THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *18 novembre 2013 (18/11/2013)* **par :**

Léa LAPORTE

La sélection de variables en apprentissage d'ordonnancement pour la recherche d'information : vers une approche contextuelle

---

JURY

Josiane MOTHE	Professeur des Universités	Co-Directrice
Sébastien DÉJEAN	Ingénieur de Recherche	Co-Directeur
Agnès FRONT	Maître de conférences HDR	Rapporteur
Eric GAUSSIER	Professeur des Universités	Rapporteur
Bernard DOUSSET	Professeur des Universités	Examinateur
Aurélien GARIVIER	Professeur des Universités	Examinateur
Jian-Yun NIE	Professeur des Universités	Examinateur
Benjamin PIWOWARSKI	Chargé de Recherche	Examinateur
Estelle DELPECH	Responsable R&D Nomao	Invitée

École doctorale et spécialité :
MITT : Image, Information, Hypermedia

Unité de Recherche :
Institut de Recherche en Informatique de Toulouse UMR 5505

Directeur(s) de Thèse :
Josiane MOTHE et *Sébastien DÉJEAN*

Rapporteurs :
Agnès FRONT et *Éric GAUSSIER*

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

AP Average Precision

CALMIP CALcul en MIdi-Pyrénées

DCG Discounted Cumulative Gain

IDF Inverse Document Frequency

INEX INitiative for the Evaluation of XML retrieval

INSA Institut National des Sciences Appliquées

FBS Forward Backward Splitting

FISTA Fast Iterative Shrinkage Thresholding Algorithm

HITS Hypertext Induced Topic Selection

LETOR LEarning TO Rank

LITIS Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes

LMIR Language Model for Information Retrieval

LSI Latent Semantic Indexing

MAP Mean Average Precision

MCP Minimax Concave Penalty

NDCG Normalized Discounted Cumulative Gain

RI Recherche d'Information

SVM Support Vector Machine

TF Term Frequency

TREC Text REtrieval Conference

Remerciements

Je tiens à remercier tout d'abord l'ensemble des membres de mon jury : Agnès Front et Eric Gaussier, qui ont accepté d'être les rapporteurs de mon mémoire, ainsi que Bernard Dousset, Aurélien Garivier, Jian-Yun Nie et Benjamin Piwowarski qui ont participé à cette soutenance en tant qu'examinateurs. Merci également à Estelle Delpech, invitée à venir représenter Nomao le jour J.

J'adresse ensuite mes plus chaleureux remerciements à mes deux directeurs de thèse, Josiane Mothe et Sébastien Déjean. Diriger une étudiante en Statistique ignorant tout de la Recherche d'Information sur une thèse en Informatique pouvait paraître un pari un peu risqué, mais il me semble que nous nous en sommes plutôt bien sorti ! Merci à Josiane pour avoir accepté de m'encadrer alors qu'elle ne me connaissait pas, pour son management énergique mais toujours humain, pour son investissement et pour tous ses conseils au cours de la thèse, que ce soit pour la recherche ou les recrutements ! Merci à Sébastien qui m'a offert l'opportunité en 2009 d'effectuer un stage dans le cadre de la plateforme biostatistique et qui m'a finalement recontactée quelques mois plus tard pour me dire qu'il aurait « peut être un sujet de thèse avec une collègue en informatique ». Merci pour tous ses conseils avant, pendant et après la thèse, pour son calme légendaire (si si c'est bien le mot) et pour sa disponibilité, surtout quand je viens à l'improviste dans son bureau pour cinq minutes et que je reste finalement une heure. Enfin, merci à tous les deux de m'avoir fait confiance, pour avoir supporté mon stress pendant ces trois ans, pour avoir su doser la pression sur mes épaules et pour vos réponses rapides et efficaces à mes nombreux mails désespérés !

Bien entendu, cette thèse n'aurait pas été possible sans l'action d'autres personnes et organismes. J'associe ici la région Midi-Pyrénées et l'entreprise Nomao, qui m'ont permis de réaliser ces travaux. Je tiens à remercier particulièrement Gilles Moncaubeig, co-fondateur de Nomao, qui m'a accueilli au sein de son entreprise pendant trois ans, m'offrant ainsi l'opportunité de découvrir l'univers des start-up.

J'associe à ces remerciements l'ensemble de mes collègues de Nomao, Overblog et Ebuzzing, qui m'ont accompagnée au cours de ces trois années. Alors merci aux anciens de Nomao qui étaient là quand je suis arrivée : JH et sa playlist endiablée, JB, Serge, Laurent, ...et à ceux qui ont rejoint l'aventure : Nico, Etienne, Nizar, Elisabeth, Estelle l'autre chercheuse qui a repris le flambeau de la R&D, Sam qui s'est déguisé en abeille pour ouvrir une bonne bouteille de champagne le jour de mon départ ! et tous les autres ...Merci à Damien, thésard avec Josiane chez Overblog qui m'a guidée dans l'entreprise et dans l'équipe ; aux joueurs de jeux de société du midi d'OB, à Adeline, la pourvoyeuse de biscuits (entre autres) ...La liste est longue et ils m'excuseront de ne pas tous les citer.

Je remercie très chaleureusement Stéphane Canu et Rémi Flamary, avec lesquels j'ai eu le privilège de collaborer sur la problématique de la sélection de variables et des SVM parcimonieux. Merci de m'avoir si gentiment accueillie d'abord à Rouen,

puis à Nice. C'est un véritable plaisir d'avoir travaillé avec vous. Merci Rémi pour m'avoir fait découvrir les sushis nutella banane lors de ma dernière soirée à Nice, c'est vrai qu'ils sont très bons !

J'adresse mes remerciements à l'ensemble des personnels techniques de l'IRIT qui m'ont aidée au cours de ma thèse, avec une pensée particulière pour Jean-Pierre Baritaud qui a été d'une aide précieuse sur la dernière ligne droite.

Je remercie l'ensemble de mes collègues de l'équipe SIG qui m'ont accompagnée au cours de cette thèse, qu'ils soient doctorants, MCF ou professeurs. Merci à Bernard Dousset, mon tuteur pédagogique, pour m'avoir fait partager son expérience en tant qu'enseignant-chercheur et pour toutes nos longues et joyeuses discussions à l'IRIT ou autour d'un bon repas. Merci notamment à Gilles, Guillaume, Max, Ronan, Olivier, Karen et les autres, qui m'ont toujours accueillie autour de leur table le midi, qui m'ont donné des conseils pour les dossiers de qualif et qui m'ont beaucoup appris sur les méandres de l'université et des postes. Merci aux doctorants et plus particulièrement à mes compères de bureaux, Anthony, compagnon de galère sur CalMip, Adrian, le bassiste qui me pique mon amoureux une fois par semaine pour jouer du Métal et Jonathan qui a eu le bonheur (malheur ?) d'accepter de bosser en binôme avec moi sur Yandex. Vous êtes tous les trois de joyeux lurons, c'est un véritable plaisir de partager le bureau avec vous !

Les derniers mais non les moindres, j'adresse ici mes plus chaleureux remerciements à ma famille. Merci papa et maman, Paul et Julien, mes deux grands frères adorés. Je peux dire sans hésitation que je n'aurais jamais réussi à en arriver là sans votre amour et votre soutien inconditionnels au cours des 26 dernières années. Merci à vous et à mes deux merveilleuses belle-soeurs pour m'avoir accompagnée dans cette aventure et pour m'avoir soutenue le jour J. Merci pour les cannelés, les merveilles et toutes les bonnes choses que vous avez préparées pour mon pot. Merci maman pour toutes tes relectures. Merci papa pour être tellement fier de moi. Les mots me manquent pour exprimer à quel point je suis heureuse d'être à vos côtés.

Enfin, bien que le mot merci ne soit pas suffisant ici pour celui qui est tous les jours à mes côtés depuis bientôt quatre ans, qui me soutient sans faiblir, qui m'aime, que j'aime : Merci Romain.

Résumé

L'augmentation massive et continue des volumes d'informations sur le Web et la diversité des documents disponibles complexifient la recherche de l'information pertinente vis-à-vis d'une requête d'un utilisateur. Le développement de systèmes de recherche d'information efficaces, rapides, prenant en compte le contexte utilisateur est un enjeu central en Recherche d'Information (RI).

