusters ($\approx$ 10 clusters pour tous les corpus de test). Il est aussi à noter sur la figure 6.3 que la valeur de l'indice CH décroît avec l'augmentation du nombre de clusters mais croît à nouveau une fois que le nombre de clusters devient plus important que le nombre de données. Cela indique que cet indice ne peut pas être valide dans une application de fonctionnement en ligne car il peut recommander l'utilisation d'un nombre de clusters supérieur au nombre de données. En tout cas, il est clair que les deux indices, DB et CH, rapportent un nombre de clusters soit trop petit soit très grand, et dans les deux cas très loin du nombre de clusters déterminé selon l'indice de pureté pondéré, cf. Tableau 6.3.

TAB. 6.3 – Le nombre de clusters déterminé selon la valeur maximale de l'indice de pureté pondéré.

Corpus	#Classes	#Clusters	WPurity
R-FEB	38	40	0.7759
R-MAR	86	500 [200,500]	0.7210
R-APR	80	150	0.6834
R-JUN	77	300 [100,300]	0.7599
R-OCT	57	100 [90-150]	0.8489
Reuters10	10	40 [30,500]	0.8170

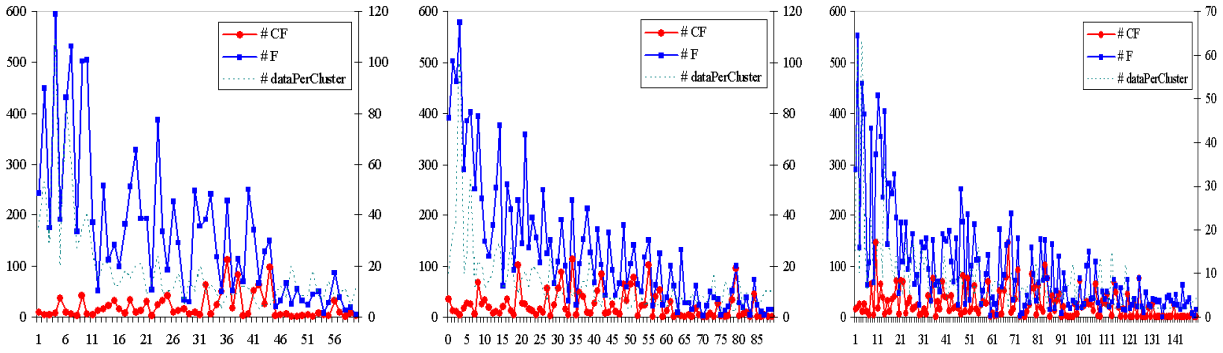


(a) R-FEB (b) R-OCT (c) R-Top10

FIG. 6.3 – Le comportement des indices de validité basés sur la distance dans le cadre de clustering des données fortement multidimensionnelles. L'axe des abscisses indique le nombre de clusters fourni à l'algorithme de clustering, et l'axe des ordonnées indique les valeurs des indices de validité : DB et CH. Plus basse est la valeur de l'indice DB, plus élevée la valeur de l'indice CH, meilleure est la solution de clustering.

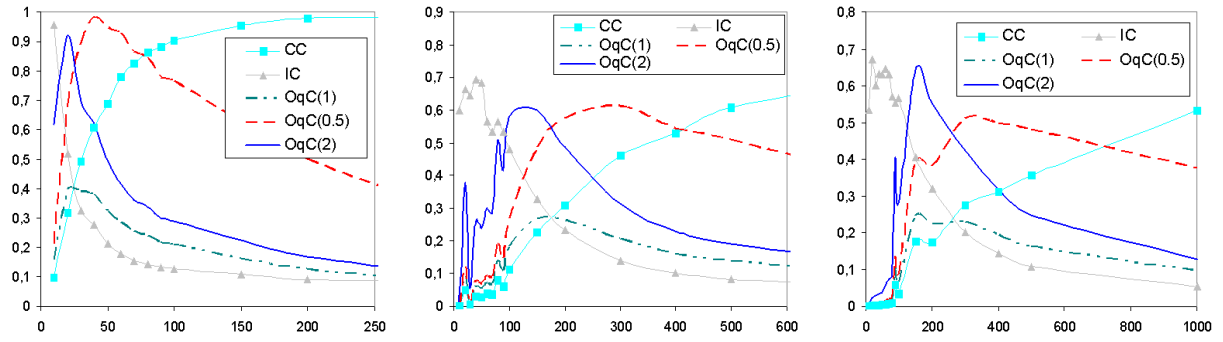
En revanche, les résultats des expérimentations qui ont été menées sur l'indice de validité $OqC(\beta)$ relèvent son potentiel dans le cas des données fortement multidimensionnelles. Le tableau 6.4 récapitule ces résultats pour différentes valeurs du paramètre β. En premier lieu, il ressort de ce tableau que plus la valeur du paramètre β est faible plus le nombre de clusters déterminé par l'indice $OqC(\beta)$ est important. En mettant en parallèle les résultats du tableau 6.4 à ceux du tableau 6.3, le nombre de clusters déterminé en fonction de l'indice $OqC(0.5)$ est plus proche du nombre de clusters déterminé en fonction de l'indice de pureté. Ceci vient du fait que l'indice $OqC(0.5)$ privilégie le coefficient de corrélation intra-cluster CC au détriment du coefficient d'isolation inter-clusters IC. Les indices $OqC(1)$ et $OqC(2)$ offrent des solutions raisonnables en termes de pureté, à l'exception du cas du corpus R-FEB où le nombre de clusters (20) est beaucoup plus petit que le nombre de catégories du corpus (38). Cette sous-estimation du nombre de clusters dans ce dernier cas peut s'expliquer par le fait que beaucoup de catégories du corpus R-FEB sont de taille très petite et peuvent ainsi être regroupées avec des clusters représentatifs des autres catégories (cf. Tableau A.2).

Pour mieux comprendre le rôle du paramètre β, nous reportons sur la figure 6.4 la distribution des variables noyaux dans les clusters dont le nombre est déterminé en fonction des indices



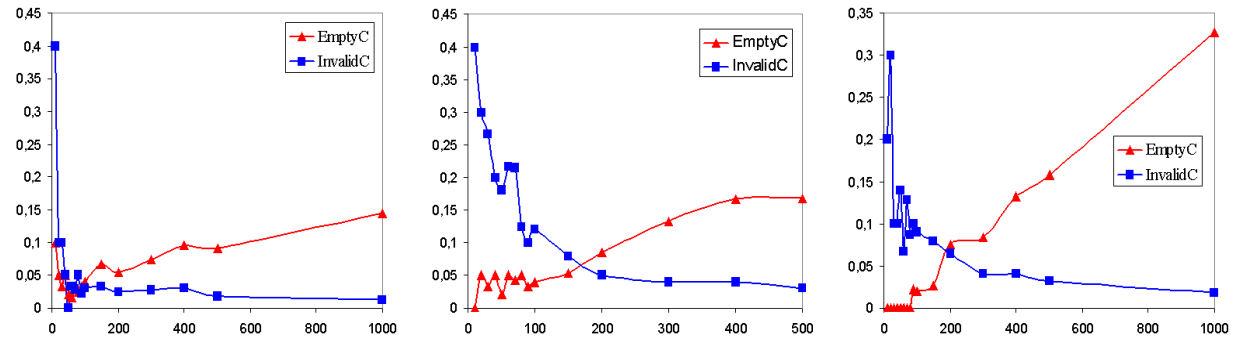
(a) $OqC(2)$: 60 clusters (b) $OqC(1)$: 90 clusters (c) $OqC(0.5)$: 150 clusters

FIG. 6.4 – La distribution des variables de type noyau CF dans les clusters pour les différentes solutions de clustering proposées par NG sur le corpus R-JUN.



(a) R-FEB (b) R-APR (c) R-MAR

FIG. 6.5 – Le comportement de l'indice de validité $OqC(\beta)$ sur trois sous-ensembles du corpus Reuters-21578 : R-FEB, R-APR, et R-MAR



(a) R-MAR (b) R-APR (c) R-JUN

FIG. 6.6 – La proportion du nombre de clusters vides et incohérents identifiés pour différentes solutions de clustering sur trois sous-ensembles du corpus Reuters-21578 : R-MAR, R-APR, et R-JUN

TAB. 6.4 – Le nombre de clusters déterminé selon l'indice de validité $OqC(\beta)$ pour différentes valeurs du paramètre β

		OqC(2)		OqC(1)		OqC(0.5)	
Corpus	#Classes	#Clusters	WPurity	#Clusters	WPurity	#Clusters	WPurity
R-FEB	38	20	0.6757	20	0.6757	40	0.7759
R-MAR	86	150	0.6987	150	0.6987	300	0.7204
R-APR	80	150	0.6834	150	0.6834	300	0.6687
R-JUN	77	60	0.7396	90	0.7435	150	0.7595
R-OCT	57	80	0.8325	100	0.8489	150	0.8474
R-Top10	10	150	0.7997	300	0.7966	400	0.7910

$OqC(2)$, $OqC(1)$ et $OqC(0.5)$. De manière générale, on constate que plus le nombre de clusters est important plus la corrélation intra-cluster est forte. Dès lors, bien que la solution de $OqC(0.5)$ soit la meilleure en termes de pureté, la solution de $OqC(1)$ serait meilleure en termes d'isolation inter-clusters, ce qui est fondamental dans les processus d'analyse des données.

Le variation de la valeur de l'indice $OqC(\beta)$ en fonction du nombre de clusters est montrée sur la figure 6.5. Deux points importants ressortent de cette figure. Premièrement, l'indice $OqC(\beta)$ a toujours un maximum global bien défini qui peut être identifié dès qu'une bonne stabilité du clustering est atteinte. Deuxièmement, il permet de repérer stabilité/instabilité des clusters produits par les différentes solutions de clustering et d'identifier un ensemble de maxima locaux qui pourraient offrir des alternatives intéressantes à la solution optimale.

Pour terminer, nous présentons les résultats en termes de la proportion des clusters vides et incohérents parmi les clusters produits par les différentes solutions de clustering dans la figure 6.6. Le premier constat qui se dégage de ces résultats est assez intuitif : Plus le nombre de clusters est important, plus le nombre de clusters vides est important et plus le nombre de clusters incohérents est réduit. Le deuxième constat est que le nombre de clusters vides et incohérents est très faible ($< 10\%$) pour les différentes solutions déterminées selon l'indice $OqC(\beta)$.

6.2 Approche neuronale à des niveaux d'abstraction multiples

Les réseaux de neurones à apprentissage compétitif sont des outils analytiques très puissants pour le clustering et la préservation de la topologie de la distribution des données, mais sont limités dans le sens où ils ne peuvent pas réaliser les deux tâches en même temps : Lorsqu'ils sont utilisés pour le clustering, chaque neurone est censé représenter un des clusters naturels inhérents aux données. Mais, dans le cas où l'on cherche à identifier la structure topologique de la distribution des données, il est nécessaire de prévoir un nombre plutôt important de neurones (beaucoup plus élevé que dans le cas du clustering) (Rizzo, 2001). La démarche que nous proposons ici permet de réaliser les deux buts avec un seul modèle multi-niveaux : *A-CLN*. L'idée sous-jacente consiste à maintenir un niveau de base reflétant la structure topologique et permettant une représentation assez précise des données. En exploitant les connexions entre les neurones de ce premier niveau (en particulier celles insérées via un apprentissage hebbien compétitif), un ou plusieurs niveaux d'abstraction peuvent être formés pour faciliter une analyse à granularité variable des données et leurs relations.

6.2.1 Principe de mise en œuvre du modèle A-CLN

Le modèle *A-CLN* est fondamentalement un réseau neuronal à deux niveaux : Le premier niveau correspond à un réseau neuronal à apprentissage compétitif dans lequel les connexions entre neurones sont établies via un apprentissage hebbien compétitif (cf. Section 2.2.3). À ce premier niveau, le nombre de neurones devrait être suffisamment élevé de façon à pouvoir précisément capturer les caractéristiques exactes des données et représenter leur structure latente. Dès lors, seules les données très semblables devraient être groupées ensemble et représentées par un seul neurone (ou cluster). Le deuxième niveau est une abstraction du premier niveau dans lequel les neurones du premier niveau sont regroupés dans des neurones d'ordre plus élevé et la structure du premier niveau est reconstituée sous une forme plus simplifiée en exploitant les connexions entre les neurones du premier niveau. Ainsi, la mise en œuvre du modèle *A-CLN* passera par trois grandes étapes successives : 1) Identification des clusters du niveau de base du modèle *A-CLN* via une méthode d'apprentissage de type CL-CHL. 2) Identification des pools des clusters de ce premier niveau ; chaque pool représente donc un cluster du niveau d'abstraction du modèle *A-CLN* (second niveau) et sera représenté par un de ses clusters membres, appelé *"cluster de référence"*. 3) Établissement de connexions entre les clusters du niveau d'abstraction reflétant la structure du premier niveau mais de manière plus simplifiée.

De manière formelle, la mise en œuvre du modèle *A-CLN* peut être décrite comme suit. Soit un réseau neuronal $\mathcal{N}$ appris sur un ensemble de données $\mathcal{X}=\{x_1, x_2, \ldots, x_m\}$, $x_k \in \mathbb{R}^{\mathfrak{p}}$, via un apprentissage de type CL-CHL. Le réseau est caractérisé par :

- Un ensemble de neurones
$$\mathcal{A} = \{c_1, c_2, \ldots, c_n\}$$
avec leurs vecteurs référents
$$W = \{w_1, w_2, \ldots, w_n\}, \quad w_k \in \mathbb{R}^{\mathfrak{p}}$$
qui représentent leurs positions dans l'espace des données d'apprentissage $\mathcal{X}$, chaque neurone c_k représente ainsi un sous-ensemble des données $\mathcal{X}$.
- Un ensemble de connexions $\mathcal{C}$ qui s'établissent entre les neurones via un apprentissage hebbien compétitif ; chaque connexion (i, j) possède un poids correspondant à son âge, noté par $age(i, j)$, qui lui est attaché selon l'apprentissage hebbien compétitif.

Le processus d'abstraction consiste simplement à identifier un ensemble de pools de clusters
$$\mathcal{P} = \{P_1, P_2, \ldots, P_q\}, \quad q < n$$
regroupant les neurones du réseau ($\equiv$ clusters du premier niveau du modèle *A-CLN*). Chaque pool de clusters du niveau d'abstraction P_k comporte au moins un cluster du premier niveau, i.e. son cluster de référence, et probablement d'autres clusters du premier niveau qui lui sont proches.

L'abstraction est principalement réalisée à l'aide des connexions insérées entre les neurones du premier niveau du modèle *A-CLN*. Typiquement, un réseau de neurones comporte un nombre très important de connexions dont beaucoup peuvent entretenir des relations très faibles et non significatives entre les neurones. La présence de telles connexions, en particulier celles entre les neurones appartenant à différents pools de clusters, peuvent rendre l'identification de ces pools difficile. De ce fait, nous procédons parfois à un élagage virtuel du réseau en appliquant un seuil d'élagage ρ sur les connexions entre les neurones. Ce seuil est identifié en retirant progressivement les connexions dont l'âge est le plus important jusqu'à ce qu'un certain nombre de neurones N_{sn}

soient entièrement déconnectés des autres neurones (c.-à-d. ils n'ont aucune connexion avec les autres neurones vérifiant $age \leq \rho$).

Le processus d'abstraction est effectué itérativement sur la base d'un classement des neurones par ordre de voisinage. À chaque étape, un score de voisinage *NS* est calculé pour chaque neurone c_u du premier niveau qui n'a pas encore été assigné à un des pools du second niveau, le neurone est vu dans ce cas comme un *"neurone non marqué"* (Nous notons $\mathcal{M}$ l'ensemble des neurones marqués) ; Le calcul de *NS* est fait comme suit : Pour chaque neurone c_l possédant une connexion valide avec c_u (c.-à-d. une connexion vérifiant $age(u,l) \leq \rho$) si c_l est un neurone non marqué alors

$$c_u.NS \leftarrow c_u.NS + \frac{1}{age(u,l)}$$

Un classement des neurones est ensuite effectué en fonction de leurs scores de voisinage et le neurone c_r (ou les neurones) ayant le plus haut score de voisinage est marqué comme un cluster de référence d'un nouveau pool (ou des nouveaux pools). Mais, pour que ce processus soit validé, il faut vérifier quelques conditions :

1. Le score de voisinage de c_r est supérieur à 1. Cela signifie que le neurone c_r doit au moins être doté d'une connexion très forte avec un autre neurone c_l (i.e. $age(r,l) = 1$) ou des connexions relativement moins fortes mais avec plusieurs neurones (i.e. c_r est un neurone de type centré) ;
2. Le neurone c_r n'est pas un voisin direct d'un neurone marqué c_l (i.e. $C(r,l) = 0$) ; ou il est un voisin direct d'un neurone marqué mais via une connexion non valide (i.e. $C(r,l)$=1 et $age(r,l) > \rho$).

La prise en compte des conditions précitées et la mise à jour des scores de voisinage des neurones sont deux éléments importants pour éviter l'accumulation des clusters de référence dans les régions à forte densité, et en conséquence, pour mieux détecter les différents pools de clusters indépendamment de la densité relative des neurones du premier niveau dans les différentes régions représentées par les neurones. Une fois qu'un neurone est validé comme cluster de référence, un nouveau pool de clusters est créé qui comportera ce neurone avec probablement certains de ses voisins directs ou indirects qui ne sont pas encore affectés à un autre pool de clusters. La profondeur du processus de regroupement est contrôlé par un paramètre de granularité (*GL*). La valeur de ce paramètre sera discutée plus loin après la présentation de l'algorithme d'abstraction du modèle *A-CLN*.

Dans le modèle *A-CLN*, l'abstraction peut être faite selon deux modes différents : macro-abstraction et micro-abstraction. La différence entre ces deux modes tient aux valeurs respectives des paramètres et à l'application d'étapes supplémentaires dans le cas de micro-abstraction :

1. La macro-abstraction consiste à 1) regrouper les neurones du premier niveau dans des pools correspondant aux clusters naturels principaux qui modélisent les données d'entrée et 2) identifier la structure globale inhérente à ces données. Ce mode d'abstraction peut être vu comme un processus de clustering qui ne repose sur aucune connaissance a priori en matière de nombre de clusters présents dans les données.
2. La micro-abstraction est conçue de façon à capturer des clusters et des sous-structures potentielles présents dans les données en fonction d'un certain degré de granularité. Cela permet de plus d'avoir des informations supplémentaires concernant les clusters formés dans le niveau de macro-abstraction. Comme par exemple, l'évaluation de l'homogénéité

des clusters (i.e. vérifier si un cluster est composé de plusieurs sous-clusters ou non) et de la taille des sous-clusters potentiels. De ce fait, ce processus peut aussi être utilisé pour la détection des clusters aberrants ou marginaux qui pourraient être formés dans le niveau de macro-abstraction.

Les pools de clusters formés dans les niveaux d'abstraction peuvent être reliés pour rétablir la structure inhérente des données de manière plus simplifiée. Enfin, une organisation hiérarchique du niveau de base (1er niveau) avec un ou plusieurs niveaux d'abstraction peut être mise à profit pour une analyse à la fois quantitative et qualitative des données.

Ci-dessous, nous décrivons l'algorithme de base du processus d'abstraction dans le modèle *A-CLN* et discutons ensuite comment déterminer les valeurs de quelques paramètres qui lui sont relatifs.

6.2.2 Algorithme

1. Initialiser l'ensemble des pools de clusters (i.e. l'ensemble des clusters du niveau d'abstraction) et l'ensemble des neurones marqués en ensembles vides :

$$\mathcal{P} \leftarrow \emptyset$$

$$\mathcal{M} \leftarrow \emptyset$$

2. Calculer le seuil d'élagage ρ comme suit :
   - Trouver la valeur maximale des âges des connexions ($MaxAge$) et la valeur minimale des âges des connexions ($MinAge$) dans le réseau du niveau de base du modèle *A-CLN*.
   - Initialiser la valeur du seuil à $\rho \leftarrow MaxAge$.
   - Diminuer itérativement la valeur du seuil de 1 jusqu'à pouvoir isoler un nombre minimal de neurones N_{sn}, i.e.

$$|\{c_i \in \mathcal{A} \mid \forall c_l \in \mathcal{A},\ C(i,l) = 0 \vee (C(i,l) = 1 \wedge age(i,l) > \rho)\}| > N_{sn}$$

   Si aucun neurone n'a pu être isolé pour une valeur de $\rho > MinAge{+}1$, remettre le seuil à sa valeur initiale : $\rho \leftarrow MaxAge$.

3. Répéter les étapes suivantes jusqu'à ce que tous les neurones soient marqués, i.e. associés à un des pools formés, ou que leurs scores de voisinage soient inférieurs ou égaux à 1 ($\forall c_i \in \mathcal{A},\ c_i.NS \leq 1$).
   (a) Calculer les scores de voisinage de chacun des neurones c_k :
   - Si $c_k \in \mathcal{M}$ alors $c_k.NS \leftarrow -1$
   - Si $c_k \notin \mathcal{M}$ faire :
     $c_k.NS \leftarrow 0$
     Pour chaque neurone c_l possédant une connexion avec c_k vérifiant $age(k,l) \leq \rho$, et n'étant pas encore marqué comme membre d'un des pools existants, faire :

$$c_k.NS \leftarrow c_k.NS + \frac{1}{age(k,l)}$$

   (b) Classer les neurones par ordre décroissant de leurs scores de voisinage *NS*.
   (c) Soit $\mathcal{T}$ l'ensemble des neurones dont la valeur de *NS* est maximale et supérieure à 1. Pour chacun des neurones $c_t \in \mathcal{T}$ faire les étapes suivantes :

– Si c_t remplit les conditions exigées pour être un cluster de référence, faire :

$$c_t.IsReferenceCluster \leftarrow TRUE$$

$$\mathcal{P}_{newpool} \leftarrow \{c_t\}$$

$$\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}_{newpool}$$

$$\mathcal{M} \leftarrow \mathcal{M} \cup \{c_t\}$$

– Pour chaque neurone $c_k \notin \mathcal{M}$ qui est directement relié à c_t par une connexion dont l'âge vérifie les deux conditions suivantes :

i.
$$\forall\ c_l,\ c_l \notin \mathcal{M}\ \wedge\ c_l \neq c_t\ \wedge\ C(l,k) = 1\ :\ \ age(l,k) \geq age(t,k) - GL$$

ii.
$$\forall\ c_l,\ c_l \notin \mathcal{M}\ \wedge\ c_l \neq c_t\ \wedge\ C(t,l) = 1\ : age(t,k) \leq age(t,l)$$

faire :

$$c_k.IsReferenceCluster \leftarrow False$$

$$\mathcal{P}_{newpool} \leftarrow \mathcal{P}_{newpool} \cup \{c_k\}$$

$$\mathcal{M} \leftarrow \mathcal{M} \cup \{c_k\}$$

Répéter ensuite récursivement la même procédure après le remplacement de c_t par c_k.

Au moment de la fin des étapes ci-dessus, il est possible que certains neurones ne soient pas encore associés à aucun des pools de clusters formés dans le niveau d'abstraction du modèle *A-CLN*. C'est particulièrement le cas des neurones possédant uniquement des connexions non valides ($age > \rho$). Dans le cas de la micro-abstraction, pour tenir compte de ces neurones particuliers $\mathcal{S}$ les étapes (4) à (6) doivent être répétées jusqu'à ce que tous les neurones de $\mathcal{S}$ soient marqués ou qu'ils aient un score de voisinage inférieur à 0 :

4. Recalculer le score de voisinage de chaque neurone $c_s \in \mathcal{S}$ qui n'est pas encore été marqué, comme suit :
   Si $\exists c_l,\ C(s,l) = 1\ \wedge\ c_l.IsReferenceCluster = TRUE\ :\ c_s.NS \leftarrow -1$
   Sinon Pour tout $c_l, C(s,l) = 1 : c_s.NS \leftarrow c_s.NS + \dfrac{1}{age(s,l)}$
5. Classer les neurones $\mathcal{S}$ par ordre décroissant de leurs scores de voisinage *NS*.
6. Pour tout neurone $c_s \in \mathcal{S}$ dont le score est supérieur ou égal à 0 et en commençant par les neurones ayant le score le plus élevé, faire :
   Si c_s n'est pas connecté à un des clusters de référence :

$$c_s.IsReferenceCluster \leftarrow TRUE$$

$$\mathcal{P}_{newpool} \leftarrow \{c_s\}$$

$$\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}_{newpool}$$

$$\mathcal{M} \leftarrow \mathcal{M} \cup \{c_s\}$$

et marquer ensuite les voisins non marqués de c_s de la même manière qu'expliqué à l'étape (3.(c)).

7. Les neurones qui ne sont toujours pas marqués à l'issue des étapes ci-dessus dans le cas de micro-abstraction, ou à l'issue de l'étape 3 dans le cas de macro-abstraction sont associés chacun au pool auquel est associé le neurone le plus proche.
8. (optionnel) Chaque pool de clusters pourrait être représenté au niveau d'abstraction par son cluster de référence. Une alternative est d'utiliser son centroïde comme un vecteur de référence.
9. (optionnel) Les clusters de référence pourraient être reliés l'un à l'autre de manière très simple, consistant à relier les clusters de référence de deux pools de clusters dans le niveau d'abstraction si au moins un des clusters membres du premier pool possède une connexion avec un des clusters membres du deuxième pool dans le niveau de base du modèle *A-CLN*.

Détermination des valeurs des paramètres

- *Le seuil d'élagage ρ et le nombre minimal de neurones à isoler N_{sn}.*

  Un seuil d'élagage est appliqué sur les connexions du réseau du premier niveau du modèle *A-CLN* lors de certaines étapes de notre algorithme afin d'atténuer l'effet négatif des connexions reliant différents pools de clusters. Comme évoqué précédemment, notre stratégie d'ajustement du seuil d'élagage consiste à retirer progressivement les connexions dont l'âge est le plus important jusqu'à ce qu'un certain nombre de neurones N_{sn} soient entièrement déconnectés des autres neurones. Le nombre minimal de neurones N_{sn} est déterminé selon la formule suivante :

$$N_{sn} = (\frac{1}{1 + exp^{-\gamma f}} - 0.5) \times |\mathcal{A}| \tag{6.8}$$

  avec

$$f = \frac{|\mathcal{A}| \times \mathfrak{p}}{|\mathcal{X}|}, \quad \gamma = 0.1$$

  où $|\mathcal{A}|$ représente le nombre de clusters (neurones) du niveau de base du modèle *A-CLN* ; $\mathfrak{p}$ est la dimension de l'espace de représentation des données d'entrée ; et $|\mathcal{X}|$ représente le nombre de données d'entrée. On voit facilement sur l'équation 6.8 que le nombre de neurones à isoler s'élève au maximum à la moitié du nombre total de neurones.

- Le niveau de granularité GL.

  Ce paramètre est conçu pour contrôler le processus de marquage des neurones lors de la création des pools de clusters. L'idée est qu'un certain niveau de corrélation entre les neurones devrait être satisfait pour qu'ils soient regroupés dans le même pool de clusters. Ainsi, pour qu'un neurone soit associé au même cluster qu'un autre neurone, l'âge de la connexion qui les relie ne doit pas être sensiblement supérieur aux âges des autres connexions qui le relient avec d'autres neurones. Il est également apparent que cela est plus exigeant dans le cas de micro-abstraction que dans le cas de macro-abstraction. De ce fait, nous utilisons deux valeurs différentes pour le paramètre GL :

$$GL = \begin{cases} \rho & \text{cas de macro-abstraction} \\ 1 & \text{cas de micro-abstraction} \end{cases} \tag{6.9}$$

6.2.3 Évaluation

Les expérimentations que nous avons menées afin de valider la pertinence du modèle *A-CLN* portent sur des jeux de données synthétiques bidimensionnelles et des données réelles issues du corpus Reuters-21578. Nous présenterons ci-après quelques résultats de ces expérimentations.

Résultats sur des données synthétiques

Les données synthétiques offrent l'opportunité d'illustrer clairement les deux modes d'abstraction, macro-abstraction et micro-abstraction, et l'importance déterminante de leur combinaison. Nous rapportons nos résultats sur deux types principaux de données. D'abord, celles réparties en différents clusters présentant des taux de recouvrement et des densités différentes (jeux de données #1 à #4). Ensuite, celles qui ne sont pas classifiables mais distribuées selon des formes topologiques diverses : distribution discontinue en forme rectangulaire (jeu de données #5), distribution en forme d'anneau (jeu de données #6), et distribution en forme de cactus (jeu de données #7).

Sur la figure 6.7, nous présentons les résultats sur le premier type de données. De gauche à droite, les résultats du processus de macro-abstraction, le nombre de pools de clusters identifiés selon ce mode d'abstraction en fonction du nombre de clusters dans le niveau de base du modèle *A-CLN*, et les résultats du processus de micro-abstraction. De cette figure, on peut tirer les observations suivantes :

- Dans le cas du jeu de données #1 où les clusters sont bien séparés, on remarque que le nombre de clusters présents dans les données est correctement identifié par le processus de macro-abstraction dans environ 100% des cas correspondant à différents nombres de clusters dans le niveau de base du modèle. La micro-abstraction décompose plus finement certains clusters en groupes plus homogènes.
- Sur le deuxième jeu de données, la détection exacte du nombre de clusters naturels présents dans les données semble plus difficile en raison de la forte corrélation entre deux des clusters. En revanche, ces deux clusters sont correctement identifiés par le processus de micro-abstraction.
- Très similaire au cas du deuxième jeu de données, le nombre de clusters présents dans les données du troisième jeu est aussi sous-estimé par le processus de macro-abstraction, mais cette sous-estimation peut être décelée par une analyse simultanée des résultats de la micro-abstraction.
- L'identification du nombre de clusters par le processus de macro-abstraction devient encore plus difficile dans le cas du jeu de données #4 en raison du taux très élevé de recouvrement entre les clusters. Une meilleure identification des différents clusters est obtenue par le processus de micro-abstraction.

Les observations ci-dessus soulignent l'intérêt de considérer de manière simultanée les deux types d'abstraction lors de l'analyse de données. Il convient aussi de noter que, lors de l'utilisation du processus de macro-abstraction comme une méthode de détermination du nombre de clusters présents dans les données, on peut choisir soit le nombre le plus élevé de clusters identifié soit le nombre le plus souvent identifié pour différents nombres de clusters dans le niveau de base du modèle *A-CLN*.

Les résultats obtenus sur le deuxième type de données sont représentés sur la figure 6.8. Cette figure montre le rétablissement de la structure topologique de la distribution des données selon

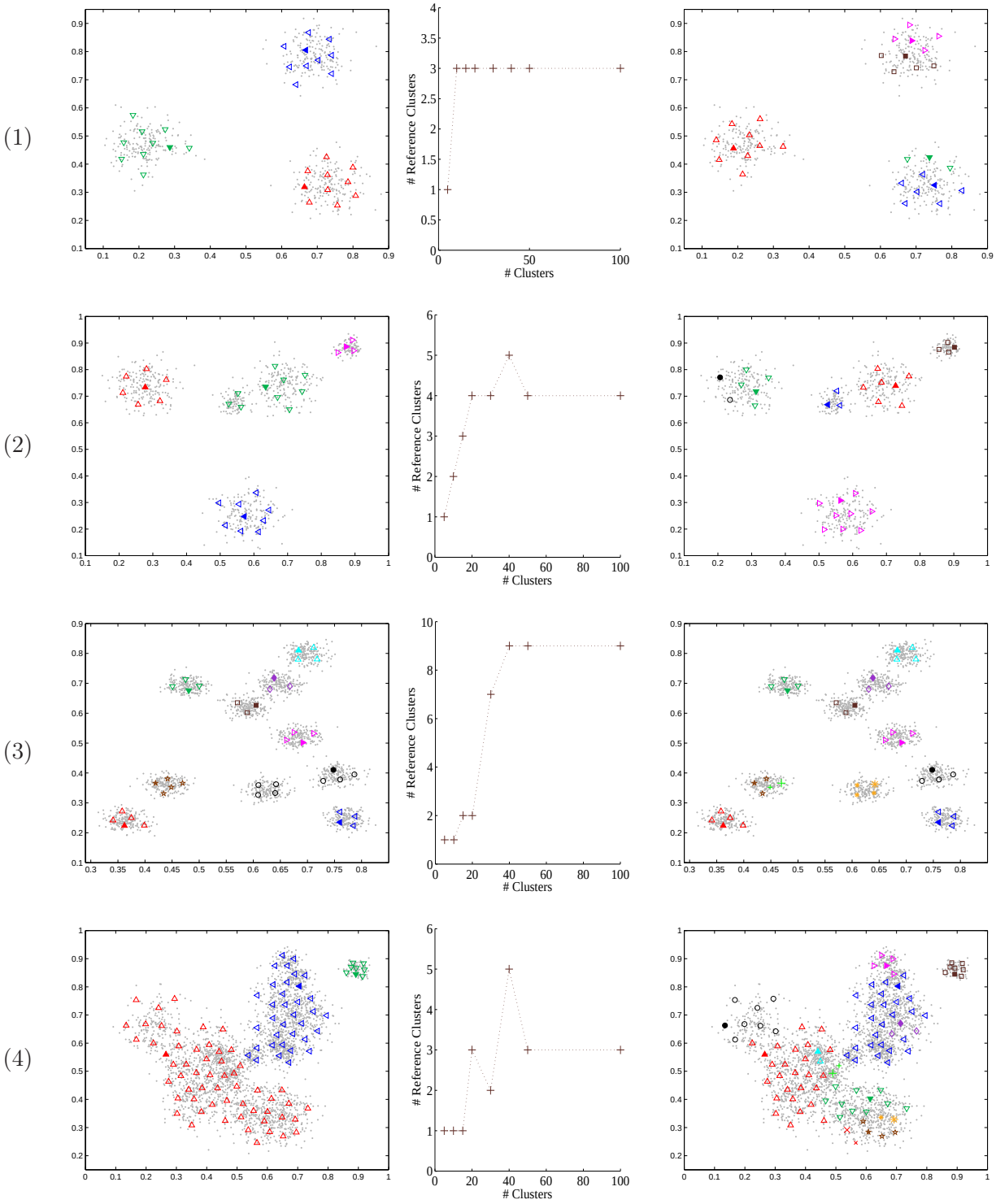


FIG. 6.7 – Une illustration de l'identification des pools de clusters réalisée par le modèle *A-CLN*. La figure montre des exemples sur 4 jeux des données bidimensionnelles réparties en clusters plus ou moins éloignés et de différentes densités. De gauche à droite, les résultats de la macro-abstraction, le nombre de clusters de référence dans le niveau de la macro-abstraction (≡ nombre de pools de clusters) en fonction du nombre de clusters (neurones) dans le niveau de base du modèle *A-CLN*, et les résultats de la micro-abstraction. Les points en gris dénotent les données d'entrée et toutes les formes géométriques dénotent les clusters (neurones) du niveau de base du modèle. Les différentes formes sont utilisées pour dénoter différents pools de clusters dans les niveaux d'abstraction, et les clusters de référence de ces pools sont représentées par des formes pleines ou plus grandes que les autres membres des pools.