Au cours de la dernière décennie, de nombreux travaux se sont concentrés sur la proposition de méthodes permettant d'optimiser de façon automatique l'ordonnancement des résultats restitués par les systèmes. Ce domaine de recherche, appelé apprentissage d'ordonnancement ou *learning to rank*, a conduit au développement de nombreuses approches et algorithmes. Ceux-ci, en combinant un grand nombre de modèles d'ordonnancement existants au sein d'une unique fonction utilisée pour ordonner les documents, ont permis d'améliorer la qualité des listes de résultats restitués.

Malgré tout, l'apprentissage d'ordonnancement présente plusieurs limites. D'une part, les algorithmes apprennent une unique fonction d'ordonnancement pour l'ensemble des requêtes soumises au système. Le contexte de la requête et l'utilisateur sont ainsi ignorés par le processus d'ordonnancement. D'autre part, le grand nombre de caractéristiques utilisées par les approches d'apprentissage d'ordonnancement peut influencer négativement la qualité du classement ainsi que la vitesse d'exécution des algorithmes. En effet, les algorithmes peinent à gérer l'importante volumétrie imposée par les données et la présence d'un grand nombre de caractéristiques potentiellement non pertinentes pour l'ordonnancement des résultats.

Des méthodes ont été proposées pour pallier l'une et l'autre de ces limites. D'une part, l'apprentissage d'ordonnancement dépendant des requêtes permet d'apprendre une fonction spécifique à chaque groupe de requêtes. L'ordonnancement est ainsi adapté au contexte. D'autre part, la sélection de variables, qui permet de supprimer des caractéristiques bruitées ou non pertinentes pour l'apprentissage, permet de réduire la dimensionnalité des modèles tout en contrôlant la qualité des caractéristiques utilisés. Chacune de ces méthodes, adaptées à l'apprentissage d'ordonnancement, s'est révélée efficace pour pallier la limite concernée.

Néanmoins, ces problématiques et leur résolution restent relativement peu étudiées. Par exemple, peu de travaux se sont intéressés à incorporer la sélection de variables à l'apprentissage de la fonction d'ordonnancement, préférant ajouter une étape de sélection préliminaire. Par ailleurs, à notre connaissance, il n'existe pas de travaux combinant l'adaptation au contexte et la sélection de variables à l'apprentissage d'ordonnancement.

Dans cette thèse, nous nous intéressons à l'utilisation de la sélection de variables pour la mise en place d'un système d'ordonnancement adaptatif.

Nous proposons et décrivons un système d'ordonnancement adaptatif dépendant des requêtes et basé sur la sélection de variables. Nous nous concentrons sur

le développement d'algorithmes de sélection de variables dédiés à l'apprentissage d'ordonnancement.

Nous proposons des méthodes basées sur l'utilisation de Support Vector Machine (SVM) qui incorporent la sélection à l'apprentissage du modèle. Nous proposons quatre algorithmes différents, exploitant les propriétés des régularisations mathématiques parcimonieuses convexes et non convexes pour effectuer la sélection. Nous montrons sur des jeux de données de référence que ces méthodes sont particulièrement efficaces en apprentissage d'ordonnancement. En effet, elles suppriment, en moyenne, quatre fois plus de variables que les méthodes de l'état de l'art, tout en étant généralement plus rapides et en conservant une qualité d'ordonnancement équivalente.

Afin d'évaluer les algorithmes et le système adaptatif dans un contexte d'exploitation industriel, nous avons également proposé un modèle d'évaluation de la pertinence à partir des actions utilisateurs. Ce modèle nous permet de construire de façon automatique des jeux de données de référence de grande taille à partir des fichiers de connexion des moteurs de recherche. L'évaluation des algorithmes sur un jeu de données issus d'un moteur de recherche commercial confirme la performance des algorithmes de sélection de variables. Les excellents résultats obtenus par les algorithmes de sélection de variables et par le modèle d'évaluation de la pertinence ont conduit à leur intégration au sein du moteur de recherche commercial partenaire Nomao.

Introduction générale

Au cours de ces dernières décennies, le volume d'informations disponibles sur Internet n'a pas cessé de croître. L'utilisateur se retrouve submergé de données de plus en plus nombreuses et diverses : pages web, tweets, flux rss, blogs, etc. Le développement de système de RI efficaces, capables de trouver l'information pertinente et adaptée au besoin de l'utilisateur dans de courts temps de réponse est un enjeu de la RI.

Depuis le début des années 70, de nombreux modèles de recherche et d'ordonnancement (modèles vectoriels, probabilistes, de langue, ...) ont été proposés [Amini 2013]. Ils permettent de déterminer les documents répondant au besoin exprimé par l'utilisateur au travers de sa requête et de les restituer sous forme de listes de résultats triés selon leur degré de pertinence. L'amélioration de ces listes, passant par l'optimisation de l'ordonnancement des résultats, est devenu un problème central. Pour le résoudre, certains travaux se sont concentrés sur la proposition de nouveaux modèles, tandis que d'autres se sont intéressés à l'optimisation des paramètres des modèles existants. Finalement, au cours de la dernière décennie, un nouveau champ de recherche, dédié à l'optimisation des classements des résultats et basé sur des techniques d'apprentissage automatique, a émergé : l'apprentissage d'ordonnancement ou *Learning to rank* [Liu 2011].

L'idée principale de l'apprentissage d'ordonnancement est de combiner les modèles existants afin de créer une unique fonction, dite d'ordonnancement, de meilleure qualité. Considérant des jeux de données composés de couples requête-document pour lesquels la pertinence est connue, les méthodes d'apprentissage d'ordonnancement apprennent automatiquement, à partir de ces données, la meilleure façon de combiner les modèles pour obtenir une liste optimale des résultats. Les modèles sont vus comme les caractéristiques des fonctions d'ordonnancement. De nombreux travaux ont mis en évidence que cette approche permet d'améliorer la qualité des listes restituées [Joachims 2002, Freund 2003, Burges 2005, Chapelle 2010].

Néanmoins, l'apprentissage d'ordonnancement présente deux limites principales.

Premièrement, les fonctions d'ordonnancement utilisent un grand nombre de caractéristiques, ce qui pose des problèmes liés, d'une part à la gestion de cette volumétrie et d'autre part, à la pertinence effective des caractéristiques utilisées. En effet, le nombre important de caractéristiques, couplé à la masse de couples requête-document considérés, est susceptible d'affecter fortement les temps de calcul des algorithmes d'apprentissage d'ordonnancement. Par ailleurs, beaucoup de caractéristiques sont utilisées par les algorithmes d'apprentissage d'ordonnancement, alors qu'elles peuvent être non pertinentes ou bruitées.

Deuxièmement, une seule fonction d'ordonnancement est utilisée pour l'ensemble des requêtes et des besoins utilisateurs. Les méthodes actuelles échouent ainsi à prendre en compte le contexte de la recherche et la diversité des besoins. Or,

de nombreux travaux ont montré que la prise en compte du contexte permet d'améliorer la satisfaction de l'utilisateur [Doan 2009, Hubert 2010, Adomavicius 2011, Bellot 2011].

Deux réponses ont été apportées par la communauté pour pallier ces limites. La sélection de variables, qui supprime des fonctions des caractéristiques inutiles pour l'ordonnancement, permet de réduire la dimension des problèmes et améliore généralement la qualité de la liste de résultats restituée [Geng 2007, Lai 2013a]. L'apprentissage d'ordonnancement dépendant des requêtes, qui apprend des fonctions spécifiques à chaque type de requêtes, permet de prendre en compte le contexte et améliore la qualité des listes restituées [Bai 2008, Geng 2008].

Les travaux actuels sur la sélection de variables dédiée à l'apprentissage d'ordonnancement ont donné de bons résultats, mais des méthodes potentiellement plus performantes n'ont pas été explorées. Peu de travaux se sont ainsi intéressés aux méthodes de sélection embarquées, qui permettent de supprimer les caractéristiques tout en apprenant le modèle final [Sun 2009, Lai 2013a, Lai 2013b]. Ils se sont globalement concentrés sur l'utilisation de SVM parcimonieux en norme ℓ_1. L'utilisation de normes non convexes, pourtant plus parcimonieuses, n'a, à notre connaissance, pas été étudiée. Elle constitue un champ d'amélioration que nous explorons dans ces travaux. La proposition d'algorithmes efficaces utilisant ce types de normes est la contribution majeure de cette thèse.

Par ailleurs, bien que les approches de sélection de variables et d'apprentissage d'ordonnancement soient efficaces, elles permettent de résoudre les problématiques séparément et non simultanément. Dans nos travaux, nous nous intéressons au développement d'un système d'ordonnancement adaptatif basé sur la sélection de variables, qui permet d'adapter l'ordonnancement au contexte tout en contrôlant le nombre de variables utilisé par les modèles. Notre contribution majeure est la proposition et l'évaluation de nouveaux algorithmes de sélection de variables dédiées à l'apprentissage d'ordonnancement, qui peuvent être utilisés dans le système adaptatif.

Plus spécifiquement, nos contributions se situent sur trois axes théoriques et un axe applicatif. Le premier axe est la proposition et la description d'un système d'apprentissage d'ordonnancement adaptatif, dépendant des requêtes et basé sur la sélection de variables. Le cœur de ce système est l'utilisation de méthodes d'apprentissage d'ordonnancement qui incorpore la sélection de variables, ce qui fait de la proposition de nouveaux algorithmes de sélection de variables l'élément central de nos travaux.

Le deuxième axe de nos contributions, le plus important, est la proposition de quatre algorithmes de sélection de variables dédiés à l'apprentissage d'ordonnancement, que nous évaluons. Nos algorithmes sont basés sur l'utilisation des SVM. Nous montrons qu'ils sont particulièrement performants, que nous considérions la qualité d'ordonnancement, le nombre de caractéristiques supprimées ou les temps d'exécution.

Le troisième axe des contributions concerne la proposition d'un modèle d'évaluation de la pertinence basé sur les clics utilisateurs. L'un des verrous rencontrés lors

de notre thèse est l'absence de jeux de données d'apprentissage pour une évaluation dans le cadre commercial. Le modèle proposé permet d'évaluer automatiquement, à partir des fichiers de connexion, la pertinence des documents vis-à-vis des requêtes soumises. Il est dédié à l'évaluation de la pertinence sur des moteurs à clics multiples, c'est-à-dire sur lesquels plusieurs actions sont possibles sur les résultats de recherche. Ce type de moteurs est de plus en plus fréquent, notamment dans le cas de recherches géoréférencées où les résultats sont présentés sous forme de liste et sur une carte géographique. Aucune collection d'évaluation n'existait à ce jour, ce qui a motivé le développement d'un modèle dédié.

Notre quatrième et dernière contribution est l'évaluation des algorithmes de sélection de variables dans un cadre commercial. Cette étude a permis de montrer que nos algorithmes sont performants, tant du point de vue de la qualité d'ordonnancement que de leur capacité à apprendre les modèles utilisant le moins de caractéristiques possibles en un temps raisonnable.

Ce manuscrit présente les travaux réalisés au cours des trois années de thèse. Il est structuré en trois parties, chacune décomposée en plusieurs chapitres.

La première partie introduit les notions de RI et d'apprentissage d'ordonnancement nécessaires à la compréhension de nos travaux. Les enjeux de l'apprentissage adaptatif sont explicités et nos contributions sont détaillées. Cette partie est décomposée en trois chapitres.

- Le chapitre 1 dresse un état de l'art des modèles de recherche et d'ordonnancement existants. Il présente la méthodologie d'évaluation en RI ainsi que les mesures d'évaluation utilisées.
- Le chapitre 2 introduit le principe de l'apprentissage d'ordonnancement. Il détaille les différents approches proposées au cours de la dernière décennie, ainsi que les algorithmes et méthodes associées. Les jeux de données d'évaluation de référence sont décrits. Les différentes approches sont comparées.
- Le chapitre 3 introduit l'apprentissage d'ordonnancement adaptatif. Il présente les différentes contributions. Les publications et présentations réalisées au cours de la thèse sont détaillées.

La deuxième partie regroupe nos contributions théoriques, ainsi que des références complémentaires spécifiques à l'apprentissage dépendant des requêtes et à la sélection de variables. Elle se décompose en cinq chapitres.

- Le chapitre 4 présente le système d'apprentissage d'ordonnancement adaptatif basé sur la sélection de variables que nous proposons. Nous dressons un état de l'art des approches d'apprentissage d'ordonnancement adaptatif existantes. Nous motivons l'utilisation de la sélection de variables dans ce cadre. Nous présentons les approches existantes de sélection de variables dédiées à l'apprentissage d'ordonnancement. Nous positionnons nos travaux par rapport à elles. Nous justifions la nécessité de proposer de nouvelles approches. Nous expliquons pourquoi nous choisissons d'étudier des méthodes basées sur les SVM.

- Le chapitre 5 explique l'utilisation des SVM pour la sélection de variables. Tout d'abord, nous rappelons le principe des SVM utilisés comme méthode d'apprentissage supervisé. Ensuite, nous indiquons comment les SVM doivent être modifiés dans le cadre de l'apprentissage d'ordonnancement. De plus, nous montrons comment l'utilisation de régularisations mathématiques dites parcimonieuses (ℓ_0, ℓ_1, *log*, ℓ_q $q < 1$ et MCP) permet d'effectuer la sélection de variables avec les SVM, simultanément à l'apprentissage de la fonction d'ordonnancement. Enfin, nous justifions l'utilisation de ces méthodes.
- Le chapitre 6 présente les deux premiers algorithmes de sélection de variables dédiés à l'apprentissage d'ordonnancement que nous proposons. Ceux-ci effectuent la sélection des caractéristiques en repondérant un algorithme d'apprentissage d'ordonnancement de l'état de l'art. Le schéma de repondération et l'utilisation d'un seuil annulant les caractéristiques permettent de supprimer les caractéristiques. Nous évaluons nos propositions sur cinq jeux de données de référence et les comparons à des algorithmes de l'état de l'art.
- Le chapitre 7 présente les deux derniers algorithmes de sélection de variables dédiés à l'apprentissage d'ordonnancement que nous proposons. Le premier algorithme effectue la sélection de variables par l'utilisation de la norme mathématique ℓ_1 et d'une approche de type Forward Backward Splitting (FBS) que nous définissons. Nous nommons cet algorithme RankSVM-ℓ_1. Le deuxième algorithme repondère RankSVM-ℓ_1 pour effectuer la sélection à l'aide des normes *log*, ℓ_q $q < 1$ et MCP. Contrairement aux méthodes présentées au chapitre 6, celles-ci n'ont pas besoin d'utiliser un seuil de nullité des variables : elles sont naturellement parcimonieuses. Les algorithmes sont évalués sur des jeux de données de référence et les résultats sont commentés.
- Le chapitre 8 présente le modèle d'évaluation de la pertinence pour les moteurs à clics multiples que nous proposons. Nous motivons la proposition de cette nouvelle méthode. Puis, nous explicitons et évaluons le modèle à partir de fichiers de connexion issus du moteur de recherche commercial Nomao. Les résultats sont présentés et commentés.

La troisième partie présente l'évaluation des algorithmes de sélection de variables dans un cadre d'exploitation commercial, qui constitue une étape préliminaire de l'évaluation du système d'ordonnancement adaptatif. Elle est décomposée en deux chapitres.

- Le chapitre 9 présente la construction du jeu de données d'apprentissage à partir des fichiers de connexion du moteur de recherche Nomao. Nous déterminons les jugements de pertinence des couples requête-document à l'aide du modèle d'évaluation présenté au chapitre 8. Nous détaillons les propriétés du jeu de données et nous décrivons brièvement les caractéristiques utilisées.
- Le chapitre 10 présente les résultats de l'évaluation des algorithmes de sélection de variables dans un cadre d'exploitation commercial. Les différents algorithmes sont comparés entre eux et à l'état de l'art. Les résultats expérimentaux sont commentés. Nous expliquons également comment ces travaux

ont été intégrés au sein du moteur de RI géoréférencé Nomao.