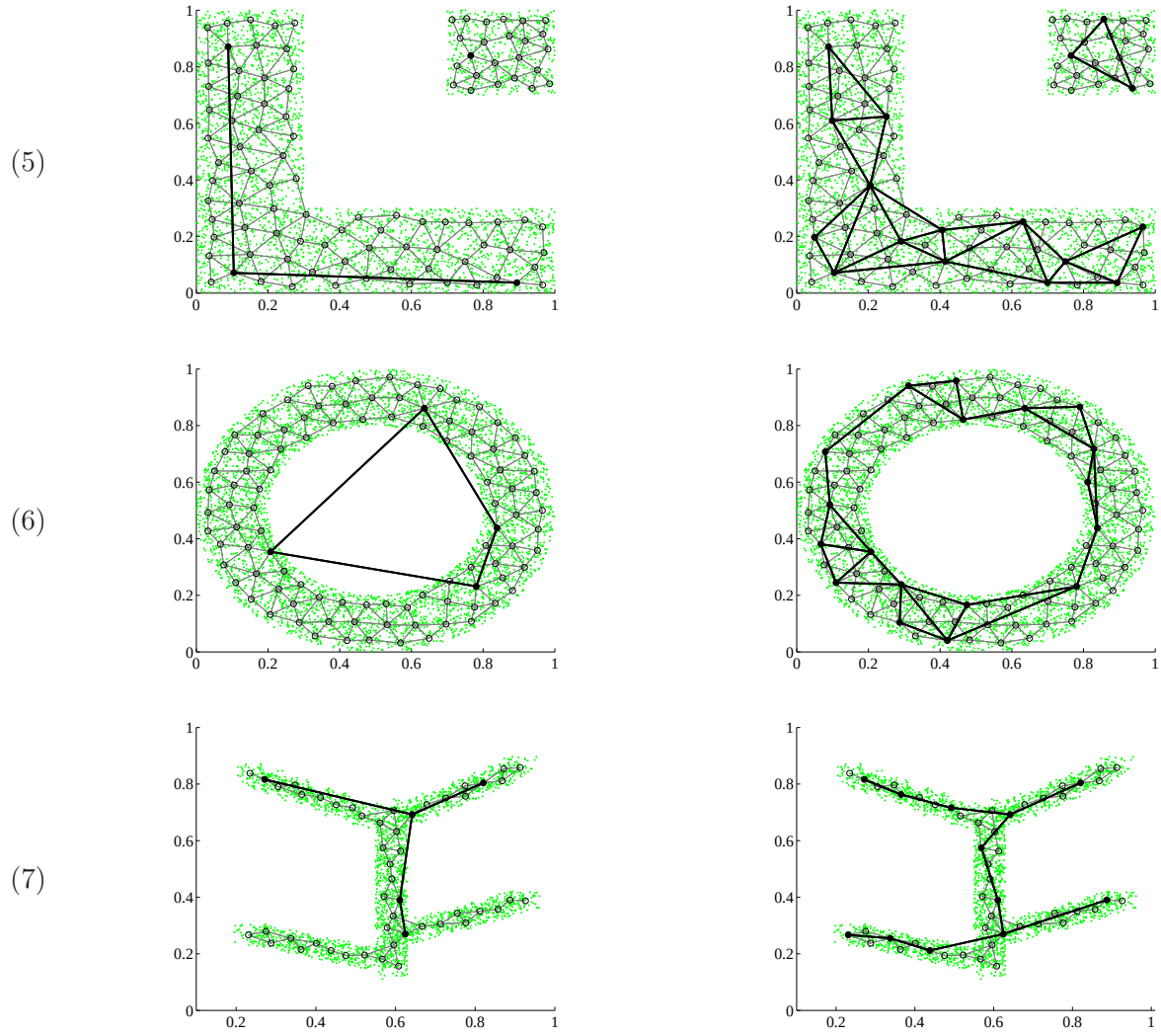


FIG. 6.8 – Le rétablissement de la structure topologique de différentes distributions de données par le modèle d'abstraction *A-CLN*. (1) Une distribution discontinue de données en forme rectangulaire (2) Une distribution de données en forme d'anneau (2) Une distribution de données en forme de cactus. À gauche, la structure topologique selon le mode de macro-abstraction. À droite, la structure topologique selon le mode de micro-abstraction. Les cercles vides dénotent les neurones dans le niveau de base du modèle *A-CLN*, alors que les cercles pleins dénotent les clusters de référence des pools formés dans le niveau d'abstraction.

TAB. 6.5 – Évaluation en termes des mesures de pureté et d'entropie pour les différents niveaux du modèle A-CLN sur les différents sous-ensembles du corpus Reuters-21578. Le nombre de clusters est également présenté pour chaque niveau du modèle A-CLN.

	Niveau de base			Micro-abstraction			Macro-abstraction		
Corpus	#Cls	Purity	Entropy	#Cls	Purity	Entropy	#Cls	Purity	Entropy
R-FEB	40	0.6871	0.3116	6	0.4417	0.9850	4	0.4417	0.9850
R-MAR	300	0.6693	0.3328	42	0.4299	0.8196	10	0.3681	0.9520
R-APR	300	0.6541	0.3135	49	0.4092	0.8726	11	0.3136	1.0823
R-JUN	150	0.7818	0.3584	25	0.6208	0.8734	8	0.5122	1.1437
R-OCT	150	0.8875	0.1628	24	0.7188	0.5811	9	0.6150	0.8108
R-Top10	400	0.8916	0.1619	55	0.6503	0.5574	10	0.4601	0.7480

les deux modes d'abstraction : macro-abstraction (gauche) et micro-abstraction (droite). On peut observer à partir de ces résultats que les deux modes d'abstraction peuvent présenter de manière plus simplifiée la structure topologique des données à partir de la structure originale identifiée par le réseau de neurones du niveau de base. La macro-abstraction ne représente que les caractéristiques principales de la structure topologique des données alors que la micro-abstraction rend davantage de détails sur cette structure, ce qui renforce à nouveau l'intérêt de combiner les deux modes d'abstraction.

Résultats sur des données réelles

Ici, nous rapportons les résultats d'expérimentations effectuées sur des données réelles issues du corpus *Reuters10* et de la partition temporelle du corpus *Reuters-21578* (cf. Annexe A). Ces expérimentations ont pour objectif de montrer la différence entre les deux modes d'abstraction en matière de degré d'homogénéité intra-cluster et d'hétérogénéité inter-clusters et l'intérêt potentiel de leur combinaison avec le niveau de base.

Tout d'abord, nous utilisons comme mesures d'évaluation la pureté et l'entropie (cf. Section 1.5.1). La pureté permet de caractériser l'homogénéité des clusters produits à chaque niveau du modèle *A-CLN*, alors que l'entropie permet de caractériser la répartition des différentes catégories à l'intérieur des clusters. Selon les résultats du tableau 6.5, on voit que le niveau de base offre les meilleurs résultats en termes de pureté et d'entropie dû au nombre important de clusters déterminé selon l'indice $OqC(0.5)$. La diminution du nombre de clusters dans les niveaux d'abstraction entraîne une diminution relative de la pureté et une augmentation relative de l'entropie des clusters. Ces variations permettent d'avoir différentes vues sur les relations entre les données.

Les variations entre les différents niveaux du modèle *A-CLN* sont davantage marquées en fonction des coefficients de corrélation intra-cluster (CC) et d'isolation inter-clusters (IC). On observe, au vu des résultats du tableau 6.6, que la valeur du coefficient CC diminue, tandis que celle du coefficient IC augmente, en procédant du niveau de base vers l'un des niveaux d'abstraction. Ces résultats valident notre objectif et justifient la proposition du modèle *A-CLN*.

TAB. 6.6 – Évaluation en termes des coefficients de corrélation intra-cluster (CC) et d'isolation inter-clusters (IC) pour les différents niveaux du modèle A-CLN .

	Niveau de base			Micro-abstraction			Macro-abstraction		
Corpus	#Cls	CC	IC	#Cls	CC	IC	#Cls	CC	IC
R-FEB	40	0.5950	0.2356	6	0.1841	0.6375	4	0.3214	0.4688
R-MAR	300	0.2917	0.2026	42	0.3163	0.6531	10	0.0053	0.7658
R-APR	300	0.4580	0.1324	49	0.3280	0.5671	11	0.1207	0.6114
R-JUN	150	0.4555	0.1776	25	0.1661	0.7442	8	0.0027	0.6250
R-OCT	150	0.5284	0.1712	24	0.1410	0.7923	9	0.0296	0.6667
R-Top10	400	0.2392	0.1181	55	0.1017	0.4045	10	0.0035	0.4254

6.3 Étiquetage des clusters

L'étiquetage des clusters est une nécessité absolue pour une interprétation à la fois intuitive et synthétique du résultat de clustering. Cependant, il n'existe que très peu de travaux qui se sont concentrés sur le problème d'étiquetage des clusters. L'approche générale consiste à choisir un ou plusieurs termes représentatifs qui caractérisent le mieux toutes les données membres de chacun des clusters et qui peuvent ainsi servir à définir leurs étiquettes. Pour ce faire, il existe principalement trois stratégies d'étiquetage :

1. L'étiquetage basé sur le vecteur référent associé au cluster dont le principe est d'attribuer au cluster le ou les termes dominants dans son vecteur référent (Lin et al., 1991) ;
2. L'étiquetage basé sur les vecteurs représentatifs des données membres du cluster (Cutting et al., 1993). Le principe est d'étiqueter chaque cluster par le nom du ou des termes les plus fréquents dans l'ensemble de ses données membres.
3. L'étiquetage basé sur les méthodes de sélection de variables qui consiste à comparer la distribution des termes dans un cluster à celle dans les autres clusters, comme la statistique de χ^2 (Treeratpituk et Callan, 2006).

Un problème commun aux stratégies précitées est de déterminer le nombre d'étiquettes à retenir pour chacun des clusters et d'identifier les clusters pertinents. De plus, les étiquettes sont souvent inutiles pour la compréhension du contenu sémantique des clusters.

Pour le modèle *A-CLN*, nous attribuons à chaque cluster un maximum de n étiquettes correspondant aux termes noyaux les plus corrélés au contenu du cluster en question. Dans le cas où le nombre de termes noyaux est supérieur à n, les termes noyaux sont triés en ordre décroissant en fonction des scores de corrélation $CS(f)$ et les top n termes sont attribués comme étiquettes. L'ensemble des étiquettes E_{c_i} attribuées à un cluster c_i peut être défini comme suit :

$$\begin{array}{ll} \text{Si } |CF(c_i)| \leq n \text{ alors} & E_{c_i} = \{f \mid f \in CF(c_i)\} \\ \text{Sinon} & E_{c_i} = \{f \mid f \in Top(n, CF(c_i), \leq_{CS(f)})\} \end{array}$$

Les résultats de cette stratégie d'étiquetage sont comparés dans le tableau 6.7 à ceux obtenus par une méthode basé sur la sélection des termes qui ont les plus forts poids dans le vecteur référent de chacun des clusters. L'étiquetage est appliqué à des clusters formés dans le niveau de base du modèle *A-CLN* dans le cas du corpus R-OCT.

TAB. 6.7 – Comparaison des étiquettes attribuées à des clusters formés dans le niveau de base du modèle *A-CLN* par la méthode basée sur les termes noyaux (CFL) et par la méthode basée sur les termes qui ont les plus forts poids dans le vecteur référent de chacun des clusters (RCL) dans le cas du corpus R-OCT.

Cluster	CFL	RCL
1	interest march year repres cost	gain includ rev respect dlr
2	reduc rev exclud earn effect	end septemb sale period sept
3	rev mth	rev mth farm corp dlr
4	capit control amount make acquir	total incom june capit corp
5	deficit equiti unlik growth billion	inflat rate bank govern economist
6	credit rev	primari dilut shr rev dlr
7	dlr polici merger tender ad	equiti pacif common stock file
8	financi group repres offic fund	board meet compani committe propos
9	corp mth dlr	corp mth dlr western sale
10	contract price transact increas sale	steel report sale concern contract
11	grain last harvest record analyst	soviet tonn sugar estim year
12	dilut loss provis quarter loan	dilut loss provis quarter loan
13	street trade hous texa point	bill french trade hous foreign
14	payment note dlr pension repres	loss profit prefer payment dividend
15	acquir corp pacif condit comment	offer world corp acquir pacif
16	rate bank	prime rate bank rev sept
17	corn senat propos wheat reduct	limit loan rate requir reduct
18	tonn trade dealer estim busi	corn sold tonn report trade
19	profit primari dilut includ gain	primari dilut profit loss gain
20	weinberg ship mondai destroi	platform iranian attack iran mile
21	bancorp third asset agreement dlr	bancorp third data asset agreement
22	malaysia govern economist budget sector	malaysia govern output budget sector
23	global rais dlr petroleum	barrel global rais dlr petroleum
24	market dollar lower	lower dollar bui market bundesbank
25	system reserv feder repurchas economist	feder reserv system repurchas fund
26	share	group share corp sector agre
27	barrel texa post crude increas	barrel texa increas post bring
28	barrel crude	crude barrel dlr grade texa
29	crude dlr	crude dlr barrel grade texa
30	bancorp dlr mth	bancorp dlr mth data asset

Toutes les méthodes et techniques présentées jusqu'à maintenant entrent dans le cadre purement statique d'analyse de données. Pour prendre en compte l'aspect temporel d'un flux de données, nous appliquons ces méthodes, en particulier, notre modèle A-CLN, sur différentes fenêtres temporelles du flux de données. Des mécanismes de mise en correspondance sont ensuite établis pour rendre possible des comparaisons entre les clusters issus de ces fenêtres temporelles, le suivi de l'évolution des clusters et l'évaluation de l'importance de changements potentiels survenant dans le temps. Ces étapes seront présentées en détail dans la section qui suit.

6.4 Suivi de l'évolution des clusters à travers le temps

6.4.1 Mise en correspondance

Dans le but primaire de pouvoir supporter une analyse de l'évolution des clusters en termes d'émergence ou de disparition, nous nous intéressons ici au développement d'une méthode de comparaison des résultats de clustering obtenus à partir de différentes fenêtres temporelles d'un flux de données. Les ensembles de données correspondant à ces fenêtres peuvent donc être partiellement ou entièrement différents. De même, les espaces de représentation des données dans les différentes fenêtres temporelles peuvent être partiellement ou entièrement différents. Il est donc aisément concevable que les nombreuses mesures existantes qui ont été mises en place pour calculer la similarité/dissimilarité entre les résultats obtenus par différentes méthodes de clustering, telles que l'indice de Rand (Rand, 1971) et l'indice de Jaccard (Hamers et Hemeryck, 1989), ne peuvent pas être adoptées dans notre cas. La raison est qu'elles sont conçues pour comparer les résultats de clustering obtenus sur le même ensemble de données et s'appuient ainsi sur l'analyse de la distribution des données dans les clusters produits par les différentes méthodes de clustering.

Notre méthode est basée sur une mise en correspondance entre les clusters construits dans le niveau de base du modèle *A-CLN* correspondant à une fenêtre active (T_n) et les micro-clusters construits dans le niveau de micro-abstraction du modèle *A-CLN* correspondant à une fenêtre précédente (T_{n-k}). Nous utilisons donc les clusters du niveau de base du modèle *A-CLN* en tant que données et les micro-clusters en tant que catégories ou que classes. La modélisation des micro-clusters est faite en utilisant deux modèles ILoNDF par micro-cluster : Le premier est appris à partir des profils des clusters qui y sont associés (exemples positifs d'apprentissage) et le second modèle ILoNDF est appris à partir des profils des clusters qui sont associés aux macro-clusters autres que celui qui contient le micro-cluster en question (exemples négatifs d'apprentissage).

Pour réaliser la mise en correspondance entre les clusters de deux fenêtres temporelles, la manière la plus simple et directe est d'assigner chacun des clusters construits à partir d'une des fenêtres au micro-cluster de l'autre fenêtre qui lui correspond le plus au sens d'une mesure de corrélation (e.g. similarité cosinus). Cependant, il est évident que des clusters pourraient être associées à des micro-clusters qui ne leur correspondent pas vraiment ou pas du tout. Pour régler ce problème, il faut fixer un seuil de corrélation au-dessous duquel le cluster ne sera assigné au micro-cluster qui semble lui correspondre le plus. À cet effet, nous procédons comme suit. Pour chaque micro-cluster μ_i, nous calculons la valeur moyenne (m_{μ_i}) et l'écart-type (σ_{μ_i}) des scores de corrélation entre le profil du micro-cluster et ses membres clusters. Le seuil est fixé à :

$$\tau_{\mu_i} \; = \; m_{\mu_i} \; - \; k \; \sigma_{\mu_i} \tag{6.10}$$

Donc, un cluster c_l sera associé au plus proche micro-cluster trouvé dans l'autre fenêtre μ_i seulement si leur similarité dépasse le seuil τ_{μ_i} fixé pour ce micro-cluster.

6.4.2 Quantification de changements

Maintenant que nous avons mis en vidence les relations entre les clusters issus de deux différentes périodes de temps, T_n et T_{n-k}, il est possible de quantifier facilement les changements qui pourraient être survenir entre ces périodes. À cet effet, nous distinguons cinq types de changement :

- Apparition de nouveaux clusters : Un cluster de la fenêtre active T_n est qualifié de nouveau si : 1) il ne possède aucune relation avec un micro-cluster de la fenêtre précédente T_{n-k}, et 2) il appartient à un macro-cluster dont tous les clusters membres n'ont aucune relation avec les micro-clusters de la fenêtre précédente. Formellement :

$$\begin{array}{ll} \text{Si} & \forall c_i \in \mathcal{P}_t^{Macro} \text{ de } T_n \wedge \quad \forall c_j \in \mathcal{P}_s^{Micro} \text{ de } T_{n-k} \ , \quad C(c_i, c_j) = 0 \\ \text{Alors} & c_i \ \text{ est nouveau.} \end{array}$$

- Disparition d'anciens clusters : De manière analogue, un cluster de la fenêtre précedente T_{n-k} est considéré comme disparu si : 1) il possède aucune relation avec un micro-cluster de la fenêtre active T_n et 2) il appartient à un macro-cluster dont tous les clusters membres n'ont aucune relation avec les clusters de la fenêtre active. Formellement :

$$\begin{array}{ll} \text{Si} & \forall c_i \in \mathcal{P}_s^{Macro} \text{ de } T_{n-k}, \quad \forall c_j \in \mathcal{P}_t^{Micro} \text{ de } T_n \ , \quad C(c_i, c_j) = 0 \\ \text{Alors} & c_i \ \text{ est considéré comme disparu.} \end{array}$$

- Fusion d'anciens clusters : Un micro-cluster de la fenêtre active μ_t est une fusion de deux ou plusieurs micro-clusters de la fenêtre précédente si : 1) tous les clusters membres de μ_t sont liés à un de ces micro-clusters, et 2) tous les clusters membres de ces micro-clusters ne possèdent que des relations avec μ_t ou ne possèdent aucune relation avec les micro-clusters de la fenêtre active.

- Éclatement d'anciens clusters : De manière inverse à celle d'une fusion, un micro-cluster de la fenêtre précédente μ_s est éclaté en deux ou plusieurs micro-clusters de la fenêtre active si : 1) tous les clusters membres de μ_s sont liés à un de ces micro-clusters, et 2) tous les clusters membres de ces micro-clusters ne possèdent que des relations avec μ_s ou ne possèdent aucune relation avec les micro-clusters de la fenêtre active.

- Développement ou sous-développement d'anciens clusters : Un micro-cluster de la fenêtre précédente μ_s qui s'est développé ou sous-développé dans la fenêtre active est un micro-cluster qui n'a pas subi de fusion ou d'éclatement et qui vérifie les deux propriétés suivantes : 1) tous les clusters membres de μ_s sont liés au même micro-cluster de la fenêtre active μ_t, 2) les clusters membres du micro-cluster μ_t sont lié soit au micro-cluster μ_s (nous désignons cet eensemble de clusters par $S \subset \mu_t$) soit à aucun des micro-clusters de la fenêtre précédente (nous désignons cet ensemble de clusters par $N \subset \mu_t$). Le degré de développement peut ainsi être exprimé par :

$$d_{\mu_s} = \frac{|S| - |\mu_s|}{|S| + |N|}$$

Pour rendre compte de ces différents types de changement, il est primordial d'intégrer à notre modèle des fonctionnalités de visualisation dont il existe plusieurs variantes. Citons, à titre d'exemples, la visualisation par projection cartographique Kaski et al. (1998), la visualisation des graphes (Battista et al., 1999), et la visualisation hyperbolique (Ritter, 1999; Ontrup et Ritter, 2006). Développer une telle visualisation est une perspective importante de notre travail.

6.5 conclusion

L'introduction d'une stratégie d'analyse des données issues de différentes périodes temporelles est une voie à haut potentiel dans des domaines demandant une veille régulière des changements dans le temps. Par exemple, dans le domaine de la veille scientifique, des outils pour analyser et suivre l'évolution de thématiques de recherche d'un domaine particulier à partir de ses publications scientifiques sont nécessaires pour évaluer les avancées et identifier des problèmes émergents. Bien que la stratégie que nous avons proposée fasse recours, dans sa version actuelle, à des méthodes statiques de clustering, elle promet d'aboutir à une analyse temporelle grâce à notre choix de maintenir des niveaux multiples de granularité et d'abstraction. Ce choix est entièrement original et bien adapté pour permettre des comparaisons entre deux périodes de temps bien distinctes. Il reste aussi intéressant d'appliquer le principe du modèle A-CLN pour la surveillance en ligne des changements survenant sur les flux de données.

Conclusion

L'émergence des applications orientées flux de données s'est accompagnée non seulement d'une indispensable révision des méthodes traditionnelles d'analyse de données, mais aussi d'une nécessité absolue de mettre en place de nouvelles méthodes adaptées à leurs spécificités et leurs exigences propres. Il s'agit plus particulièrement des méthodes adaptées au traitement en ligne des flux de données en temps quasi réel. Ces méthodes doivent prendre en compte, d'une part, l'important volume des données issues des flux et, d'autre part, l'aspect dynamique et évolutif de tels flux. Dans ce contexte, de manière générale, l'objectif de ce travail de thèse est de proposer un cadre fédérateur pour l'analyse des flux de données changeant au cours du temps. Le cadre principal d'application envisagé est celui du traitement des données en provenance du web ou d'autres sources documentaires accessibles en ligne. Les données documentaires sont en effet réputées difficiles à traiter, car elles s'organisent la plupart du temps selon des distributions à la fois fortement multidimensionnelles, éparses, et très bruitées. Le choix du traitement de ces données représente donc une option intéressante pour généraliser les méthodes proposées à tous les types de données multidimensionnelles. De plus, le type d'analyse envisagé a pu bénéficier des premières applications importantes dans le domaine particulièrement porteur des systèmes de filtrage personnalisé d'informations, en l'occurrence, le système de diffusion ciblée de sites web par satellite CASABLANCA.

Selon une logique plus précise de déroulement, les principales contributions de notre travail de thèse peuvent être résumées comme suit.

Dans un premier temps, un état de l'art est conçu à partir d'une analyse bibliographique et d'une description des travaux de recherche les plus pertinents relatifs au domaine de l'analyse des flux de données. Cet état de l'art est organisé en trois parties :

- La première partie, introductive, présente les tendances, les défis, et les aspects fondamentaux relatifs à l'analyse des flux de données, mais aussi, quelques généralités sur l'analyse de données statiques et les méthodes classiques associées, la priorité ayant été donnée jusqu'à présent à repenser les solutions existantes afin d'étendre leur cadre actuel de manière à prendre en compte l'ensemble des caractéristiques inhérentes aux flux de données. La constatation que nous avons pu faire est que la plupart des méthodes classiques sont conçues de manière à fonctionner selon un mode d'apprentissage hors ligne à partir des données statiques sur lesquelles plusieurs passages sont possibles. De plus, ces méthodes sont fondées sur certaines hypothèses, comme celle qui suppose que les données sont indépendantes et identiquement distribuées, qui ne peuvent plus se soutenir dans le cadre des flux de données évolutifs.
- La deuxième partie est dédiée à la présentation des différentes méthodes utilisées dans le domaine de l'analyse des flux de données, en insistant plus particulièrement sur le cas des flux de données multidimensionnelles. Dans ce cadre, nous retrouvons deux grandes

directions. La première direction s'intéresse particulièrement à la constitution de résumés de l'historique de flux de données sur lesquels on peut appliquer des méthodes classiques d'analyse de données. Parmi les méthodes possibles, nous avons privilégié les méthodes de clustering puisqu'elles peuvent s'appliquer efficacement au cas des flux de données multi-dimensionnelles. La seconde direction cherche plutôt à étendre les méthodes classiques ou bien à proposer de nouvelles méthodes appropriées à l'analyse de flux de données. Nous avons présenté à la fois les méthodes utilisées pour la classification supervisée et non supervisée :

- Dans le cadre de la classification non supervisée, nous avons mis en évidence l'absence des méthodes adaptées à un mode de fonctionnement en ligne, et la non-prise en compte de la composante temporelle des données lors de l'analyse des flux de données. Nous avons donc introduit les méthodes connexionnistes comme des méthodes potentiellement puissantes pour l'intégration de la composante temporelle des données. Pour l'instant, l'application de ces méthodes dans le contexte des flux de données peut être réalisée à l'aide des fenêtres temporelles ou sur des résumés représentant l'historique des flux de données, en attendant de pouvoir proposer des extensions ou des variantes adaptées au traitement en ligne des flux de données. De l'ensemble de ces méthodes, nous avons retenu les méthodes à topologie adaptative qui permettent, d'une part, de bien s'adapter au traitement de données multidimensionnelles complexes et, d'autre part, de prendre bien en compte l'évolution des flux non-stationnaires et leur aspect temporel.
- Lors de la présentation des méthodes de classification supervisée, nous avons surtout mis l'accent sur le problème de la dérive de concept et les différentes approches mises en œuvre pour tenter de le résoudre. À ce titre, nous avons vu que les approches qui reposent sur un apprentissage adaptatif en ligne représentent la solution la plus évidente à ce problème, cependant, très peu de méthodes permettent aujourd'hui d'envisager de tels types d'apprentissage. Pour cela, la sélection de données à l'aide de fenêtres temporelles et les approches à base d'ensemble de classifieurs restent à ce jour les solutions les plus fréquemment employées dans le cadre des flux de données évolutifs.

Nous avons enfin insisté sur le problème de la détection de changements qui pourraient intervenir sur les flux de données. Ce problème, encore peu investi par manque de méthodes efficaces adaptées aux particularités des flux de données, a été abordé dans le cadre général de la classification à partir d'une seule classe. À ce titre, nous avons vu deux directions principales pour remédier au problème d'absence d'exemples négatifs dans ce type de classification. La première direction cherche à déduire artificiellement des exemples négatifs à partir d'un ensemble de données non étiquetées. Les approches adoptées pour ce faire présentent des limitations sérieuses faisant obstacle à leur application dans le cadre des flux de données. Quant à la seconde direction, l'apprentissage est effectué directement à partir des exemples positifs uniquement. Les méthodes associées à cette direction, semblant un choix plus évident dans le cas des flux de données, ne sont pas immédiatement compatibles avec les contraintes de traitement des flux de données. Seul le modèle du filtre détecteur de nouveauté (NDF) nous paraissait théoriquement susceptible de satisfaire à ces contraintes. Nous l'avons donc retenu comme point de départ vers la conception d'un nouveau modèle, ILoNDF, plus performant et plus adapté au cas des flux de données évolutifs.

- La troisième et dernière partie de l'état de l'art est consacrée à la description du cadre applicatif de notre travail, à savoir le filtrage d'informations. Nous avons présenté les différents types de filtrage, en faisant notamment état des fondements du filtrage basé sur le contenu. Dans ce cadre, nous avons pu constater que la majorité des méthodes proposées dans la littérature sont inspirées des méthodes utilisées habituellement en recherche

d'informations. Les méthodes de modélisation du profil utilisateur ne se focalisent donc sur l'analyse du contenu des documents fournis par l'utilisateur que pour construire son profil, le plus souvent sous forme de listes de termes pondérés, de façon analogue à celle des requêtes. Or, à notre avis, pour bénéficier de toute la richesse du contenu des documents, il est nécessaire de moduler l'analyse des documents, de tirer des conclusions sur le type de besoin d'informations de l'utilisateur (ex. précis, thématique ou exploratoire) et d'évaluer la cohérence des décisions prises par l'utilisateur au regard de la pertinence des documents délivrés par le système. L'intégration de telles connaissances sur le besoin de l'utilisateur dans le système de filtrage l'aide sûrement à délivrer des informations plus proches des attentes réelles de l'utilisateur.

L'état de l'art que nous avons présenté peut paraître un peu étendu, mais il convient de ne pas perdre de vue qu'il se rapporte à plusieurs domaines qui, même s'ils sont complémentaires, restent distincts. De plus, il nous paraissait particulièrement intéressant d'isoler les méthodes utilisées ou potentiellement utilisables dans le domaine des flux de données, celui-ci étant encore en plein développement et un tel état de l'art n'étant pas, à notre connaissance, proposé à ce jour par quelqu'un d'autre.

C'est à partir de l'étude de cet état de l'art que nous avons pu constater un manque évident de méthodes dotées d'un fonctionnement en ligne adaptées aux spécificités du traitement des flux de données, et ceci notamment au niveau de la détection des changements susceptibles de survenir sur les flux de données proprement dits. Nous sommes donc plus particulièrement intéressés à la mise en place d'un cadre fédérateur pour l'analyse des flux de données changeant au cours du temps. Dans ce but, nous considérons plusieurs contextes d'apprentissage selon les connaissances dont on dispose a priori sur les données issues des flux, ce qui correspond aux différentes tâches de classification : classification à partir d'une seule classe, classification supervisée et classification non supervisée (clustering).

Comme nous l'avons déjà mentionné, notre point de départ était le modèle de filtre détecteur de nouveauté (NDF). Ce modèle est en effet facile à mettre en œuvre selon un mode de fonctionnement en ligne et sans répétition d'apprentissage, ce qui répond aux contraintes les plus limitantes liées au traitement des flux de données. La règle d'apprentissage associée à ce modèle souffre cependant de faiblesses intrinsèques qui constituent des obstacles sérieux à son application efficace. Il s'agit principalement de l'habituation très rapide de ce modèle aux données d'apprentissage et à leurs variables, ce qui le rend particulièrement sensible au bruit qui pourrait être présent dans la description des données d'apprentissage. De plus, son incapacité à oublier des données déjà apprises rend inefficace le traitement en ligne des données issues des flux évolutifs où la distribution des données change au cours du temps. C'est pour ces raisons que nous avons apporté des modifications à la règle d'apprentissage du modèle NDF, ainsi que de nouvelles stratégies pour l'exploitation du modèle pour des fins de classification, qui sont reportées dans un nouveau modèle, ILoNDF (Incremental data-driven Learning of NDF). C'est particulièrement ce modèle qui constitue le cœur de notre travail et qui nous a permis de répendre aux contraintes les plus exigeantes dans le domaine des flux de données.

Notre modèle conserve tous les avantages du modèle NDF ainsi que le principe de la détection de nouveauté avec son mode d'apprentissage qui n'exige typiquement que des exemples positifs issus d'une classe pour apprendre un modèle de classification. Un des effets majeurs de la modification que nous avons apportée à la règle d'apprentissage du modèle original tient à la capacité

du modèle ILoNDF à acquérir constamment de nouvelles connaissances relatives aux fréquences d'occurrence des variables et à leurs dépendances de co-occurrence dans les données utilisées pour faire l'apprentissage, ce qui rend le modèle robuste au bruit. Un autre aspect intéressant qui est survenu suite à la modification de la règle d'apprentissage du modèle NDF, concerne la capacité du modèle ILoNDF à incorporer de manière implicite quelques effets d'oubli sur les données occasionnelles ou anciennes, ce qui l'aide à mieux s'adapter au changement de la distribution des données au cours du temps. En outre, le modèle ILoNDF ne comporte aucun paramètre à régler avant ou pendant l'apprentissage ; il n'y a donc aucun besoin de faire des calculs supplémentaires et coûteux en matière d'optimisation de paramètres. La validation du modèle ILoNDF pour des fins de classification à partir d'une seule classe a été opérée sur deux collections standard de données textuelles : Reuters et WebKB (cf. Chapitre 4). Les résultats expérimentaux obtenus ont prouvé à plusieurs reprises la supériorité du modèle ILoNDF vis-à-vis de l'ensemble des méthodes proposées jusqu'à aujourd'hui, notamment en termes de performance et de robustesse vis-à-vis des différents aspects expérimentaux tels que les schémas de pondération et la dimensionnalité de l'espace de représentation des données.

Dans le cadre général de la classification supervisée à deux classes, nous avons proposé d'associer à chaque classe un modèle ILoNDF qui permet d'acquérir leur caractéristiques synthétiques en termes d'habituation et de nouveauté. À partir des deux modèles construits, deux valeurs peuvent être obtenues pour chaque nouvelle donnée en utilisant des stratégies inédites pour l'exploitation du contenu du modèle ILoNDF. Ces deux valeurs sont ensuite combinées sous forme de somme pondérée pour obtenir le score de classification de la donnée en question. La prise en compte du problème de la dérive de concept a nécessité un raffinement des fonctionnalités d'adaptation du modèle ILoNDF. En effet, l'analyse expérimentale du comportement du modèle ILoNDF nous a fait percevoir que le délai nécessaire à la récupération de la performance pourrait être très long, en particulier, dans le cas d'une dérive brusque de concept. Nous avons donc introduit un facteur de biais dans la règle d'apprentissage du modèle ILoNDF qui permet d'accélérer son habituation aux données les plus récentes en cas de détection d'une dérive de concept.

La validation du fonctionnement du modèle ILoNDF en mode supervisé a été particulièrement réalisée dans le cadre du filtrage d'informations. Dans ce cadre, notre objectif primaire était la conception d'un système de filtrage basé sur le contenu, orienté utilisateur. À ce titre, notre contribution a porté principalement sur la modélisation du profil utilisateur. Cette modélisation est faite par l'intermédiaire de deux modèles de type ILoNDF appris respectivement à partir d'exemples positifs et négatifs du besoin d'informations de l'utilisateur. Le potentiel d'utilisation du modèle ILoNDF comme profil utilisateur s'articule autour des points suivants :

- La possibilité de la modélisation du profil utilisateur à partir d'exemples positifs uniquement ;
- L'avantage d'un apprentissage incrémental du profil utilisateur qui constitue une nécessité absolue dans le cas du filtrage adaptatif ;
- La spécification du type du besoin de l'utilisateur en termes de critères purement objectifs. Ceci nous a permis de définir une méthode de seuillage basée sur la précision attendue de l'utilisateur qui dépend directement de son comportement et des caractéristiques intrinsèques de son besoin d'informations ;
- La détection et le suivi de l'évolution des centres d'intérêt de l'utilisateur au cours du temps.

Les résultats des expérimentations que nous avons obtenus sont très encourageants. Ils ont montré, d'une manière un peu analogue au cas de la classification à partir d'une seule classe, que

le modèle ILoNDF peut montrer une performance comparable à celle des meilleures méthodes, comme SVM, tout en offrant nettement plus d'avantages. Ils ont également permis de mettre en évidence le potentiel de l'intégration de l'utilisateur comme une composante très importante dans la stratégie d'évaluation d'un système de filtrage d'informations. Nous avons enfin montré expérimentalement le potentiel de l'application du mode d'apprentissage biaisé du modèle ILoNDF en cas d'une dérive brusque ou progressive du besoin de l'utilisateur.

Une grande partie du travail précité a fait l'objet d'une implémentation dans le système de distribution ciblée de sites web multilingues par satellite CASABLANCA. Nous avons de plus intégré de nouvelles fonctionnalités adaptées aux exigences spécifiques du système. Rappelons, principalement, l'indexation conceptuelle indépendante de la langue des sites web, la gestion de l'intégration de nouveautés, et la combinaison de filtrage par contenu et de filtrage collaboratif.

Enfin, nous avons procédé au cas où les données arrivant en flux peuvent être réparties en classes multiples non-étiquetées. Dans ce cas-ci, nous avons proposé une stratégie consistant à découper le flux de données en fenêtres temporelles sur lesquelles on applique une approche statique de classification non supervisée. Ensuite, une analyse de l'évolution temporelle en termes d'apparition et de disparition des informations contenues dans le flux est faite en comparant les classes construites à partir des données issues de chaque fenêtre de temps. La méthode de classification non supervisée constitue le fondement de notre stratégie. La méthode que nous avons développée est basée sur un modèle neuronal à des niveaux d'abstraction multiples, nommée A-CLN. Le niveau de base de ce modèle est formé par un réseau neuronal à apprentissage compétitif dont les connexions entre les neurones sont insérées par un apprentissage hebbien compétitif. Le nombre de neurones dans ce niveau doit être suffisamment élevé de façon à pouvoir précisément capturer les caractéristiques exactes des données et représenter leur structure latente. Pour déterminer le nombre de neurones, nous avons donc mis en place de nouveaux indices de validité qui permettent de privilégier l'homogénéité des classes et qui sont plus adaptés au cas des données fortement multidimensionnelles. En plus de ce niveau de base, le modèle A-CLN comporte fondamentalement deux niveaux d'abstraction de type macro et micro offrant deux différents degrés de granularité. Ces niveaux sont construits à partir du niveau de base en exploitant les connexions entre ses neurones pour les regrouper dans des classes d'ordre plus élevé. Ce modèle, qui est statique par construction, permet de prendre en compte l'aspect temporel en comparant les clusters construits à partir de deux fenêtres temporelles tout en considérant leur appartenance aux différentes niveaux du modèle.

En résumé, nous pensons que ce travail est le premier à proposer un modèle d'apprentissage qui répond, par construction, aux exigences relatives à l'analyse des flux de données multidimensionnelles. De plus, ce modèle a été particulièrement étudié pour la tâche de classification à partir d'une seule classe, ce qui est important et difficile notamment dans le cadre des flux de données. La capacité du modèle ILoNDF à maintenir une performance supérieure ou au moins égale aux méthodes statiques, à partir d'un apprentissage en ligne et en une seule passe, nous fait penser qu'il pourrait faire l'objet d'une attention particulière tant dans ce domaine que dans d'autres. La capacité d'oubli du modèle ILoNDF qui lui permet la prise en compte de la dimension temporelle des flux de données de manière implicte (apprentissage classique du modèle) ou explicite (apprentissage biaisé du modèle), nous fait aussi penser qu'il présente un intérêt futur important en matière d'analyse de données évolutives. L'application du modèle ILoNDF comme modèle utilisateur dans les systèmes de filtrage d'informations, offre une nouvelle manière de regarder le profil utilisateur en termes de critères purement objectifs caractérisant le type de besoin de

l'utilisateur. Ceci nous a permis, dans une première étape vers la conception d'un système de filtrage orienté utilisateur, de définir une méthode de seuillage basée sur la précision attendue de l'utilisateur qui dépend directement du type de son besoin, et d'intégrer donc l'utilisteur comme composante importante dans la stratégie d'évaluation d'un système de filtrage d'informations. Cette approche est tout à fait originale et pertinente dans le domaine, et, point intéressant ici, seul le modèle ILoNDF permet de la réaliser. Notons, cependant, que notre objectif n'est pas de remplacer les méthodes de filtrage existantes mais d'étendre leur richesse et de faire apparaître des fonctionnalités complémentaires qui permettent une statisfaction plus complète de l'utilisateur.