Afin de faciliter la lecture, des listes des abréviations utilisées et des algorithmes présentés sont intégrées à ce manuscrit. Pour des explications mathématiques détaillées concernant l'optimisation ou les méthodes d'apprentissage supervisées, le lecteur est invité à se référer aux chapitres 21 et 22 de l'ouvrage de Tie-Yan Liu [Liu 2011].

Première partie

Vers un système d'ordonnancement adaptatif : état de l'art, enjeux et contributions

Résumé de la première partie

Dans cette première partie, nous présentons le contexte général de nos travaux, l'apprentissage d'ordonnancement (*learning to rank*) en RI, ainsi que nos propositions et contributions. Elle est composée de trois chapitres.

Dans le chapitre 1, nous rappelons des notions de RI nécessaires à la bonne compréhension des travaux. Nous présentons les modèles de recherche et d'ordonnancement utilisés comme caractéristiques des fonctions d'ordonnancement. Nous décrivons la méthodologie d'évaluation des systèmes de RI ainsi que les mesures utilisées.

Dans le chapitre 2, nous présentons le principe de l'apprentissage d'ordonnancement en RI. Nous le positionnons par rapport aux modèles de référence présentés précédemment. Nous dressons un état de l'art des approches d'apprentissage d'ordonnancement et des algorithmes de référence. Nous décrivons les collections de référence utilisées pour l'évaluation et comparons les résultats obtenus par les différentes approches sur ces jeux de données.

Dans le chapitre 3, nous motivons nos travaux. Nous détaillons nos contributions et nous indiquons les publications, soumises et acceptées, qui y sont rattachées. Nous incluons également la liste complète des publications et communications effectuées au cours de la thèse.

Chapitre 1

Bases de la recherche d'information : modèles et évaluation

Sommaire

Dans ce chapitre, nous présentons un ensemble de notions de Recherche d'Information (RI) nécessaires pour la bonne compréhension du contexte et des travaux. Nous introduisons dans un premier temps les modèles de recherche et d'ordonnancement basés sur la pertinence d'un document vis-à-vis d'une requête ainsi que les modèles d'ordonnancement basés sur l'importance propre des documents. Dans un second temps, nous présentons le principe de l'évaluation en RI et les mesures utilisées.

1.1 Modèles de recherche pour l'ordonnancement

L'ordonnancement, ou *ranking*, des documents restitués par un système de RI suite à une requête d'un utilisateur est une thématique majeure en RI. Les modèles de recherche fournissent généralement un score qui est utilisé pour ordonner les documents restitués. Ces modèles peuvent être regroupés en deux grandes catégories : ceux basés sur la similarité des documents par rapport à la requête soumise et ceux basés sur l'importance des documents [Liu 2011].

1.1.1 Modèles de recherche basés sur la similarité requête-document

Ces modèles visent à déterminer la ressemblance d'un document par rapport à une requête d'un utilisateur. Le modèle booléen [Lancaster 1974] a été le premier proposé dans les années 70. Basé sur la théorie des ensembles, il restitue l'ensemble

des documents qui contiennent exactement tous les termes de la requête. Les modèles les plus récents fournissent généralement un score de similarité entre la requête et le document. Ce dernier a d'autant plus de chance d'être pertinent que son score de similarité vis-à-vis de la requête considérée est important. Ce score est alors utilisé pour ordonner les documents restitués par le système pour une requête donnée. Il est calculé différemment suivant le modèle de RI considéré.

Le modèle vectoriel [Salton 1975] représente requête et document sous forme de vecteurs dans l'espace des termes et utilise leur cosinus comme degré de similarité.

Le modèle Latent Semantic Indexing (LSI) [Deerwester 1990] utilise la décomposition en valeurs singulières pour réduire l'espace des termes à un espace des concepts dans lequel la similarité entre requête et document est mesurée.

Les modèles probabilistes, comme BM25 [Robertson 1994] ou les modèles de langue [Ponte 1998], se basent sur la théorie des probabilités pour mesurer la similarité entre requêtes et documents.

Dans la suite de ce chapitre, nous présentons de façon plus détaillée les modèles vectoriels, BM25 et de langue, dont les résultats sont utilisés comme caractéristiques dans les approches d'apprentissage d'ordonnancement.

Le lecteur désirant plus d'informations pourra se reporter aux ouvrages du domaine [Baeza-Yates 1999, Croft 2009, Amini 2013].

1.1.1.1 Le modèle vectoriel (*Vector Space Model*)

Le modèle vectoriel [Salton 1975] permet de calculer un degré (ou score) de ressemblance ou similarité entre la requête et le document. Requêtes et documents sont représentés par les vecteurs des poids des termes qui les constituent. Le score de ressemblance est exprimé comme la similarité entre le vecteur de la requête et le vecteur du document, généralement définie comme le cosinus entre les deux vecteurs (*cf.* définition 1).

Définition 1 (Modèle vectoriel)**.**

Soient
- m le nombre de termes
- w_{tq} le poids du terme t dans la requête q
- w_{ti} le poids du terme t dans le document d_i
- $\mathbf{q} = [w_{1q} \dots w_{mq}]$ et $\mathbf{d}_i = [w_{1i} \dots w_{mi}]$ les représentations de la requête et du document dans l'espace des termes

Alors, le degré de pertinence du document d_i par rapport à la requête q est donné par la mesure de similarité exprimée comme le cosinus entre les deux vecteurs tel que :

$$s(\mathbf{d}_i, \mathbf{q}) = \frac{\sum_{t=1}^{m}(w_{ti}.w_{tq})}{\sqrt{\sum_{t=1}^{m}(w_{ti})^2}\sqrt{\sum_{t=1}^{m}(w_{tq})^2}} \tag{1.1}$$

Le poids d'un terme dans une requête ou un document est généralement exprimé à l'aide de la pondération $TF.IDF$ [Robertson 1976]. Elle est définie à partir des quantités *Term Frequency (TF)* et *Inverse Document Frequency (IDF)*.

La quantité TF représente le nombre d'occurrences d'un terme au sein d'un document ou d'une requête.

La quantité IDF exprime l'importance d'un terme au sein du corpus considéré, telle que :

$$IDF(t) = \log \frac{N}{n_t} \tag{1.2}$$

où N est le nombre total de documents du corpus et n_t le nombre de documents qui contiennent le terme t. La pondération $TF.IDF$ d'un terme t dans un document d_i est alors exprimée comme le produit des deux quantités précédentes :

$$TF.IDF(t, d_i) = TF(t, d_i).IDF(t) \tag{1.3}$$

Cette pondération prend en compte le fait que tous les documents n'ont pas le même pouvoir discriminant. L'importance d'un terme est définie à partir de sa fréquence dans le document et les termes peu discriminants car trop fréquents sont pénalisés.

D'autres modèles plus récents, comme BM25 et les modèles de langue, sont basés sur les probabilités. Ils sont considérés comme plus performants que le précédent.

1.1.1.2 Le modèle BM25 (*Okapi BM25 model*)

Le modèle BM25, ou Okapi BM25 en référence au premier système sur lequel il a été implémenté, fait partie d'une famille de modèles probabilistes proposés par Robertson et Walker en 1994 [Robertson 1994]. Ces modèles de pertinence définissent le score de similarité $s(d_i, q)$ d'un document d_i pour une requête q à partir des probabilités de pertinence et de non pertinence des documents tel que :

$$s(d_i, q) = \frac{P(R|d_i)}{P(\overline{R}|d_i)} \tag{1.4}$$

où $P(R|d_i)$ est la probabilité que le document d_i fasse partie de l'ensemble des documents pertinents tandis que $P(\overline{R}|d_i)$ est la probabilité que le document d_i fasse partie de l'ensemble des documents non pertinents pour la requête q. Ce score peut être réécrit comme la somme des poids de pertinence w_j des termes t_j, $j = 1, \ldots, m$ présents dans le document :

$$s(d_i, q) = \sum_{j=1}^{m} w_j \mathbb{1}_{\{t_j \in d_i\}} \tag{1.5}$$

où $\mathbb{1}_{\{t_j \in d_i\}}$ est la fonction indicatrice qui vaut 1 si le terme t_j est présent dans le document d_i et 0 sinon. Les poids de pertinence sont exprimés à partir des probabilités de pertinence des termes pour le document. Robertson et Walker [Robertson 1994] ont proposé plusieurs approximations de ces probabilités et de ces scores. La version

la plus connue est le modèle BM25 (*cf.* définition 2), exprimé à partir des quantités TF et IDF présentées à la section 1.1.1.1.