Pour la poursuite de notre travail, nos perspectives à court terme consistent à intégrer des techniques de visualisation dans notre modèle d'analyse temporelle des flux de données non-étiquetées. Ces techniques doivent permettent, d'une part, de visualiser les clusters dans les différents niveaux du modèle A-CLN, ainsi que les différentes relations qui les lient, d'autre part, de mettre en évidence les relations entre deux modèles A-CLN associés à deux fenêtres temporelles. Bien que le domaine des techniques de visualisation soit très riche, il n'existe pas de techniques qui s'appliquent directement à notre modèle. En effet, nous nous intéressons à une approche combinant des techniques de visualisation de graphes avec des techniques de visualisation hiérarchique ou hyperbolique, mais aussi, avec des techniques de visualisation de l'évolution temporelle des clusters. Dans ce même cadre, nous comptons remplacer la méthode NG, qui a été utilisée jusque maintenant juste à titre illustratif, par une autre méthode plus adaptée pour la prise en compte de l'aspect temporel. Plusieurs méthodes qui ont été présentées dans la partie de l'état de l'art peuvent faire l'objet de ce remplacement (cf. Sections 2.2.2 et 2.2.3). Nous pensons plus particulièrement à des adaptations des méthodes de Furao et Hasegawa (2006) et de Prudent et Ennaji (2005).

À moyen terme, nous devons envisager des méthodes de clustering en mode incrémental et adaptées au traitement des flux de données non stationnaires. C'est dans ce cas-ci que nous pouvons mettre en valeur les capacité de fonctionnement en ligne des modèles ILoNDF qui seront associés aux micro-clusters formés par ces méthodes. Ce problème est cependant assez complexe et peu de solutions ont été proposées jusqu'à aujourd'hui. À notre avis, l'adaptation conjointe des solutions que nous avons proposées au sujet de l'analyse du contenu du modèle ILoNDF et nos indices de validité de la qualité des clusters doivent nous permettre de développer une méthode dynamique de clustering où la création/élimination/fusion, des clusters sera en termes de critère de spécificité et exhaustivité de leur contenu. Dans ce cadre, nous devons également considérer l'aspect temporel lors du prétraitement des données et la mise à jour incrémentale de l'espace de représentation des données issues des flux.

Au terme de ce travail, nous comptons sortir du cadre des données documentaires pour nous intéresser à d'autres types de données de manière à valider la généralité du modèle ILoNDF. Nous pensons en particulier à l'analyse des données fournies par les puces à ADN. Ce dernier domaine représente un terrain d'expérimentation très intéressant pour le traitement des changements temporels. En effet, un certain nombre d'expériences basées sur les puces ADN ont pour but des changements métaboliques intervenants dans les cellules en situation pathologique ou anormale et peu de solutions existent à ce jour pour traiter les données issues de telles expériences. Il serait également intérestant d'étudier le comportement du modèle ILoNDF dans la résolution de problèmes non linéaires complexes et de le comparer aux méthodes de noyaux non linéaires.

À long terme et de manière plus générale, nous pensons qu'il serait interessant de proposer des architectures distribuées des modèles orientés flux de données pour des raisons de répartition de charge et de tolérance aux pannes. Nous pensons aussi qu'un véritable travail reste à faire en ce qui concerne l'évaluation des méthodes évolutives et dynamiques. Notons, à ce sujet, qu'une telle évaluation n'est possible à présent qu'au sein d'une vraie application et seulement de manière purement subjective.

Annexe A

Présentation des corpus de test

L'évaluation des résultats des méthodes d'apprentissage tant supervisé que non supervisé impose le recours à des collections de test adéquates et correctement étiquetées. Pour rendre comparables nos résultats à ceux d'autres travaux, il nous a donc paru évident l'usage de corpus standard, souvent disponibles gratuitement pour des fins de recherche. Dans la plupart des cas, ces corpus sont très homogènes. Cette homogénéité porte en particulier sur le format, sur la langue d'écriture et sur le contenu thématique. Un corpus correspond alors à un ensemble de documents qui sont classés dans une ou plusieurs catégories.

Cette annexe présente les collections de test auxquelles nous faisons référence dans ce travail de thèse. Principalement, trois collections ont été utilisées dans nos expérimentations : Reuters-21578, WebKB, et Google. Le choix de ces corpus a été fait de façon à couvrir le plus de caractéristiques possible des documents textuels soumis à des fins d'apprentissage : nombre de catégories par document, nombre de documents par catégorie, distribution des documents entre catégories, etc.

A.1 Reuters-21578

La distribution 1.0 du corpus Reuters-21578[43] est très largement utilisée en classification et en filtrage de documents textuels (Schapire et al., 1998; Debole et Sebastiani, , 2005; Dumais et al., 1998). Ce corpus est constitué de 21578 documents extraits à partir de dépêches de l'agence de presse Reuters[44]. Les documents ont été classés manuellement dans l'une ou plusieurs des 135 catégories sémantiques. Tous les documents sont structurés à l'aide de balises de présentation indiquant par exemple le titre, la date et le contenu de chaque document. La figure A.1 montre un exemple de document issu de ce corpus.

Plusieurs découpages du corpus Reuters-21578 ont été proposés pour le diviser en deux sous-ensembles : les données d'apprentissage et celles de test (ModHayes, ModLewis, ModApte et ModTrivial)[45]. La version ModApte est la plus commune (Apté et al., 1994), nous l'utilisons donc dans nos travaux pour extraire deux sous-ensembles : un ensemble d'apprentissage constitué de 9603 documents et un ensemble de test constitué de 3299 documents. Les catégories

[43] http ://www.research.att.com/ lewis/reuters21578.html
[44] http ://www.reuters.com/
[45] Voir le fichier README pour plus de détails sur les différents découpage du corpus Reuters-21578.

```
<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="2041" NEWID="13072">
<DATE> 3-APR-1987 14:21:09.53 </DATE>
<TOPICS><D>acq</D></TOPICS>
<PLACES><D>usa</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<TEXT>
<TITLE>SEC CLARIFIES POSITION ON TENDER OFFER CHANGES</TITLE>
<DATELINE>  WASHINGTON, April 3 - </DATELINE>
<BODY>The Securities and Exchange Commission reminded corporate raiders and others tendering for
the shares of companies that they must extend the period their offers are open if key conditions
are changed. Specifically, the agency said those making tender offers for companies stock must
extend the offers if they decide to eliminate conditions requiring a minimum number of shares to be
tendered in order for the offers to be valid. Tender offers typically include minimum share
conditions. As a result, a purchaser would not be bound to buy the shares that were tendered if the
minimum level were not reached. ..... Officials declined to identify the two offers. "If a bidder
makes a material change near or at the end of its offer, it will have to extend the offer to permit
adequate dissemination," the SEC said. Federal securities law requires that all tender offers
remain open for at least 20 business days.
</BODY>
</TEXT>
</REUTERS>
```

FIG. A.1 – Un exemple de document issu du corpus Reuters-21578 (version du texte raccourcie). Le document est issu de l'ensemble d'apprentissage de la catégorie "acq".

retenues sont celles pour lesquelles il existe au moins un document sur l'ensemble d'apprentissage et un document pertinent sur l'ensemble de test, ce qui permet de retenir 90 catégories. Ce corpus possède un ensemble de caractéristiques qui le rendent fort intéressant à des fins d'expérimentation :

- Il s'agit d'un corpus multi-labels où les documents peuvent être classés dans une ou plusieures catégories ; Le nombre moyen de catégories par document est de 1.3 avec un maximum de 14 catégories pour quelques documents ;
- La distribution des documents entre les différentes catégories est fortement déséquilibrée. Certaines catégories ont très peu de documents (exemples positifs) tandis que d'autres en ont des milliers (cf. Figures A.2). En fait, les 10 catégories les plus fréquentes contiennent 75% de documents contre 25% pour le reste de catégories. ;
- Il existe des relations sémantiques, parfois très fortes, entre les catégories. Par exemple, les catégories "corn" et "wheat" sont souvent des sous-catégories de "grain". De même, plus de 40% des documents d'apprentissage de la catégorie "interest" appartiennent également à la catégorie "money-fx". Cependant, ces relations ne sont pas explicitées par aucune organisation hiérarchique ou autre.

Pour certaines de nos expérimentations, nous nous limitons aux 10 catégories les plus fréquentes. Pour cela, nous désignons le corpus Reuters-21578 dans ce cas par le nom *Reuters10* et, parfois, le corpus de base par le nom *Reuters90*. Le nombre de documents d'apprentissage et de test assignés à chacune de ces catégories est donné dans le tableau A.1.

La transformation des documents du corpus pour les mettre sous la forme vectorielle est réalisée, sauf si indiqué différemment, selon des étapes standard de prétraitement : tokenisation, suppression des mots vides, et lemmatisation. Comme termes d'indexation, nous ne retenons que les mots simples les plus pertinentes apparaissant dans l'ensemble des documents. Ces termes sont choisis localement selon la statistique du chi^2 (cf. Section 3.1.3). Les 50 termes les plus

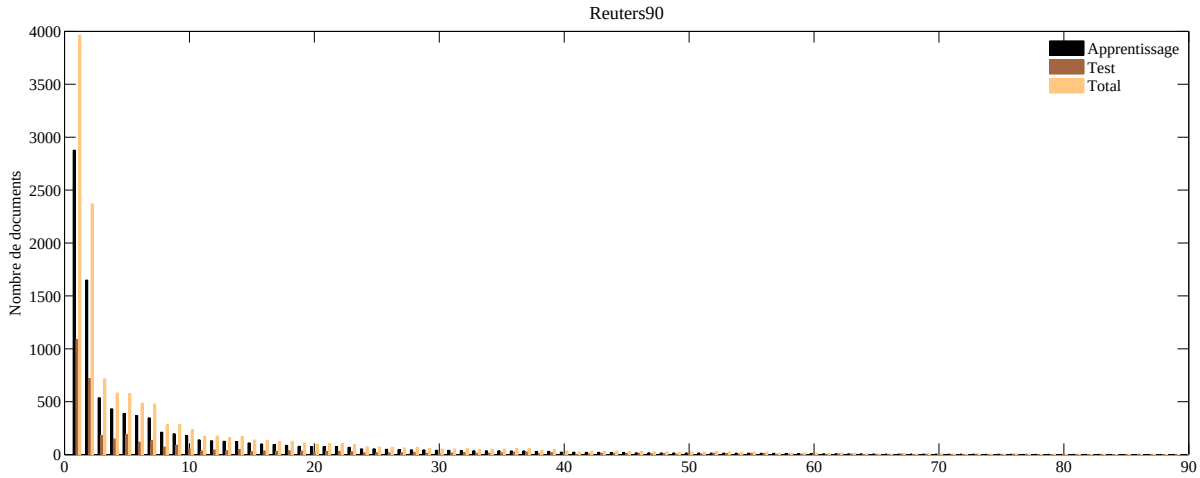


FIG. A.2 – Répartition des documents dans les catégories du corpus Reuters-21578

pertinents sont sélectionnés pour chacune des catégories du corpus et sont ensuite réunis pour former un espace global de représentation des documents[46]. L'espace de représentation obtenu est constitué de 379 termes dans le cas du corpus *Reuters10* et de 3290 termes dans le cas du corpus *Reuters90*. La pondération des termes est basée sur un schéma de type TF-IDF dans le cas des documents d'apprentissage et de type TF dans le cas des documents de test ($TF = 1+log_2(tf_{td})$). Une normalisation de type cosinus est ensuite appliquée à la fois aux vecteurs des documents d'apprentissage et de test (voir Section 3.1.2 pour un rappel sur les méthodes de pondération et de normalisation).

Partition temporelle du corpus Reuters-21578

Nous avons partitionné le corpus Reuters-21578 en périodes mensuelles couvrant les cinq mois représentant les dates des documents du corpus. Les différents sous-corpus obtenus (*R-FEB, R-MAR, R-APR, R-JUN*, et *R-OCT*) sont soumis à des étapes standard de prétraitement permettant l'extraction d'une liste des termes candidats. La sélection des termes est réalisée selon la méthode *"mean TF-IDF"*, cf. Section 3.1.3. Le tableau A.2 résume les différentes caractéristiques des cinq sous-corpus obtenus : *R-FEB, R-MAR, R-APR, R-JUN*, et *R-OCT*. Le nombre de catégories communes entre les différentes sous-corpus est donné dans le tableau A.3.

A.2 WebKB

Le corpus WebKB a été constitué par une équipe de l'université Carnegie Mellon (CMU) dans le cadre du projet "World Wide Knowledge Base". Il est composé de 8280 documents représentant des pages web recueillies de sites web de départements d'informatique de plusieurs universités américaines, classés en sept catégories : "student", "faculty", "staff", "department", "course", "project", et "other". Comme beaucoup d'autres travaux (Yang et al., 2008; Hoi et al., 2006; Mihalcea et Hassan, 2005), nous avons fait le choix d'exclure la catégorie "other" du corpus WebKB à

[46] Notre choix de 50 termes uniquement par catégorie n'est fondé que sur le fait que la longueur moyenne des documents d'apprentissage après achèvement des étapes de prétraitement est de 47.

TAB. A.1 – Nombre de documents dans les 10 catégories les plus fréquentes du corpus Reuters-21578

Nom de catégories	# doc. apprentissage	# doc. test
earn	2877	1087
acq	1650	719
money-fx	538	179
grain	433	149
crude	389	189
trade	369	117
interest	347	131
wheat	212	71
ship	197	89
corn	181	56

TAB. A.2 – La partition temporelle du corpus Reuters-21578 en cinq sous-corpus correspondant à cinq différentes périodes de temps.

Nom du corpus	#Doc.	#Termes	#Catégories	#Doc./catégorie
R-FEB	163	193	38	9,21
R-MAR	8087	1838	86	169,15
R-APR	2749	1004	80	61,79
R-JUN	1068	676	77	29,29
R-OCT	800	454	57	33,90

TAB. A.3 – Nombre de catégories communes entre les différentes partitions temporelles du corpus Reuters-21578.

	R-FEB	R-MAR	R-APR	R-JUN	R-OCT
R-FEB	38	36	36	37	30
R-MAR		86	77	75	57
R-APR			80	70	52
R-JUN				77	52
R-OCT					57

cause de sa définition très générale[47]. Il reste alors 4517 documents. Le nombre de documents de formation par catégorie varie de 116 à 1090 avec une moyenne de 711.7 documents par catégorie (cf. Tableau A.4). Contrairement au corpus Reuters-21578, chaque document est associé à une seule catégorie. Néanmoins, le corpus WebKB est relativement difficile du fait que les termes extraits de ce corpus sont très dispersés sur les catégories, ce qui signifie qu'il y a beaucoup moins de termes discriminatifs que de termes non discriminatifs dans le cas de ce corpus.

Tab. A.4 – Répartition des documents d'apprentissage et de test dans les six catégories tirées du corpus WebKB.

Nom de catégories	# doc. apprentissage	# doc. test
student	1513	128
faculty	1089	34
course	886	44
project	484	20
depart	181	1
staff	116	21
Total	4269	248

A.3 Google

Dans le cadre de projet ESA Sat-N-Surf, nous avons établi notre propre corpus de test vu qu'il n'existait pas, au début de ce projet, un corpus qui convenait aux contraintes inhérentes au système CASABLANCA :

- Les pages web sont multilingues (anglais, français, allemand, italien et polonais) ;
- Le nombre de pages web est limité par la taille de bouquet de transmission et il ne dépasse donc pas quelques centaines de pages web ;
- Le nombre de votes est restreint du fait que le système CASABLANCA ne considère que les votes des utilisateurs soumis pendant les 15 derniers jours (ceci étant pour s'assurer d'un vote régulier des utilisateurs).

Pour la constitution du corpus, nous nous sommes servi de pages web référencées dans les catégories de l'annuaire de Google[48]. Le corpus construit comporte 11 catégories et 1004 pages web en 5 différentes langues supportées par le système CASABLANCA. Le tableau A.5 indique le nombre de documents d'apprentissage et de test dans les catégories du corpus Google.

[47] En effet, la catégorie "other" regroupe toutes les pages web qui n'ont pas été classées dans une des autres six catégories. Elle peut donc être vue comme une catégorie "poubelle".

[48] http ://www.google.fr/dirhp ?hl=fr

TAB. A.5 – Répartition des documents d'apprentissage et de test dans le corpus établi à partir de l'annuaire de Google.

Nom de catégories	Monolingue (anglais)		Multilingue (5 langues)	
	# doc. apprentissage	# doc. test	# doc. apprentissage	# doc. test
Computers	36	14	67	61
Shopping	32	11	64	51
Science	20	8	51	31
Society	27	12	44	41
Business	28	47	45	67
News	15	20	50	66
Arts	16	18	24	46
Recreation	18	12	40	28
Sports	12	18	22	34
Health	23	12	46	25
Games	32	11	65	36
Total	259	183	518	486

Annexe B

Publications

Ce travail de thèse a donné lieu à un article dans une revue internationale, six publications dans des conférences internationales dont un poster, une publication dans une conférence nationale, et trois rapports techniques.

Revue internationale

- Kassab, R., et Alexandre, F. (2009). Incremental Data-driven Learning of a Novelty Detection Model for One-Class Classification Problem with Application to High-Dimensional Noisy Data. Machine Learning. 74(2) :191–234.

Conférences internationales avec comité de lecture et actes

- Kassab, R., Lamirel, J.-C., et Nauer, E. (2005). Novelty Detection for Modeling User's Profile. The 18th International FLAIRS Conference. 830–831.
- Kassab, R., et Lamirel, J.-C. (2006). A new approach for intelligent text filtering based on novelty detection. Australasian Database Conference (ADC). 149–156.
- Kassab, R., et Lamirel, J.-C. (2006). An innovative approach to intelligent information filtering. The 21st Annual ACM Symposium on Applied Computing - Information Access and Retrieval (SAC-IAR). 1089–1093.
- Kassab, R., et Lamirel, J.-C. (2007). Towards a synthetic Analysis of the User's need for Effective Information Filtering. The 22st Annual ACM Symposium on Applied Computing-Information Access and Retrieval (SAC-IAR). 852–859.
- Kassab, R., et Lamirel, J.-C. (2008). Feature-Based Cluster Validation for High-Dimensional Data. IASTED International Conference on Artificial Intelligence and Applications. 232–239.
- Kassab, R., et Lamirel, J.-C. (2008). A Multi-level Abstraction Model for Competitive Learning Neural Networks. IASTED International Conference on Artificial Intelligence and Applications. 97–103.

Conférences nationales avec comité de lecture et actes

- Kassab, R., Lamirel, J.-C., et Nauer, E. (2005). Une nouvelle approche pour la modélisation du profil de l'utilisateur dans les systémes de filtrage d'information : le modèle de filtre détecteur de nouveauté. The Deuxième Conférence en Recherche d'information et Applications, CORIA'05. 185–200.

Rapports techniques

- Boyer, A., Castagnos, S., Kassab, R., et Lamirel, J.-C. (2005). State-of-the-art report on filtering and profiling techniques. Technical Report for ESA Sat-N-Surf Project : A.1-D1 Bibliography.
- Boyer, A., Castagnos, S., Kassab, R., et Lamirel, J.-C. (2005). Selection of profiling, filtering and content analysis techniques. Technical Report for ESA Sat-N-Surf Project : A.1-D3 Analysis.
- Boyer, A., Castagnos, S., Kassab, R., et Lamirel, J.-C. (2005). Proposal of valuation methods for the filtering algorithms. Technical Report for ESA Sat-N-Surf Project : A.1-D2 Evaluation.

Bibliographie

Aeyels, D. (1990). On the Dynamic Behaviour of the Novelty Detector and the Novelty Filter. *Bonnard, B., Bride, B., Gauthier, J., et Kupka, I., (Eds.), Analysis of Controlled Dynamical Systems*. 1–10.

Aggarwal C. C. (2006). On biased reservoir sampling in the presence of stream evolution. *Proceedings of the 32nd International Conference on Very Large Data Bases*. 607–618.

Aggarwal, C. (2007). *Data Streams : Models and Algorithms*. Springer.

Aggarwal, C. C., Han, J., Wang, J., et Yu, P.S. (2003). A framework for clustering evolving data streams. *Proceeings of the 29th VLDB conference, Berlin, Germany*. 81–92.

Aggarwal, C. C., Han, J., Wang, J., et Yu, P.S. (2004). A Framework for Projected Clustering of High Dimensional Data Streams. *Proc. 2004 Int. Conf. on Very Large Data Bases (VLDB'04), Toronto, Canada*. 852–863.

Alpert, J., et Hajaj, N. (2008). We knew the web was big .... *The Official Google Blog*.

Allan, J. (1996). Incremental relevance feedback for information filtering. *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 270-278.

Allan, J., Carbonell, J., Doddington, G., Yamron, J., et Yang, Y. (1998). Topic detection and tracking pilot study : Final report. *Proceedings DARPA Broadcast NewsTranscription and Understanding Workshop*. 194–218.

Alpha, S., Dixon, P., Liao, C., et Yang, C. (2001). Oracle at TREC 10 : Filtering and Question-Answering. *Text REtrieval Conference*.

Annika, W. (2004). User involvement in automatic filtering : an experimental study. *Information Processing and Management*. 14(2-3) :201–237.

Apté, C., Damerau, F., et Weiss, S. (1994). Automated Learning of Decision Rules for Text Categorization. *ACM Transactions on Information Systems*. 12(3) :233–251.

Arampatzis, A., Beney, J., Koster, C. H. A., et van der Weide, T. P. (2001). Incrementality, half-life, and threshold optimization for adaptive document filtering. *The Ninth Text REtrieval Conference (TREC-9)*. 589–600.

Aslam, J. A., Pelekhov, E., et Rus, D. (2004). The Star Clustering Algorithm for Static and Dynamic Information Organization. *Journal of Graph Algorithms and Applications*. 8(1) :95–129.

Ault, T., et Yang, Y. (2000). kNN at TREC-9. *Proc. of the 9th Text REtrieval Conference (TREC-9)*. 127–134.

Ault, T., et Yang, Y. (2001). KNN, Rocchio and Metrics for Information Filtering at TREC-10. *The Tenth Text REtrieval Conference (TREC 10)*. 84–93.

Babcock, B., Datar, M., et Motwani, R. (2002). Sampling from a Moving Window over Streaming Data. *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 633–634.

Balabanovic, M. (1996). An adaptive web page recommandation service. *Processing of the 1st International Conference on Autonomous Agents*. 378–385.

Baldi, P., Chauvin, Y., et Hornik, K. (1995). Back-Propagation and Unsupervised Learning in Linear Networks. *Chauvin, Y., et Rumelhart, D. E., éditeurs, Back Propagation : Theory, Architectures and Applications, Lawrence Earlbaum Associates*. 389–432.

Battista, G. D., Eades, P., Tamassia, R., et Tollis, I. (1999). *Graph Drawing Algorithms For the Visualization of Graphs*. Prentice-Hall.

Beeferman, D., Berger, A., et Lafferty, J. (2002). Statistical Models for Text Segmentation. *Machine Learning*. 34(1) :177–210.

Belkin, N., et Croft, B. (1992). Information Filtering and Information Retrieval : Two sides of the same coin ?. *Communication of the ACM*. 35(12) :29–38.

Bellman, R. (1961). *Adaptive Control Processes : A Guided Tour*. Princeton University Press, Princeton.

Ben-Israel, A., et Greville, T. N. E. (2003). *Generalized Inverse : Theory and Applications*. 2nd Edition, Springer Verlag, New York.

Billsus, D., et Pazzani, M. (2000). User Modeling for Adaptive News Access. *User-Modeling and User-Adapted Interaction*. 10(2-3) :147–180.

Bingham, E., Kab, A., et Girolami, M. (2003). Topic Identification in Dynamical Text by Complexity Pursuit. *Neural Processing Letters*. 17 :1–15.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer-Verlag.

Blei, D. M., et Moreno, P. J. (2001). Topic segmentation with an aspect hidden Markov model. *Proceedings of SIGIR*. 343-348.

Bordes, A., Ertekin, S., Weston, J., et Bottou, L. (2005). Fast Kernel Classifiers with Online and Active Learning. *Journal of Machine Learning Research*. 6 :1579–1619.

Boughanem, M., Chrisment, C., et Tmar, M. (2001). Mercure and mercurefiltre applied for web and filtering tasks at TREC-10. *Proceedings of the 10th Text REtrieval Conference (TREC-10)*.

Boughanem, M., Dkaki, T., Mothe, J., et Soulé-Dupuy, C. (1998). Mercure at TREC-7. *Proc. of TREC-7*. 135–141.

Boughanem, M., et Soulé-Dupuy, C. (1992). A Connexionist Model for Information Retrieval. *Database and Expert Systems Applications*. 260-265.

Boughanem, M., Tebri, H., et Tmar, M. (2002). IRIT at TREC 2002 : Filtering Track. *Proceedings of the Eleventh Text REtrieval Conference.*

Boughanem, M., et Tmar, M. (2002). Incremental adaptive filtering : profile learning and threshold calibration. *ACM Symposium on Applied Computing (SAC).* 640–644.

Bradley, P., Fayyad, U., et Reina, C. (1998). Scaling clustering algorithms to large databases. *Proceedings of the 4th int. conf. on Knowledge Discovery and Data Mining.* 9–15.

Breese, J ; S., Heckerman, D., et Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence.* 43–52.

Breiman, L. (1996). Bagging predictors. *Machine Learning.* 24(2) :123–140.

Brouard, C., et Nie, J.-Y. (2004). Relevance as resonance : a new theoretical perspective and a practical utilization in information filtering. *Inf. Process. Manage.* 40(1) :1-19.

Brown, M. W., et Xiang, J.-Z. (1998). Recognition memory : Neuronal substrates of the judge ment of prior occurrence. *Progress in Neurobiology.* 55 :149–189.

Buckley, C., et Salton, G. (1995). Optimization of relevance feedback weights. *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* 351–357.

Buckley, C., Salton, G., et Allan, J. (1994). The effect of adding relevance information in a relevance feedback environment. *Proceedings of the 7th annual international ACM-SIGIR conference on research and development in information retrieval, Springer-Verlag.* 292–300.

Burke, R. (2002). Hybrid Recommender Systems : Survey and Experiments. *User Modeling and User-Adapted Interaction.* 12(4) :331–370.

Calinski, T., et Harabasz, J. (1974). A Dendrite Method for Cluster Analysis. *Communications in Statistics.* 3(1) :1–27.

Callan, J. (1998). Learning while filtering documents. *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval.* 224–231.

Carbonell, J. G., Yang, Y., Robert, R. E., Brown, R. D., Geng, Y., et Lee, D. (1997). Translingual Information Retrieval : A Comparative Evaluation. *In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence.* 708–714.

Carpenter, G. A., et Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Comput. Vision Graph. Image Process..* 37(1) :54–115.

Carpenter, G. A., et Grossberg, S. (1987). ART-2 : Self-organization of stable category recognition codes for analog input patterns. *Applied Optics.* 26(23) :4919–4930.

Castagnos, S., et Boyer, A. (2006). FRAC+ : A Distributed Collaborative Filtering Model for Client/Server Architectures. *Web Information Systems and Technologies (WEBIST).* 435–440.

Cauwenberghs, G., et Poggio, T. (2001). Incremental and decremental support vector machine learning. *Advances in Neural Information Processing Systems 13.* 409–415.

Chai, K., Ng, H., et Chieu, H. (2002). Bayesian online classifiers for text classification and filtering. *Proceedings of the 25th annual international ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR)*. 97–104.

Chappell, G., et Taylor, J. (1993). The Temporal Kohonen Map. *Neural Networks*. 6 :441–445.

Chen, H., Zhang, Y., et Houston, A. (1998). Semantic indexing and searching using a Hopfield net. *Journal of Information Science*. 24(1) :3-18.

Chiu, T., Fang, D., Chen, J., Wang, Y., et Jeris, C. (2001). A robust and scalable clustering algorithm for mixed type attributes in large database environment. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD), San Francisco, CA, USA*. 263–268.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., et Sartin, M. (1999). Combining Content-Based and Collaborative filters in an Online Newspaper. *ACM SIGIR Workshop on Recommender Systems - Implementation and evaluation*.

Cohen, W. W., et Singer, Y. (1999). Context-Sensitive Learning Methods for Text Categorization. *ACM Transactions on Information Systems*. 17(2) :141–173.

Cottrell, G. W., et Munro, P. (1988). Principal components analysis of images via back propagation. *Proceedings of the Society of Photo-Optical Instrumentation Engineers*. 1070–1077.

Croft, W. B. (1983). Experiments with Representation in a Document Retrieval System. *Information Technology : Research and Development*. 1–21.

Cutting, D. R., Karger, D. R., et Pedersen, J. O. (1993). Constant interaction-time scatter/gather browsing of very large document collections. *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*. 126–134.

Davies, D. L., et Bouldin, W. (1979). A Cluster Separation Measure. *IEEE Trans. Pattern Anal. Machine Intell.*. 1(2) :224-227.

Davis, J., et Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd international conference on Machine learning, Pittsburgh, Pennsylvania*. 233–240.

Debole, F., et Sebastiani, F. (2005). An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*. 56(6) :584–596.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., et Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*. 41(6) :391–407.

Denis, F., Gilleron, R., Laurent, A., et Tommasi, M. (2003). Text Classification and Co-Training from Positive and Unlabeled Examples. *Proceedings of the ICML 2003 Workshop : The Continuum from Labeled to Unlabeled Data*. 80–87.

Denning, P. J. (1982). Electronic junk. *Communications of the ACM*. 3(25) :163–165.

Detroja, K. P., Gudi, R. D., et Patwardhan, S. C. (2007). Plant-wide detection and diagnosis using correspondence analysis. *Control Engineering Practice*. 15(12) :1468–1483.

Domingos, P., et Hulten, G. (2000). Mining High Speed Data Streams. *Proc. of the 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*. 71–80.

Domingos, P., et Hulten, G. (2001). A general method for scaling up machine learning algorithms and its application to clustering. *Proceedings of the Eighteenth International Conference on Machine Learning*. 106–113.

Dumais, S., Platt, J., Heckerman, D., et Sahami, M. (1998). Inductive Learning Algorithms and Representations for Text Categorization. *Proc. 7th International Conference on Information and Knowledge Management CIKM*. 148-155.

Emamian, V., Kaveh, M., et Tewfik, A. H. (2000). Robust clustering of acoustic emission signals using the Kohonen network. *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*. 3891–3894.

Ester, M., Kriegel, H.-P., Sander, J., et Xu, X. (1996). A Density Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proc. 2nd International Conference on Knowledge Discovery and Data Mining*. 226–231.

Fan, W. (2004). Systematic data selection to mine concept-drifting data streams. *Proceedings of 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 128–137.

Faour, A., Leray, P., et Eter, B. (2007). Growing hierarchical self-organizing map for alarm filtering in network intrusion detection systems. *International conference on New Technologes, Mobility and Security*.

Farnstrom, F. (2000). Scalability for clustering algorithms revisited. *SIGKDD Explorations*. 2(1) :51–57.

Ferret, O. (2002). Using collocations for topic segmentation and link detection. *Proceedings of the 19th International Conference on Computational Linguistics, COLING*. 260–266.

Fodor, I. K. (2002). A survey of dimension reduction techniques. *Technical Report, UCRL-ID-148494, Center for Applied Scientific Computing, Lawrence Livermore National Laboratory*.

Foltz P. (1990). Using Latent Semantic Indexing for Information Filtering. *Proceedings of the Conference on Office Information Systems*. 40–47.

Fritzke, B. (1995). A growing neural gas network learns topologies. *Advances in Neural Information Processing Systems 7*. 625–632.

Fritzke, B. (1997). A Self-Organizing Network that Can Follow Non-stationary Distributions. *Proceedings of the 7th International Conference on Artificial Neural Networks. Springer-Verlag*. 613–618.

Fritzke, B. (1997). Some competitive learning methods. *Technical report, NCRG/98/015. Institute for Neural Computation. Ruhr-Universität Bochum*.

Fung, G., et Mangasarian, O. L. (2002). Incremental support vector machine classification. *Proceedings of the Second SIAM Inter- national Conference on Data Mining*. 247–260.

Fung, G. P. C., Yu, J. X., Lu, H., et Lu, H. (2004). Classifying Text Streams in the Presence of Concept Drifts. *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. 373–383.

Fung, G. P. C., Yu, J. X., Lu, H., et Yu, P. S. (2006). Text Classification without Negative Examples Revisit. *IEEE Trans. Knowl. Data Eng.* 18(1) :6–20.

Furao, S., et Hasegawa, O. (2006). An incremental network for on-line unsupervised classification and topology learning. *Neural Networks. Elsevier Science Ltd.*. 19(1) :90–106.

Gama, J., Rocha, R., et Medas, P. (2003). Accurate decision trees for mining high-speed data streams. *KDD*. 523–528.

Gantz, J. F., Chute, C., Manfrediz, A., Minton, S., Reinsel, D., Schlichting, W., et Toncheva, A. (2008). The Diverse and Exploding Digital Universe : An Updated Forecast of Worldwide Information Growth Through 2011. *IDC White paper – sponsored by EMC*.

Gibbons, P. B., et Matias, Y. (1998). New sampling-based summary statistics for improving approximate query answers. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 331–342.

Gibbons, P. B., Matias, Y., et Poosala, V. (2002). Fast incremental maintenance of approximate histograms. *ACM Transactions on Database Systems (TODS)*. 27(3) :261–298.

Gilbert, A. C., Guha, S., Indyk, P., Kotidis, Y., Muthukrishnan, S., et Strauss, M. J. (2002). Fast, small-space algorithms for approximate histogram maintenance. *Proceedings of the 34th annual ACM symposium on Theory of computing*. 389–398.

Golab, L., et Özsu, M. T. (2003). Issues in data stream management. *SIGMOD Rec.*. 32(2) :5–14.

Golub, G. H., et Loan, C. F. V. (1991). *Matrix Computation*. John Hopkins University Press, Second Edition.

Muthukrishnan Greville, T. N. E. (1960). Some applications of the pseudoinverse of a matrix. *SIAM Rev.*. 2 :15–22.

Grossberg, S. (1976). Adaptative Pattern Classification and Universal Recording, I : Parallel Development and Coding of Neural Feature Detectors. *Biological Cybernetics*. 23 :121–134.

Guha, S., Koudas, N., et Shim, K. (2006). Approximation and streaming algorithms for histogram construction problems. *ACM Transactions on Database Systems (TODS)*. 31(1) :396–438.

Guha, S., Rastogi, R., et Shim, K. (1998). CURE : an efficient clustering algorithm for large databases. *Proceedings of ACM SIGMOD International Conference on Management of Data, Seattle, Washington, United States*. 73–84.

Guha, S., Rastogi, R., et Shim, K. (2000). ROCK : A robust clustering algorithm for categorical attributes. *Information Systems*. 25(5) :345–366.

Günter, S., et Bunke, H. (2002). Self-organizing map for clustering in the graph domain. *Pattern Recogn. Lett.*. 23(4) :405–417.

Guimarães, G., Lobo, V. S., et Moura-Pires, F. (2003). A taxonomy of Self-organizing Maps for temporal sequence processing. *Intelligent Data Analysis*. 7(4) :269–290.

Gupta, C., et Grossman, R. L. (2004). GenIc : A Single Pass Generalized Incremental Algorithm for Clustering. *SIAM International Conference on Data Mining*.

Habibi, S., et Eberts, R. (1992). Using neural networks to route messages. *Neural networks and pattern recognition in human-computer interaction. Beale, R., et Finlay, J. (Ed.), Ellis Horwood Pub.*. 229–241.

Halkidi, M., Batistakis, Y., et Vazirgiannis, M. (2001). On Cluster Validation Techniques. *Journal of Intelligent Information Systems*. 17(2) :107–145.

Hamers, L., et Hemeryck, Y. (1989). Similarity measures in scientometric research : the Jaccard index vs. Salton's cosine formula. *Info. Process. and Manage.*. 25 :315–318.

Han, J., Kamber, M., et Tung, A. K. H. (2001). Spatial clustering methods in data mining : A survey. *Miller, H. and Han, J. (Eds.) Geographic Data Mining and Knowledge Discovery, Taylor and Francis.*. 188–217.

Harman, D. (1992). The DARPA TIPSTER project. *SIGIR Forum*. 26(2) :26–28.

Harman, D. (1992). Overview of the first text retrieval conference (TREC-1). *Proceedings of the 1st Text REtrieval Conference (TREC-1). NIST, Gaithersburg, Maryland*. 1-20.

Harris, T. (1993). A Kohonen SOM based machine health monitoring system which enables diagnosis of faults not seen in the training set. *Proc. IJCNN-93-Nagoya, Int. Joint Conf. on Neural Networks, volume I*. 947–950.

Hartigan, J. A., et Wong, M. A. (1979). Algorithm AS136 : A k-means Clustering Algorithm. *Applied Statistics*. 28(1), 100–108.