Définition 2 (Modèle BM25)**.**

Soient
– TF et IDF les quantités définies en 1.1.1.1,
– $|d_i|$ le nombre de termes composant le document d_i,
– $avdl$ le nombre moyen de termes composant les documents du corpus,
– k_1 et b des paramètres d'ajustement,
Alors le score de similarité du modèle BM25 entre le document d_i et la requête q, noté $BM25(d_i, q)$, est défini de la façon suivante :

$$BM25(d_i, q) = \sum_{j=1}^{m} \frac{IDF(t_j).TF(t_j, d_i).(k_1 + 1)}{TF(t_j, d_j) + k_1.(1 - b + b\frac{|d_i|}{avdl})} \tag{1.6}$$

Le modèle BM25 est populaire en RI, car il est très performant. Les modèles de langues présentés ci-après sont également très populaires.

1.1.1.3 Les modèles de langue

L'idée fondatrice des modèles de langue (*Language Model for Information Retrieval (LMIR)*) [Ponte 1998] est d'associer à chaque document un modèle qui le caractérise. Celui-ci permet de calculer la probabilité qu'une séquence de termes ait été générée à partir d'un document. Considérant une requête q constituée de m termes, la probabilité qu'un document d_i soit pertinent pour q est égale à la probabilité $P(q|d_i)$ que la requête q ait été générée par le document d_i. Sous l'hypothèse d'indépendance des termes, il s'agit du produit des probabilités $P(t_j|d_i)$ que les termes t_j, $j = 1, \ldots, m$ aient été générés par le document, soit :

$$P(q|d_i) = \prod_{j=1}^{m} P(t_j|d_i) \tag{1.7}$$

L'estimateur le plus simple de $P(t_j|d_i)$ est celui du maximum de vraisemblance, soit $P_{MV}(t_j|d_i) = \frac{c(t_j,d_i)}{\sum_{k=1}^{m} c(t_k,d_i)}$ où $c(t, d_i)$ est le nombre d'occurrences du terme t dans le document d_i. L'inconvénient majeur de cette approximation est qu'elle affecte une probabilité nulle aux termes non présents. Ainsi, si un document contient tous les termes de la requête sauf un, sa probabilité de pertinence sera nulle, alors qu'il répond potentiellement à une partie de la requête. Des méthodes de lissage ont été proposées pour pallier ce problème. Zhai *et al.* ont étudié les propriétés des trois méthodes d'estimations les plus fréquemment utilisées [Zhai 2001], à savoir les lissages de Jelinek-Mercer, de Dirichlet et le lissage *Absolute Discounting*, présentées respectivement aux définitions 3, 4 et 5.

Définition 3 (Lissage de Jelinek-Mercer)**.**

Soient
- $P(t_j|d_i)$ (resp. $P(t_j|C)$) la probabilité que le terme t_j ait été généré par document d_i (resp. par le corpus de documents C)
- $c(t_j, d_i)$ (resp. $c(t_j, C)$) le nombre d'occurrences du terme t_j dans le document d_i (resp. dans le corpus C)
- $P_{MV}(t_j|d_i) = \frac{c(t_j,d_i)}{\sum_{k=1}^{m} c(t_k,d_i)}$ (resp. $P_{MV}(t_j|C) = \frac{c(t_j,C)}{\sum_{k=1}^{m} c(t_k,C)}$) l'estimateur du maximum de vraisemblance de $P(t_j|d_i)$ (resp. de $P(t_j|C)$)
- $\lambda \in [0, 1]$ un facteur de lissage

Alors la probabilité que le terme t_j ait été généré par le document d_i selon le lissage de Jelinek-Mercer s'écrit sous la forme d'un modèle de mélange :

$$P_{JM}(t_j|d_i) = (1 - \lambda)P_{MV}(t_j|d_i) + \lambda P_{MV}(t_j|C)$$

Définition 4 (Lissage de Dirichlet)**.**

Soient
- $c(t_j, C)$ le nombre d'occurrences du terme t_j dans le corpus C
- $P_{MV}(t_j|C) = \frac{c(t_j,C)}{\sum_{k=1}^{m} c(t_k,C)}$ l'estimateur du maximum de vraisemblance de $P(t_j|C)$, probabilité que le terme t_j ait été généré par le corpus C
- $\mu > 0$ un paramètre qui contrôle le lissage

Alors la probabilité que le terme t_j ait été généré par le document d_i selon le lissage de Dirichlet s'écrit de la façon suivante :

$$P_{DIR}(t_j|d_i) = \frac{c(t_j, d_i) + \mu P_{MV}(t_j|C)}{\sum_{k=1}^{m} c(t_k, d_i) + \mu}$$

Définition 5 (Lissage *Absolute Discounting*)**.**

Soient
- $c(t_j, d_i)$ le nombre d'occurrences du terme t_j dans le document d_i
- $|d_i|_u$ le nombre de termes uniques dans le document
- $\delta \in [0, 1]$ une constante qui diminue l'importance des termes connus par le modèle de langue

Alors la probabilité que le terme t_j ait été généré par le document d_i selon le lissage *Absolute Discounting* s'écrit de la façon suivante :

$$P_{ABS}(t_j|d_i) = \frac{max(c(t_j, d_i) - \delta, 0) + \delta|d|_u}{\sum_{k=1}^{m} c(t_k, d_i)}$$

L'étude de Zhai *et al.* [Zhai 2001] suggère que la performance des modèles de langues est influencée par le choix des paramètres par défaut et qu'elle est variable suivant le type de requêtes. Ainsi, le lissage de Jelinek-Mercer donne de meilleurs résultats sur les requêtes longues que sur les requêtes courtes, contrairement aux autres lissages qui obtiennent de meilleurs résultats sur les requêtes courtes, même s'il est difficile de déterminer quelle méthode est globalement la plus efficace.

1.1.2 Modèles d'ordonnancement basés sur l'importance des documents et leurs liens

D'autres modèles permettent d'ordonner les documents en prenant en compte leur importance propre. Ils sont basés en particulier sur l'analyse de la structure des liens entre pages Internet. Nous détaillons dans cette partie les modèles *Hypertext Induced Topic Selection (HITS)* et PageRank , qui en constituent de bons exemples.

1.1.2.1 Modèle HITS

Le modèle HITS [Kleinberg 1999] est basé sur les notions d'autorité (*Authority*) et de centralité (*Hubness*) d'un document ou page web.

L'autorité $A(p)$ d'une page p traduit le fait qu'elle est une page de référence sur le sujet. En pratique, il s'agit d'un site qui dispose d'un grand nombre de liens entrants, il est donc cité par un grand nombre d'autres sites. Une page centrale, ou *hub*, possède un grand nombre de liens sortants, elle fait ainsi référence à un grand nombre d'autres sources d'informations. Le poids d'une page centrale $H(p)$ traduit sa qualité. Les poids d'autorité et de centralité des pages sont calculés de façon itérative, comme décrit à la définition 6.

Définition 6 (HITS - autorité et centralité)**.**

Soient
- p une page Internet,
- $P(p)$ l'ensemble des pages parentes de p , c'est-à-dire des pages pointant vers p
- $F(p)$ l'ensemble des pages filles de p, c'est-à-dire des pages vers lesquelles p pointe

alors, à chaque itération i, l'autorité $A(p)$ et la centralité $H(p)$ de la page p sont mises à jour de la façon suivante :

$$A^{i+1}(p) = \sum_{p_j \in P(p)} H^i(p_j) \tag{1.8}$$

et

$$H^{i+1}(p) = \sum_{p_j \in F(p)} A^i(p_j) \tag{1.9}$$

Les notions d'autorité et de centralité se renforcent ainsi mutuellement. Une bonne page centrale est une page qui pointe vers de bonnes pages de référence, tandis qu'une bonne page de référence est une page qui est citée par un grand nombre de bonnes pages centrales.

1.1.2.2 Modèle PageRank

Le modèle PageRank a été breveté en 2001 [Page 2001]. Il est notamment connu pour être utilisé par le moteur de recherche Google. Le modèle PageRank est basé sur la probabilité qu'un internaute cliquant de façon aléatoire sur des liens arrive sur la page p. Ce modèle détermine un score d'importance de chaque page, présenté à la définition 7, utilisé pour le classement de celles-ci.

Définition 7 (PageRank d'une page)**.**

Soient
- d la probabilité qu'un utilisateur continue de parcourir des pages,
- N le nombre total de pages,
- $P(p)$ l'ensemble des pages parentes de p, c'est-à-dire pointant vers p,
- $S(p_j)$ le nombre de liens sortant de la pages p_j

Alors le PageRank d'une page p, noté $PR(p)$ est défini de la façon suivante :

$$PR(p) = \frac{1-d}{N} + d \sum_{p_j \in P(p)} \frac{PR(p_j)}{S(p_j)} \tag{1.10}$$

Le PageRank d'un document ou d'une page web est ainsi défini à partir du PageRank des autres pages.

D'autres modèles d'importance basés sur l'analyse de la structure des liens ont été proposés. Ainsi, Nie *et al.* [Nie 2006] se sont intéressés à la prise en compte de la thématique dans le calcul des scores HITS et PageRank. Ainsi, les scores proposés ajustent l'importance des pages en fonction de leur adéquation avec une thématique donnée. Par exemple, dans le modèle HITS, une page pourra avoir un score d'autorité élevé pour une certaine thématique et faible pour une autre. Les modèles originaux ne considèrent pas ces nuances pour l'évaluation de l'importance des pages. Xue *et al.* [Xue 2005] se sont intéressés aux limites rencontrées par les modèles précédents lorsque peu de liens sont présents entre pages, ce qui arrive lorsque les pages commencent à devenir importantes. Ils ont ainsi proposé le modèle HostRank, qui tient compte de la structure hiérarchique du web et améliore l'ordonnancement des nouvelles pages. Enfin, d'autres travaux [Shakery 2003][Qin 2005] se sont intéressés à l'élaboration de modèles utilisant à la fois la structure des liens entre pages et le contenu de celles-ci.

Tous les modèles présentés peuvent être utilisés pour ordonner les documents sélectionnés pour une requête donnée, en utilisant le score d'importance ou de pertinence. L'apprentissage d'ordonnancement utilise les résultats de ces modèles comme caractéristiques des fonctions d'ordonnancement, comme nous l'expliquons au chapitre 2. Dans le domaine, la qualité de la liste des résultats obtenue est évaluée à l'aide d'un protocole d'évaluation et de mesures dédiées que nous présentons dans la section suivante.

1.2 Évaluation des modèles de RI

L'évaluation est une composante centrale en RI. La méthodologie la plus largement utilisée pour évaluer une approche est celle décrite dans le projet Cranfield [Cleverdon 1966]. Dans cette section, nous présentons dans un premier temps la méthodologie d'évaluation. Dans un second temps, nous détaillons les principales mesures d'évaluation et nous précisons celles que nous utilisons dans ces travaux.

1.2.1 Méthodologie d'évaluation

L'objectif de l'évaluation est de quantifier la capacité d'un modèle à retrouver les documents pertinents et à les restituer selon leur degré de pertinence. La procédure d'évaluation est basée sur la construction de collections de référence pour lesquelles la pertinence de chaque document vis-à-vis d'une requête est connue. Nous présentons le principe général de cette méthodologie dans une première section, puis, dans une seconde section, nous détaillons plus spécifiquement la construction des jugements de référence.

1.2.1.1 Principe général

La procédure d'évaluation se décompose en quatre étapes.

Dans un premier temps, un ensemble de requêtes est sélectionné ; pour chacune d'elles, un sous-ensemble de documents, retrouvés par un système de RI, est collecté.

Dans un second temps, un jugement de pertinence est affecté à chaque couple requête-document, qui traduit le fait que le document est pertinent vis-à-vis de la requête. Il peut s'agir d'un jugement binaire (pertinent *vs* non pertinent), catégoriel (très pertinent, moyennement pertinent, peu pertinent, non pertinent) ou encore d'un score réel.

Dans un troisième temps, le ou les modèles de recherche à évaluer sont utilisés pour calculer les scores de similarité de chaque couple requête-document (ou le sous-ensemble de documents à restituer dans le cas de modèles ensemblistes comme le modèle booléen). Pour chaque requête, un sous-ensemble de documents supposés pertinents, éventuellement ordonnés, est ainsi obtenu.

Enfin, la dernière étape consiste à comparer ce sous-ensemble à la liste de documents pertinents de référence. Plusieurs mesures d'évaluation (*cf.* section 1.2.2) sont utilisées pour quantifier la performance du ou des modèles étudiés.

Le point le plus délicat de cette méthodologie est la construction de la référence : comment évaluer la pertinence pour chaque couple requête-document ? Plusieurs approches ont été utilisées. Nous les détaillons dans la section suivante.

1.2.1.2 Différentes approches pour la construction de la référence

Dans la section précédente, nous avons indiqué que pour chaque requête, un ensemble de documents potentiellement pertinents, restitués par un système de RI, est collecté. La référence est alors construite en évaluant la pertinence de chaque document vis-à-vis de la requête considérée.

Au cours des expérimentations Cranfield [Cleverdon 1966], qui ont défini la méthodologie d'évaluation employée aujourd'hui, le nombre de documents était relativement faible, de l'ordre de quelques milliers. Le jugement manuel de la pertinence par des experts était réalisable de façon exhaustive.

Au cours des campagnes d'évaluation plus récentes, comme Text REtrieval Conference (TREC) ou INitiative for the Evaluation of XML retrieval (INEX), le nombre de systèmes interrogés et de documents restitués est devenu trop important pour une évaluation exhaustive. TREC a proposé d'utiliser la méthode du sondage (*pooling*) pour la collecte des documents. Chaque requête est soumise à plusieurs systèmes. Les 1000 premiers documents restitués par chaque système sont regroupés dans un même ensemble. L'évaluation de ce sous-ensemble réduit est alors effectuée manuellement par des experts.

Malgré l'utilisation du *pooling*, la constitution des références pour les campagnes d'évaluation reste une procédure longue et coûteuse. Elle est aussi considérée comme déconnectée des utilisations réelles. Des travaux [Joachims 2002] proposent d'utiliser les logs de connexion et des modèles de clics pour extraire des jugements de pertinence implicites à partir des requêtes et actions des utilisateurs sur un moteur réel. Ces méthodes sont détaillées au chapitre 9, où nous présentons un nouveau modèle d'évaluation de la pertinence, adapté aux documents multicliquables, que nous avons proposé au cours de la thèse.

Dans la section suivante, nous introduisons les mesures d'évaluation utilisées pour comparer la performance des différents modèles de RI.

1.2.2 Mesures d'évaluation

Les principales mesures utilisées en RI sont basées sur les notions de rappel et de précision. Le rappel correspond au ratio de documents pertinents restitués par rapport au nombre total de documents pertinents de la collection. La précision correspond au ratio de documents pertinents parmi tous les documents restitués. Dans la suite de cette section, nous présentons les mesures les plus utilisées dans la littérature du domaine. Nous précisons celles que nous considérons dans ce rapport.

1.2.2.1 Précision au rang k, précision moyenne et moyenne de la précision moyenne

La moyenne des précisions moyennes, *Mean Average Precision (MAP)*, est une mesure permettant d'évaluer la qualité de l'ordonnancement d'une liste de résultats dans le cas de jugements binaires de pertinence (documents pertinents *vs* documents non pertinents). Elle est basée sur le calcul de la précision au rang k ($P@k$) et de la précision moyenne, *Average Precision (AP)*).

La précision au rang k pour la requête q ($P_q@k$) correspond au ratio de documents pertinents présents parmi les k premiers documents restitués dans la liste de résultats pour cette requête :

$$P_q@k = \frac{\#\text{documents pertinents sélectionnés jusqu'au rang k}}{k} \tag{1.11}$$

La précision moyenne de la requête q est alors calculée à partir des $P_q@k$ de la façon suivante :

$$AP_q = \frac{\sum_{i=1}^{k} P@i.\mathbb{1}_{\{\text{le document i est pertinent}\}}}{\#\text{documents pertinents pour la requête q}} \tag{1.12}$$

La MAP est définie comme la moyenne des AP sur l'ensemble des requêtes (*cf.* définition 8).