Hearst, M. A. (1997). TextTiling : Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics*. 23(1) :33-64.

Hebb, D. O. (1949). *The Organization of Behaviour*. Wiley, New York.

Henle, M. (2001). *Modern Geometries*. Prentice Hall.

Hoashi, K., Matsumoto, K., Inoue, N., et Hashimoto, K. (1999). Experiments on the trec-8 filtering track. *Proceedings of the 8th Text REtrieval Conference (TREC-8)*. 457–463.

Hoi, S., Jin, R., et Lyu, M. (2006). Large-scale text categorization by batch mode active learning. *Proceedings of the International World Wide Web Conference*. 633–642.

Hopfield, J.J. (1982). Neural network and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA*. 79 :2554–2558.

Hore, P., Hall, L. O., et Goldgof, D. B. (2007). Single Pass Fuzzy C Means. *IEEE International Fuzzy Systems Conference*. 1–7.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*. 24 :417–441, 498–520.

Housman, E. M. (1969). Survey of current systems for selective dissemination of information. *Technical Report, American Society for Information Science Special Interest Group*.

Hsu, C.-W., et Lin, C.-J. (2002). A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*. 13(2) :415–425.

Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*. 2(3) :283–304.

Hull, D. A. (1998). The TREC-7 Filtering Track : Description and Analysis. *Proceedings of the 7th Text Retrieval Conference (TREC-7)*. 33–56.

Hull, D. A., et Robertson, S. (2000). The TREC-8 Filtering Track final report. *The 8th Text Retrieval Conference (TREC-8)*. 35–56.

Hulten, G., Spencer, L., et Domingos, P. (2001). Mining time-changing data streams. *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 97–106.

Hyvarinen, A., Karhunen, J., et Oja, E. (2001). *Independent Component Analysis*. Wiley.

Indyk, P., Koudas, N., et Muthukrishnan, S. (2000). Identifying Representative Trends in Massive Time Series Data Sets Using Sketches. *Proceedings of the 26th International Conference on Very Large Data Bases*. 363–372.

Jackson, J. E., et Mudholkar, G. S. (1979). Control Procedures for Residuals Associated with Principal Component Analysis. *Technometrics*. 21(3) :341–349.

Jagadish, H. V., Koudas, N., Muthukrishnan, S., Poosala, V., Sevcik, K. C., et Suel, T. (1998). Optimal Histograms with Quality Guarantees. *Proceedings of the 24rd International Conference on Very Large Data Bases*. 275–286.

Jain, A. K., Murty, M. N., et Flynn, P. J. (1999). Data Clustering : A Review. *ACM Computing Surveys*. 31(3) :264–323.

Japkowicz, N. (1999). Are we Better off without Counter Examples ?. *Proceedings of the 1st International ICSC Congress on Computational Intelligence Methods and Applications (CIMA-99)*. 242–248.

Japkowicz, N. (2001). Supervised Versus Unsupervised Binary-Learning by Feedforward Neural Networks. *Machine Learning*. 42(1-2) :97–122.

Japkowicz, N., Hanson, S. J., et Gluck, M. A. (2000). Nonlinear Autoassociation Is Not Equivalent to PCA. *Neural Comput.*. 12(3) :531–545.

Japkowicz, N., Myers, C., et Gluck, M. A. (1995) . A Novelty Detection Approach to Classification. *Proceedings of the Fourteenth Joint Conference on Artificial Intelligence*. 518–523.

Jebara, T. (2003). *Machine Learning : Discriminative and Generative*. Kluwer Academic.

Jennings, A. et Higuchi, H. (1993). A User Model Neural Network for a Personal News Service. *User Modeling and User-Adapted Information*. 3(1) :1–25.

Jin, R., et Agrawal, G. (2003). Efficient decision tree construction on streaming data. *KDD*. 571–576.

Joachims, T. (1998). Text categorization with support vector machines : learning with many relevant features. *Proceedings of 10th European Conference on Machine Learning, ECML*. 137–142.

Joachims, T. (1999). Making large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning, Schölkopf, B., Burges, C., et Smola, A. (ed.), MIT Press*. 169–184.

Jobbins, A. C., et Evett, L. J. (1998). Text Segmentation Using Reiteration and Collocation. *Proceedings of the 17th international conference on Computational linguistics*. 614–618.

Johnson, W.B., et Lindenstrauss, J. (1984). Extensions of Lipschitz mapping into Hilbert space. *Contemporary Mathematics*. 26 :189–206.

Jolliffe, I. T. (1986). *Principal Component Analysis*. Springer Verlag, New York.

Kan, M. Y., Klavans, J. L., et McKeown, K. R. (1998). Linear Segmentation and Segment Relevence. *Proceedings of 6th International Workshop of Very Large Corpora*. 197–205.

Kangas, J. (1991). Phoneme recognition using time-dependent versions of self-organizing maps. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*. 101–104.

Karras, P., et Mamoulis, N. (2005). One-pass wavelet synopses for maximum-error metrics. *Proceedings of the 31st Very Large Data Bases Conference (VLDB)*. 421-432.

Karypis, G., Han, E. H., et Kumar, V. 1999. CHAMELEON : A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer*. 32(8) :68–75.

Kaski, S., Honkela, T., Lagus, K., et Kohonen, T. (1998). WEBSOM–self-organizing maps of document collections. *Neurocomputing*. 21 :101–117.

Kassab, R., et Lamirel, J.-C. (2006). A New Approach to Intelligent Text Filtering Based on Novelty Detection. *the 17th Australian Database Conference, Tasmania, Australia*. 149–156.

Kassab, R., et Lamirel, J.-C. (2007). Towards a Synthetic Analysis of User's Information Need for More Effective Personalized Filtering Services. *the 22nd ACM Symposium on Applied Computing Special Track on Information Access and Retrieval (SAC-IAR)*. 852–859.

Kaufman, L., et Rousseeuw, P. J. (1990). *Finding Groups in Data : an introduction to cluster analysis*. Wiley, New York.

Kifer, D., Ben-David, S., et Gehrke, J. (2004). Detecting Change in Data Streams. *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB)*. 180–191.

Kim, J. H, et Beale, G. O. (2002). Fault Detection and Classification in Underwater Vehicles Using the T^2 Statistic. *AUTOMATIKA - Journal for Control, Measurement, Electronics, Computing and Communications*. 43(1-2) :29–37.

Kim, H.-J., Shrestha, J., Kim, H.-N., et Jo, G.-S. (2006). User Action Based Adaptive Learning with Weighted Bayesian Classification for Filtering Spam Mail . *Australian Conference on Artificial Intelligence*. 790-798.

Klinkenberg, R. (2004). Learning drifting concepts : Example selection vs. example weighting. *Intelligent Data Analysis*. 8(3) :281–300.

Klinkenberg, R., et Renz, I. (1998). Adaptive Information Filtering : Learning Drifting Concepts. *AAAI98/ICML-98 Workshop Learning for Text Categorization*. 33–40.

Klinkenberg, R., et Thorsten, J. (2000). Detecting Concept Drift with Support Vector Machines. *Proceedings of the 17th International Conference on Machine Learning*. 487–494.

Kohonen, T., et Oja, E. (1976). Fast adaptive formation of orthogonalizing filters and associative memory in recurrent networks of neuron-like elements. *Biol. Cybern.* 21 :85–95.

Kohonen, T. (1989). *Self Organisation and Associative Memory*. 3rd edition, Springer-Verlag New York, Inc., New York, NY, USA.

Kohonen, T. (2001). *Self-Organizing Maps*. 3ed., Springer.

Koychev, I. (2000). Gradual Forgetting for Adaptation to Concept Drift. *Proceedings of ECAI 2000 Workshop - Current Issues in Spatio-Temporal Reasoning*. 101–106.

Kozima, H. (1993). Text Segmentation Based on Similarity between Words. *The 31th Annual Meeting of the Association for Computational Linguistics*. 286–288.

Lam, W., et Yu, K. L. (2003). High-dimensional learning framework for adaptive document filtering. *Computational Intelligence*. 19(1) :42–63.

Lamping, J., et Rao, R. (1994). Laying out and visualizing large trees using a hyperbolic space. *Proceedings of the 7th annual ACM symposium on User interface software and technology*. 13–14.

Last, M. (2002). Online classification of nonstationary data streams. *Intelligent Data Analysis*. 6(2) :129–147.

Lebart, L., Morineau, A., et Fenelon, J. P. (1982). *Traitement des données statistiques*. Dunod, Paris.

Lee, H., et Cho, S. (2006). Application of LVQ to novelty detection using outlier training data. *Pattern Recognition Letters*. 27(13) :1572–1579.

Lelu, A. (2006). Clustering dynamique d'un flot de données : un algorithme incrémental et optimal de détection des maxima de densité. *6èmes journées d'Extraction et Gestion de Connaissances (EGC))*. 35–40.

Lemire, D. (2003). Scale and Translation Invariant Collaborative Filtering Systems. *Information Retrieval*. 8(1) :129–150.

Lewis, D. D. (1991). Evaluating Text Categorization. *Proceedings of Speech and Natural Language Workshop*. Morgan Kaufmann. 312–318.

Lewis, D. D. (1998). Naive Bayes at forty : The independence assumption in information retrieval. *Proc. Tenth European Conf. Machine Learning (ECML)*. 4–15.

Lewis, D. D., Schapire, R. E., Callan, J. P., et Papka, R. (1996). Training algorithms for linear text classifiers. *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*. 298–306.

Li, X., et Liu, B. (2003). Learning to classify texts using positive and unlabeled data. *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*. 587-594.

Lin X., Soergel D., et Marchionini, G. (1991). A self organizing semantic map for information retrieval. *Proceedings of the Fifteenth Annual lnternational ACM/SIGIR Conference on R& D in Information Retrieval*. 262–269.

Linde, Y., Buzo, A., et Gray, R. M. (1980). An Algorithm for Vector Quantizer Design. *IEEE Transaction on Communications*. 28(1) :84–95.

Littman, M. L., Dumais, S. T., et Landauer, T. K. (1997). Automatic cross language retrieval using latent semantic indexing. *Proceedings of the AAAI symposium on cross-language text and speech retrieval*. AAAI Technical Report SS-97-05.

Liu, B., Dai, Y., Li, X., Lee, W. S., et Yu, P. S. (2003). Building text classifiers using positive and unlabeled examples. *Intl. Conf. on Data Mining*. 179–186.

Liu, T., Liu, S., Chen, Z., et Ma, W. (2003). An Evaluation on Feature Selection for Text Clustering. *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*.

Liu, S., Lu, L., Liao, G., et Xuan, J. (2006). Pattern Discovery from Time Series Using Growing Hierarchical Self-Organizing Map . *International Conference on Neural Information Processing (ICONIP)*. 1030–1037.

Luhn, H. P. (1958). A Business Intelligence System. *IBM Journal of Research and Development*. 2(4) :314–319.

Malon, T., Grant, K., Turbak, F., Brobst, S., et Cohen, M. (1987). Intelligent Information-Sharing Systems. *Communication of the ACM*. 30(5) :390–402.

Manevitz, L. M., et Yousef, M. (2001). One-Class SVMs for Document Classification. *Journal of Machine Learning Research*. 2 :139–154.

Markou, M., et Singh, S. (2003). Novelty detection : a review—part 1 : statistical approaches. *Signal Process*. 83(12) :2481–2497.

Markou, M., et Singh, S. (2003). Novelty detection : a review—part 2 : neural network based approaches. *Signal Process*. 83(12) :2499–2521.

Marsland, S. (2002). On-line Novelty Detection Through Self-Organisation, With Application to Inspection Robotics. *PhD thesis, Department of Computer Science, The University of Manchester*.

Marsland, S., Nehmzow, U., et Shapiro, J. (2000). A Real-Time Novelty Detector for a Mobile Robot. *CoRR, cs.RO/0006006*.

Marsland, S., Nehmzow, U., et Shapiro, J. (2002). Environment-specific novelty detection. *From Animals to Animats, Proc. 7th International Conference on Simulation of Adaptive Behaviour*. 36–45.

Martinetz, T., et Schulten, K. (1991). A “neural gas” network learns topologies . *Artificial Neural Networks, Elsevier, Amsterdam*. 397–402.

Martinetz, T. 1993. Competitive hebbian learning rule forms perfectly topology preserving maps. *Int. Conf. on Artificial Neural Networks*. 427–434.

Martinetz, T., et Schulten, K. (1994). Topology representing networks. *Neural Networks*. 7 :507–522.

Matias, Y., Vitter, J. S., et Wang, M. (2000). Dynamic Maintenance of Wavelet-Based Histograms. *Proceedings of the 26th International Conference on Very Large Data Bases*. 101-110.

Melville, P., Mooney, R. J., et Nagarajan, R. (2002). Content-Boosted Collaborative Filtering for Improved Recommendations. *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI)*. 187–192.

Mercer, J. (1909). Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London - Series A*. 415–446.

Mihalcea, R., et Hassan, S. (2005). Using the essence of texts to improve document classification. *Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP)*.

Miller, B., Riedl, J., et Konstan, J. (1997). Experiences with grouplens : making usenet useful again. *Proceedings of the USENIX Winter Technical Conference*. 219–231.

Mitchell, T. M., Caruana, R., Freitag, D., McDermott, J., et Zabowski, D. (1994). Experience with a learning personal assistant. *Commun. ACM. New York, NY, USA*. 37(7) :1994.

Miyahara, K., et Pazsani, M. J. (2000). Collaborative Filtering with the Simple Bayesian Classifier. *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*. 679–689.

Morik, K., Brockhausen, P., et Joachims, T. (1999). Combining Statistical Learning with a Knowledge-Based Approach - A Case Study in Intensive Care Monitoring. *Proceedings of the Sixteenth International Conference on Machine Learning (ICML)*. 268-277.

Morris, J., et Hirst, G. (1991). Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text. *Computational Linguistics*. 17(1) :21–48.

Moschitti, A. (2003). A study on optimal parameter tuning for rocchio text classifier. *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR)*. 420–435.

Moya, M. R., Koch, M. W., et Hostetler, L. D. (1993). One-class classifier networks for target recognition applications. *Proc. World Congress on Neural Networks, International Neural Network Society (INNS)*. 797–801.

Muthukrishnan, S. (2005). Data Streams : Algorithms and Applications. *Foundations and Trends in Theoretical Computer Science*. 1(2) :117–236.

Ng, H. T., Goh, W. B., et Low, K. L. (1997). Feature selection, perceptron learning, and a usability case study for text categorization. *Belkin, N., Narasimhalu, A. D., et Willett, P., editeurs, Proceedings of the 20th Annual International Retrieval, Philadelphia, PA*. 67–73.

Nichols, D. (1997). Implicit ratings and filtering. *Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering*. 31–36.

Noda, M.-T., Makino, I., et Saito, T. (1997). Algebraic methods for computing a generalized inverse. *ACM SIGSAM Bulletin*. 31(3) :51–52.

Oard, D. W. (1997). Alternative Approaches for Cross-Language Text Retrieval. *In AAAI Symposium on Cross-Language Text and Speech Retrieval* . AAAI Technical Report SS-97-05.

Oard, D. W., et Dorr, B. J. (1996). A Survey of Multilingual Text Retrieval. *University of Maryland, Institute for Advanced Computer Studies. UMIACS-TR-96-19*.

Oard, D., et Marchionini, G. (1996). A conceptual framework for text filtering . *Technical Report CS-TR-3643, University of Maryland*.

O'Callaghan, L., Mishra, N., Meyerson, A., Guha, S., et Motwani, R. (2002). Streaming-Data Algorithms for High-Quality Clustering. *Proceedings of IEEE International Conference on Data Engineering (ICDE'02)*. 685–694.

Oja, E. (1982). A Simplified Neuron Model as a Principal Component Analyzer. *Journal of Mathematical Biology*. 15(3) :267–273.

Ontrup, J., et Ritter, H. (2005). A hierarchically growing hyperbolic self-organizing map for rapid structuring of large data sets. *Proceedings of the 5th Workshop on Self-Organizing Maps (WSOM)*.

Ontrup, J., et Ritter, H. (2006). Large-scale data exploration with the hierarchically growing hyperbolic SOM. *Neural Networks*. 19(6) :751–761.

Ordonez, C. (2003). Clustering binary data streams with k-means. *ACM DMKD* . 12–19.

Özgür, A., Özgür, L., et Güngör, T. (2005). Text Categorization with Class-Based and Corpus-Based Keyword Selection. *Lecture Notes in Computer Science, Vol.3733, (Proc. of the 20th International Symposium on Computer and Information Sciences)*. 607–616.

Penrose, R. (1955). A generalized inverse for matrices. *Proc. Cambridge Phil. Soc.*. 52 :406–413.

Piace C. (1991). A thesaural model of information retrieval. *In Information Processing and Management*. 27(5) :433–447.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*. 14(3) :130–137.

Prudent, Y. et Ennaji, A. (2005). A new learning algorithm for incremental self-organizing maps. *13th European Symposium on Artificial Neural Networks ESANN*. 7–12.

Ralambondrainy, H. (1995). A conceptual version of the K-means algorithm . *Pattern Recognition Letters*. 16(11) :1147–1157.

Rand, W. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*. 66 :846–850.

Raskutti, B., et Kowalczyk, A. (2004). Extreme re-balancing for SVMs : a case study. *SIGKDD Explor. Newsl*. 6(1) :60–69.

Rauber, A., Merkl, D., et Dittenbach, M. (2002). The Growing Hierarchical Self-Organizing Map : Exploratory Analysis of High-Dimensional Data. *IEEE Transactions on Neural Networks*. 13(6) :1331-1341.

Ritter, H. (1999). Self-organizing Maps in non-euclidean Spaces. *Oja, E., et Kaski, S., eds., Kohonen Maps, Elsevier*. 97-108.

Ritter, H. J., et Kohonen, T. (1989). Self organizing semantic maps. *Biological Cybernetics*. 61(4) :241–254.

Rizzo, R. (2001). LBG-m : a modified LBG architecture to extract high-order neural structures. *Proc. of International Joint Conference on Neural Networks (IJCNN)*. 779–783.

Robertson, S. E. (1977). The probability ranking principle in IR. *Journal of Documentation*. 33 :294–304.

Robertson, S., et Callan, J. (2002). Guidelines for the TREC 2002 filtering track. *Text REtrieval Conference (TREC) at http ://trec.nist.gov/data/filtering/T11filter_guide.html*.

Robertson, S., et Hull, D. A. (2001). The TREC-9 filtering track final report. *Proceedings of the Text Retrieval Conference*. 25–40.

Robertson, S., et Soboroff, I. (2002). The TREC 2002 Filtering Track Report. *The 11th Text REtrieval Conference* . 27–39.