Définition 8 (MAP)**.**

Soient
- AP_q la précision moyenne pour la requête q définie à l'équation 1.12
- Q le nombre de requêtes de la collection considérée

alors la MAP est définie comme la moyenne des AP sur l'ensemble des requêtes :

$$MAP = \frac{1}{Q}\sum_{q=1}^{Q} AP_q \tag{1.13}$$

La MAP ne permet pas d'évaluer la performance d'un système lorsque plus de deux classes de pertinence sont considérées, c'est-à-dire, lorsque plusieurs degrés de pertinence sont utilisés, par exemple : très pertinent, moyennement pertinent, peu pertinent et non pertinent. Dans ce cas, le *Normalized Discounted Cumulative Gain (NDCG)*, qui prend en compte le degré de pertinence pour l'évaluation de la performance, est préféré. Il est présenté à la section suivante.

1.2.2.2 Normalized Discounted Cumulative Gain

Le NDCG pour la requête q au rang k ($NDCG_q@k$) est calculé à partir du Discounted Cumulative Gain (DCG) au rang k pour la requête q ($DCG_q@k$), défini

de la façon suivante :

$$DCG_q@k = \sum_{i=1}^{k} \frac{2^{r(i)} - 1}{log_2(i+1)} \tag{1.14}$$

où $r(i)$ correspond au degré de pertinence du document à la position i.

La mesure ainsi obtenue présente l'inconvénient de ne pas être à valeurs dans l'intervalle $[0, 1]$. Le $DCG_q@k$ est alors normalisé par le terme Z_k, valeur maximale du $DCG_q@k, \forall k$, de façon à obtenir le $NDCG_q@k$:

$$NDCG_q@k = \frac{1}{Z_k} DCG_q@k \tag{1.15}$$

à valeurs dans $[0, 1]$.

Le $NDCG@k$ pour l'ensemble des requêtes du jeu de données est enfin défini comme la moyenne des $NDCG_q@k$ sur l'ensemble des Q requêtes (*cf.* définition 9).

Définition 9 ($NDCG@k$)**.**

Soient
- $NDCG_q@k$ le NDCG à la position k pour la requête q défini à l'équation 1.15
- Q le nombre total de requêtes de la collection considérée

Alors le NDCG à la position k est définie comme la moyenne des $NDCG_q@k$ pour l'ensemble des requêtes :

$$NDCG@k = \frac{1}{Q} \sum_{q=1}^{Q} NDCG_q@k \tag{1.16}$$

D'autres mesures que la MAP et le NDCG peuvent être utilisées pour quantifier l'adéquation entre la liste ordonnée de résultats obtenue via un modèle de RI et la liste de référence. Une alternative est de considérer une mesure de corrélation des rangs.

1.2.2.3 Corrélations des rangs

Dans cette section, nous nous limitons à une mesure de corrélation des rangs basée sur le τ de Kendall, bien qu'il soit possible d'utiliser d'autres mesures de corrélations, comme le ρ de Spearman.

Le τ de Kendall se base sur le nombre de paires concordantes et discordantes pour évaluer la corrélation entre une liste de référence π_r et une liste candidate π. Une paire est dite concordante si les deux éléments a et b qui la constituent sont classés dans le même ordre dans les deux listes : par exemple si a est au dessus de b dans π_r et dans π. Au contraire, une paire est dite discordante si les deux éléments qui la composent sont classés dans un ordre différent dans π_r et π : par exemple

si a est au dessus de b dans π_r et en dessous dans π. L'expression finale du τ de Kendall est présenté à la définition 10.

Définition 10 (τ de Kendall)**.**

Soient
- Q le nombre de paires concordantes,
- P le nombre de paires discordantes,
- n le nombre total de paires,

Alors le τ de Kendall entre deux listes π_r et π est défini de la façon suivante :

$$\tau(\pi_r, \pi) = \frac{Q - P}{0.5n(n-1)} \tag{1.17}$$

En pratique, toutes les inversions ne sont pas aussi pénalisantes les unes que les autres. Or la formule 10 ne fait pas de distinction entre une inversion concernant un document très pertinent et un document non pertinent et une inversion entre un document très pertinent et un document moyennement pertinent. Or, la deuxième inversion est bien moins pénalisante pour la qualité d'ordonnancement que la première. Pour remédier à ce problème, le τ de Kendall pondéré a été proposé. Il est présenté à la définition 11.

Définition 11 (τ de Kendall pondéré)**.**

Soient
- u et v deux documents de degré de pertinence respectif $r(u)$ et $r(v)$,
- $sgn(.)$ la fonction signe telle que $sgn(x) = 1$ si $x > 0$, $sgn(x) = -1$ si $x < 0$ et $sgn(x) = 0$ si $x = 0$,
- w_{uv} le poids de la permutation entre le document u et le document v, par exemple $w_{uv} = r(u) - r(v)$,

Alors le τ de Kendall repondéré entre les listes π_r et π est défini de la façon suivante :

$$\tau(\pi_r, \pi) = \frac{\sum_{u<v} w_{uv}(1 + sgn((\pi_r(u) - \pi_r(v))(\pi(u) - \pi(v))))}{2\sum_{u<v} w_{uv}} \tag{1.18}$$

Il est important de noter que cette dernière formule est applicable dans les cas où il existe un ordre absolu entre les documents, c'est-à-dire quand il n'y a pas d'égalité. Dans le cas où deux documents présentent le même degré de pertinence, une inversion de ces documents n'affecte pas la qualité de la liste. Pourtant, elle aura une influence sur le τ, qui pourra être dégradé. Pour remédier à ce problème, l'idée est alors de calculer l'ensemble des τ correspondant aux permutations sans influence pour l'ordonnancement et de conserver la valeur maximale possible comme

mesure finale.

Ces mesures permettent d'évaluer la qualité d'ordonnancement d'un algorithme, en le comparant aux résultats obtenus avec une méthode de référence. Elles définissent la qualité d'ordonnancement à partir de la position des documents pertinents. Les mesures de corrélation des rangs évaluent l'adéquation entre une liste de référence et une liste candidate en considérant l'ensemble des documents de la liste, tandis que la MAP et le NDCG sont construits afin d'évaluer la qualité de l'ordonnancement jusqu'à une certaine position et se concentrent généralement sur les premiers documents retournés.

Conclusion

Dans ce chapitre, nous avons introduit des concepts fondamentaux de la RI. Nous avons dressé un état de l'art des modèles de recherche et d'ordonnancement existants. Nous avons détaillé leur protocole d'évaluation et les mesures utilisées. Plusieurs approches peuvent être envisagées pour améliorer la performance des méthodes d'ordonnancement, comme le développement de nouveaux modèles ou l'optimisation des paramètres des modèles existants. Une des stratégies les plus populaires au cours des dernières années consiste à utiliser des méthodes d'apprentissage afin de déterminer automatiquement la combinaison idéale des différents modèles existants et d'optimiser ainsi le classement des résultats de recherche. Cette approche est appelée *Learning to Rank* ou apprentissage d'ordonnancement. Les travaux effectués au cours de la thèse s'intéressent à ce domaine de recherche, que nous présentons dans le chapitre suivant.

Chapitre 2

L'apprentissage d'ordonnancement : principe et approches

Sommaire

Dans ce chapitre, nous introduisons le contexte de nos travaux : l'apprentissage d'ordonnancement, ou *learning to rank* en RI. Nous présentons dans un premier temps le principe de l'apprentissage d'ordonnancement. Dans un second temps, nous introduisons les approches et algorithmes proposés au cours de la dernière décennie. Dans un troisième temps, nous décrivons de façon détaillée les jeux de données spécifiquement créés pour l'évaluation de ces algorithmes. Enfin, nous comparons les différentes approches et algorithmes.