Robertson, S. E., et Walker, S. (2000). Threshold setting in adaptive filtering. *Journal of Documentation*. 56(3) :312–331.

Rocchio, J. (1971). Relevance feedback in information retrieval. *Salton, G., éditeur : The SMART Retrieval System : Experiments in Automatic Document Processing, Prentice Hall, Englewood Cliffs, N. J.*. 313–323.

Rogati, M., et Yang, Y. (2002). High-performing feature selection for text classification. *Proceedings of the eleventh international conference on Information and knowledge management*. 659–661.

Rosenblatt, F. (1958). The Perceptron : A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*. 65(6) :386–408.

Rüping, S. (2001). Incremental Learning with Support Vector Machines. *Proceedings of the 2001 IEEE International Conference on Data Mining*. 641–642.

Rumelhart, D. E., Hinton, G. E., et Williams, R. J. (1986). Learning internal representation by error propagation. *Rumelhart, D. E., et McClelland, J. L., editeurs„ Parallel Distributed Processing : Explorations in the microstructures of cognition, MIT Press*. 318–362.

Sahlgren, M. (2005). An Introduction to Random Indexing. *Proceedings of the Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering*.

Salton, G. (1970). Automatic processing of foreign language documents. *Journal of the American Society for Information Science*. 21(3) :187–194.

Salton, G. (1971). *The SMART Retrieval System : Experiments in Automatic Document Processing*. Prentice Hall Inc. Englewood Cliffs,USA.

Salton, G. (1973). Experiments in multi-lingual information retrieval. *Information Processing Letters*. 2(1) :6–11.

Salton, G. (1975). A vector space model for information retrieval. *Communications of the ACM*. 18(11) :613–620.

Salton, G. (1989). *Automatic Text Processing : The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley.

Salton, G., et Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*. 24(5) :513–523.

Salton, G., et McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY.

Salton, G., Yang, C. S., et Yu, C. T. (1975). A Theory of Term Importance in Automatic Text Analysis. *Journal of the American Society for Information Science*. 26(1) :33–44.

Saporta, G. (1990). *Probabilités, Analyse des Données et Statistique*. Edit. Technip.

Schapire, R. E. (1990). The Strength of Weak Learnability. *Machine Learning*. 5(2) :197–227.

Schapire, R., Singer, Y., et Singhal, A. (1998). Boosting and Rocchio Applied to Text Filtering. *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*. 215–223.

Schikuta, E., et Erhart, M. (1997). The BANG-clustering system : grid-based data analysis. *Proceedings of Advances in Intelligent Data Analysis, Reasoning about Data, Second International Symposium*. 513–524.

Scholköpf, B., Platt, J., Shawe-Taylor, J., Smola, A. J., et Williamson, R. C. (1999). Estimating the support of a high-dimensional distribution. *Neural Computation*. 13(7) :1443–1471.

Scholz, M., et Klinkenberg, R. (2007). Boosting Classifiers for Drifting Concepts. *Intelligent Data Analysis*. 11(1) :3–28.

Schütze, H., Hull, D. A., et Pedersen, J. O. (1995). A comparison of classifiers and document representations for the routing problem. *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval*. 229–237.

Schultz, C. K., et Luhn, H. P. (1968). *Pioneer of Information Science – Selected Works*. Macmillan, London.

Schwab, I., Pohl, W., et Koychev, I. (2000). Learning to Recommend from Positive Evidence. *Proceedings of Intelligent User Interfaces. ACM Press*. 241–247.

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

Shah, R., Krishnaswamy, S., et Gaber, M. M., (2005). Resource-Aware Very Fast K-Means for Ubiquitous Data Mining. *Proceedings of the Second International Workshop on Knowledge Discovery for Data Streams*. 1–10.

Sheridan, P., et Ballerini, J. P. (1996). Experiments in Multilingual Information Retrieval using the SPIDER System. *Proceedings of the 19th ACM/SIGIR Conference*. 58–65.

Singhal, A., Buckley, C., et Mitra, M. (1996). Pivoted document length normalization. *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*. 21–29.

Singhal, A., Mitra, M., et Buckley, C. (1997). Learning routing queries in a query zone. *Proceedings of the annual international ACM SIGIR conference on research and development in information retrieval*. 25–32.

Sirois, S., et Mareshal, D. (2004). An interacting systems model of infant habituation. *Journal of Cognitive Neuroscience*. 16(8) :1352-1362.

Spinosa, E. J., Carvalho, A. P. L. F., et Gama, J. (2006). An online learning technique for coping with novelty detection and concept drift in data streams. *Proceedings of the 3rd International Workshop on Knowledge Discovery from Data Streams (IWKDDS 2006), in conjuntion with the 23rd International Conference on Machine Learning (ICML 2006)*.

Stanley, K. O. 2003. Learning Concept Drift with a Committee of Decision Trees. *Technical Report AI-03-302, Department of Computer Sciences, University of Texas at Austin, Austin, TX, USA*.

Steinwart, I. (2003). Sparseness of Support Vector Machines-Some Asymptotically Sharp Bounds. *Advances in Neural Information Processing Systems 16*. 1069–1076.

Stollnitz, E. J., DeRose, T. D., et Salesin, D. H. (1996). *Wavelets for Computer Graphics : Theory and Applications*. Morgan Kaufmann, San Francisco.

Syed, N., Liu, H., et Sung, K. (1999). Incremental learning with support vector machines. *Proc. of the Int. Joint Conf. on Artificial Intelligence (IJCAI)*.

Tang, B., Shepherd, M., Heywood, M., et Luo, X. (2005). Comparing Dimension Reduction Techniques for Document Clustering. *The Eighteenth Canadian Conference on Artificial Intelligence.* . 292–296.

Tax, D. M. J., et Duin, R. P. W. (2001). Uniform object generation for optimizing one-class classifiers. *Journal of Machine Learning Research*. 2 :155–173.

Tebri, H., et Boughanem, M. (2005). Optimisation de la fonction de décision dans un système de filtrage adaptatif. *International Symposium On Programming and Systems*.

Tebri, H., Boughanem, M., et Chrisment, C. (2005). Incremental profile learning based on a reinforcement method. *Proceedings of the 2005 ACM symposium on Applied computing*. 1096–1101.

Theodoridis, S., et Koutroumbas, K. (1999). *Pattern Recognition*. Academic Press.

Tmar, M., et Boughanem, M. (2000). Learning Profile in Routing : Comparison between Relevance and Gradient Back-Propagation. *String Processing in Information REtrieval (SPIRE)*. 253–259.

Treeratpituk, P., et Callan, J. (2006). Automatically labeling hierarchical clusters. *Proceedings of the 2006 international conference on Digital government research*. 167–176.

Tsymbal, A., Pechenizkiy, M., Cunningham, P., et Purronen, S. (2008). Dynamic integration of classifiers for handling concept drift. *Information Fusion*. 9(1) :56–68.

Ungar, L., et Foster, D. (1998). Clustering methods for collaborative filtering. *Workshop on Recommender systems at the 15th National Conference on Artificial Intelligence*.

Valizadegan, H., et Tan, P.-N. (2007). A Prototype-driven Framework for Change Detection in Data Stream Classification. *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. 88–95.

Valle, S., Li, W., et Qin, S. J. (1999). Selection of the Number of Principal Components : The Variance of the Reconstruction Error Criterion with a Comparison to Other Methods. *Ind. Eng. Chem. Res.*. 38(11) :4389–4401.

van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworths, London.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc.

Varsta, M., Heikkonen, J., et Millan, J. D. R. (1997). Context Learning with the Self-Organizing Map. *Proceedings of the Workshop on Self-Organizing Maps (WSOM)*. 197–202.

Varsta, M., Heikkonen, J., Lampinen, J., et Millán, J. del. R. (2001). Temporal Kohonen Map and the Recurrent Self-Organizing Map : Analytical and Experimental Comparison. *Neural Processing Letters*. 13(3) : 237–251.

Vinot, R. (2003). Improving Rocchio with weakly supervised clustering. *Proceedings of the European Conference on Machine Learning (ECML)*. 456–467.

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*. 11(1) :37–57.

Voegtlin, T. (2002). Recursive self-organizing maps. *Neural Networks*. 15(8-9) :979–991.

Voorhees, E. (2002). Overview of the TREC 2002. *Proceedings of the 11th Text REtrivel Conference (TREC-11). NIST, Gaithersburg, Maryland*. 1–15.

Waibel, A., Hanazawa, T., Hinton, F., Shikano, K., et Lang, K. J., (1989). Phoneme Recognition Using Time-Delay Neural Networks. *IEEE Transactio on Acoustics, Speech, and Signal Processing*. 37(3) :328–339.

Wang, H., Fan, W., Yu, P., et Han, J. (2003). Mining concept-drifting data streams using ensemble classifiers. *Proceedings of ACM SIGKDD International Conference on knowledge discovery and data mining*. 226–235.

Wang, B., Xu, H., Yang, Z., Liu, Y., Cheng, X., Bu, D., et Bai, S. (2001). TREC-10 experiments at CAS-ICT : Filtering, web and QA. *Proceedings of the 10th Text REtrivel Conference (TREC-10)*.

Wayne, C. L. (1998). Topic Detection and Tracking (TDT) : Overview and Perspective. *Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*.

Widmer, G., et Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*. 23(1) :69–101.

Widrow, B. et Hoff, M. E. (1960). Adaptive Switching Circuits. *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record*. 4 :96–104.

Wiener, E. D. (1995). A neural network approach to topic spotting. *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*. 317–332.

Wilkinson, R. (1991). Using the cosine measure in neural network for document retrieval. *Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*. 202–210.

Wong, S. K. M., et Raghavan, V. V. (1984). Vector space model of information retrieval : a reevaluation. *Proceedings of the 7th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR)*. 167–185.

Wu, L., Huang, X., Niu, J., Xia, Y., et Feng, Z. (2001). FDU at TREC-10 : Filtering, QA, web and video tasks. *Proceedings of the 10th Text REtrivel Conference (TREC-10)*.

Xu, L., Neufeld, J., Larson, B., et and Schuurmans, D. (2004) Maximum Margin Clustering. *Advances in Neural Information Processing Systems 17*. 1537–1544.

Yamron, J. P., Carp, I., Gillick, L., Lowe, S., et van Mulbregt, P. (1998). A hidden Markov model approach to text segmentation and event tracking. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. 1 :333–336.

Yang, Y. (1999). An Evaluation of Statistical Approaches to Text Categorization. *Journal of Information Retrieval*. 1(1/2) :69–90.

Yang, Y. (2001). A study of thresholding strategies for text categorization. *Proceedings of SIGIR'01*. 137–145.

Yang, L., Jin, R., et Sukthankar, R. (2008). Semi-supervised Learning with Weakly-Related Unlabeled Data : Towards Better Text Categorization. *Twenty-Second Annual Conference on Neural Information Processing Systems*.

Yang, Y., et Pedersen, J. O., (1997). A Comparative Study on Feature Selection in Text Categorization. *Proceedings of the 14th International Conference on Machine Learning (ICML)*. 412–420.

Yang, Y., Pierce, T., et Carbonell, J. (1998). A study of retrospective and on-line event detection. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. 28–36.

Yang, Y., Slattery, S., et Ghani, R. (2002). A Study of Approaches to Hypertext Categorization. *Journal of Intelligent Information Systems*. 18(2-3) :219-241.

Yang, Y., Zhang, J., Carbonell, J., et Jin, C. (2002). Topic-conditioned novelty detection. *Proc. of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 688–693.

Yang, C., et Zhou, J. (2006). HClustream : A Novel Approach for Clustering Evolving Heterogeneous Data Stream. *IEEE International Workshop on Mining Evolving and Streaming Data*. 682–688.

Ypma, A., et Duin, R. P. W. (1998). Novelty detection using self-organising maps. *Progress in Connectionist Based Information Systems*. 2 :1322-1325.

Yu, H., Han, J., et Chang, K. C. C. (2004). PEBL : Web page classification without negative examples. *IEEE Trans. Knowledge and Data Eng.* 16(1) :70–81.

Yu, H., Zhai, C., et Han, J. (2003). Text classification from positive and unlabeled documents. *Proceedings of the twelfth international conference on Information and knowledge management. ACM Press, New York, NY, USA.* 232–239.

Zhai, C., Jansen, P., Stoica, E., Grot, N., et Evans, D. A. (1999). Threshold Calibration in CLARIT Adaptive Filtering. *The Seventh Text REtrieval Conference (TREC-7).* 149–156.

Zhang, Y. (2004). Using bayesian priors to combine classifiers for adaptive filtering. *Proceedings of 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* 354–352.

Zhang, Y., et Callan, J. (2001). Maximum likelihood estimation for filtering thresholds. *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* 294–302.

Zhang, T., Ramakrishnan, R., et Livny, M. (1996). BIRCH : An Efficient Data Clustering Method for Very Large Databases. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada.* 103–114.

Zhang, T., Ramakrishnan, R., et Livny, M. (1997). BIRCH : A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery.* 1(2) :141–182.

Zhao, Y., et Karypis, G. (2002). Criterion functions for document clustering : experiments and analysis. *Technical Report no 01-40. Department of Computer Science/Army HPC Research Center.* . 1–30.

Zhong, S. (2005). Efficient online spherical k-means clustering. *IEEE Int. Joint Conf. on Neural Networks.* 5 :3180–3185.

Zhong, S. (2005). Efficient streaming text clustering. *Neural Networks.* 18(5-6) :790–798.

Zhu, Y., et Shasha, D. (2003). Efficient elastic burst detection in data streams. *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining.* 336–345.

Zipf, H. P. (1949). *Human Behavior and the Principle of Least Effort.* Addison-Wesley, Cambridge, Massachusetts.

Žižka, J., Hroza, J., Pouliquen, B., Ignat, C., et Steinberger, R. (2006). The selection of electronic text documents supported by only positive examples. *Proceedings of the 8th International Conference on the Statistical Analysis of Textual Data (JADT).* 19–21.

Résumé

De nombreuses applications génèrent et reçoivent des données sous la forme de flux continu, illimité, et très rapide. Cela pose naturellement des problèmes de stockage, de traitement et d'analyse de données qui commencent juste à être abordés dans le domaine des flux de données. Il s'agit, d'une part, de pouvoir traiter de tels flux à la volée sans devoir mémoriser la totalité des données et, d'autre part, de pouvoir traiter de manière simultanée et concurrente l'analyse des régularités inhérentes au flux de données et celle des nouveautés, exceptions, ou changements survenant dans ce même flux au cours du temps.

L'apport de ce travail de thèse réside principalement dans le développement d'un modèle d'apprentissage — nommé ILoNDF — fondé sur le principe de la détection de nouveauté. L'apprentissage de ce modèle est, contrairement à sa version de départ, guidé non seulement par la nouveauté qu'apporte une donnée d'entrée mais également par la donnée elle-même. De ce fait, le modèle ILoNDF peut acquérir constamment de nouvelles connaissances relatives aux fréquences d'occurrence des données et de leurs variables, ce qui le rend moins sensible au bruit. De plus, doté d'un fonctionnement en ligne sans répétition d'apprentissage, ce modèle répond aux exigences les plus fortes liées au traitement des flux de données.

Dans un premier temps, notre travail se focalise sur l'étude du comportement du modèle ILoNDF dans le cadre général de la classification à partir d'une seule classe en partant de l'exploitation des données fortement multidimensionnelles et bruitées. Ce type d'étude nous a permis de mettre en évidence les capacités d'apprentissage pures du modèle ILoNDF vis-à-vis de l'ensemble des méthodes proposées jusqu'à présent. Dans un deuxième temps, nous nous intéressons plus particulièrement à l'adaptation fine du modèle au cadre précis du filtrage d'informations. Notre objectif est de mettre en place une stratégie de filtrage orientée-utilisateur plutôt qu'orientée-système, et ceci notamment en suivant deux types de directions. La première direction concerne la modélisation utilisateur à l'aide du modèle ILoNDF. Cette modélisation fournit une nouvelle manière de regarder le profil utilisateur en termes de critères de spécificité, d'exhaustivité et de contradiction. Ceci permet, entre autres, d'optimiser le seuil de filtrage en tenant compte de l'importance que pourrait donner l'utilisateur à la précision et au rappel. La seconde direction, complémentaire de la première, concerne le raffinement des fonctionnalités du modèle ILoNDF en le dotant d'une capacité à s'adapter à la dérive du besoin de l'utilisateur au cours du temps. Enfin, nous nous attachons à la généralisation de notre travail antérieur au cas où les données arrivant en flux peuvent être réparties en classes multiples.

Mots-clés: apprentissage automatique, réseaux de neurones, classification supervisée et non supervisée, détection de nouveauté, flux de données, dérive de concept, filtrage basé sur le contenu, modélisation utilisateur, analyse des données multidimensionnelles, applications en Intelligence Artificielle

Abstract

Many applications produce and receive continuous, unlimited, and high-speed data streams. This raises obvious problems of storage, treatment and analysis of data, which are only just beginning to be treated in the domain of data streams. On the one hand, it is a question of treating data streams on the fly without having to memorize all the data. On the other hand, it is also a question of analyzing, in a simultaneous and concurrent manner, the regularities inherent in the data stream as well as the novelties, exceptions, or changes occurring in this stream over time.

The main contribution of this thesis concerns the development of a new machine learning approach — called ILoNDF — which is based on novelty detection principle. The learning of this model is, contrary to that of its former self, driven not only by the novelty part in the input data but also by the data itself. Thereby, ILoNDF can continuously extract new knowledge relating to the relative frequencies of the data and their variables. This makes it more robust against noise. Being operated in an on-line mode without repeated training, ILoNDF can further address the primary challenges for managing data streams.

Firstly, we focus on the study of ILoNDF's behavior for one-class classification when dealing with high-dimensional noisy data. This study enabled us to highlight the pure learning capacities of ILoNDF with respect to the key classification methods suggested until now. Next, we are particularly involved in the adaptation of ILoNDF to the specific context of information filtering. Our goal is to set up user-oriented filtering strategies rather than system-oriented in following two types of directions. The first direction concerns user modeling relying on the model ILoNDF. This provides a new way of looking at user's need in terms of specificity, exhaustivity and contradictory profile-contributing criteria. These criteria go on to estimate the relative importance the user might attach to precision and recall. The filtering threshold can then be adjusted taking into account this knowledge about user's need. The second direction, complementary to the first one, concerns the refinement of ILoNDF's functionality in order to confer it the capacity of tracking drifting user's need over time. Finally, we consider the generalization of our previous work to the case where streaming data can be divided into multiple classes.

Keywords: machine learning, neural networks, supervised and unsupervised classification, novelty detection, data streams, concept drift, content-based filtering, user modeling, multidimensional data analysis, Artificial Intelligence applications