2.1 Principe de l'apprentissage d'ordonnancement

Les premières publications en apprentissage d'ordonnancement pour la RI remontent aux début des années 2000, mais le nombre de travaux et d'évènements (conférences, workshops, challenges) dédiés à ce domaine de la RI a réellement explosé à partir de 2005. L'apprentissage d'ordonnancement reste aujourd'hui un domaine de recherche majeur qui alimente de nombreuses thématiques et travaux [Liu 2011] : sélection de variables dédiées à l'ordonnancement, algorithmes d'ordonnancement en ligne, robustesse des algorithmes, ordonnancement à partir des

logs de requêtes, etc. Dans cette section, nous présentons le principe général de l'apprentissage d'ordonnancement ainsi que les concepts clés sur lesquels il repose.

L'apprentissage d'ordonnancement consiste à optimiser automatiquement à l'aide d'algorithmes d'apprentissage une fonction d'ordonnancement. Celle-ci est utilisée pour classer le plus pertinemment possible les documents vis-à-vis des requêtes des utilisateurs. L'apprentissage d'ordonnancement repose sur deux concepts : la représentation des couples requête-document dans l'espace des caractéristiques et l'utilisation d'une structure d'apprentissage en deux étapes.

2.1.1 Représentation dans l'espace des caractéristiques

Nous avons vu au chapitre 1 que les requêtes et documents sont représentés dans l'espace des termes qui les constituent. Les modèles de recherche et d'ordonnancement calculent alors la similarité entre la représentation de la requête et la représentation du document. En apprentissage d'ordonnancement, l'objectif est de combiner les scores ainsi obtenus, avec éventuellement d'autres caractéristiques propres à la requête ou au document, afin d'apprendre une fonction qui permettra d'ordonner les documents restitués pour une requête donnée.

Des couples requête-document sont représentés dans l'espace des caractéristiques (*feature-based approach*). Soient une requête q, un document d_j associé et d le nombre de caractéristiques utilisées par la fonction d'ordonnancement, alors le couple requête-document (q, d_j) est représenté par le vecteur $\mathbf{x} = \phi(q, d_j) \in (R)^d$ où $\phi(.,.)$ est un extracteur de caractéristiques tel que $\mathbf{x}_i = \phi_i(q, d_j)$ est la valeur de la caractéristique i pour le couple (q, d_j).

Les caractéristiques considérées sont principalement les scores de similarité entre la requête et le document obtenus par les modèles présentés à la section 1.1 et les scores d'importance des documents présentés à la section 1.1.2. D'autres caractéristiques propres à la requête, par exemple le nombre de termes de la requête, ou au document peuvent également être utilisées.

Ces vecteurs de caractéristiques sont par la suite utilisés par les algorithmes pour apprendre la fonction d'ordonnancement unique qui classe les résultats. Ce processus d'ordonnancement se décompose en deux phases que nous détaillons dans la section suivante.

2.1.2 Structure en deux étapes de l'apprentissage d'ordonnancement

Le processus d'ordonnancement se décompose en deux phases, qui correspondent aux différentes étapes de l'apprentissage automatique : l'apprentissage de la fonction et l'ordonnancement proprement dit sur de nouvelles requêtes de test. Ce principe est présenté dans la figure 2.1.

Dans l'étape d'apprentissage, un jeu de données composé des paires requête-document est considéré. Chaque paire (q_i, d_j) est représentée dans l'espace des caractéristiques par de le vecteur $\mathbf{x}_{i,j} \in \mathbb{R}^d$ tel que $\mathbf{x}_{i,j} = [x_{i,j}^{(1)} \dots x_{i,j}^{(d)}]$. Chaque paire

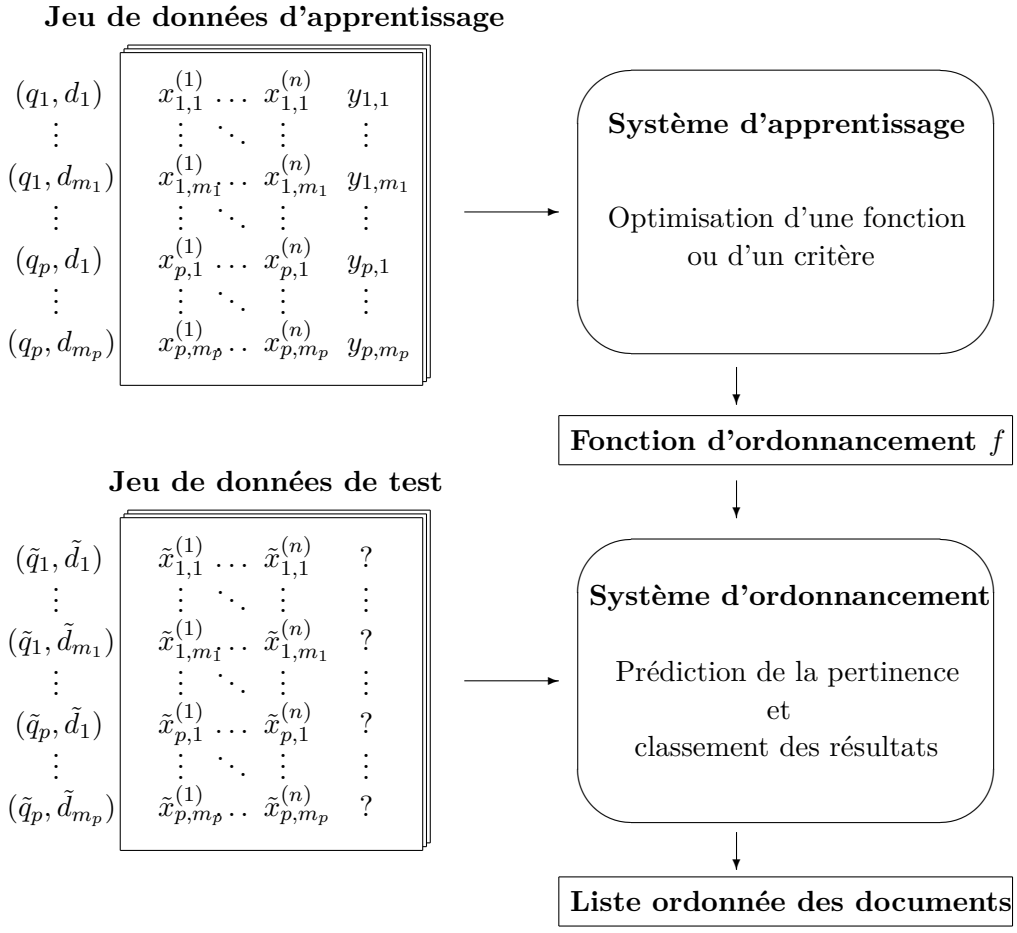


FIGURE 2.1 – Structure d'apprentissage d'ordonnancement en deux phases

est associée à un score de pertinence $y_{i,j}$ entre la requête et le document. Ce score peut être soit un nombre réel, soit un entier codant pour une classe de pertinence (par exemple, les entiers de 0 à 4 qui codent pour les classes allant de "non pertinent" à "parfaitement pertinent"). Les scores sont généralement déterminés manuellement par des experts humains, bien que des modèles de clics puissent être utilisés pour extraire des jugements de pertinence à partir des logs de connexion et des actions utilisateur (*cf.* chapitre 9). Le jeu de données est utilisé par le système d'apprentissage, c'est-à-dire l'algorithme, pour apprendre la fonction d'ordonnancement qui prédit les scores de pertinences à partir des caractéristiques fournies avec la plus petite erreur possible.

Dans la phase de test, la fonction apprise est utilisée par le système d'ordonnancement pour prédire les scores de pertinence de nouveaux couples requête-document qui n'ont pas été annotés. Le système d'ordonnancement retourne ainsi, pour chaque requête, la liste des documents ordonnés suivant la valeur de pertinence prédite.

Un grand nombre d'algorithmes ont été développés afin de résoudre au mieux le problème d'apprentissage. Dans la section suivante, nous présentons les différentes approches proposées dans la littérature pour résoudre le problème d'apprentissage et détaillons quelques algorithmes de référence.

2.2 Approches en apprentissage d'ordonnancement

Au cours de la dernière décennie, un grand nombre d'algorithmes ont été proposés pour l'apprentissage d'ordonnancement. Ils sont généralement regroupés sous trois grands types d'approches : par point (*pointwise*), par paire (*pairwise*) et par liste (*listwise*).

2.2.1 Approche par point

Dans l'approche par point, les documents sont considérés indépendamment en entrée du système d'apprentissage. Le jugement de pertinence peut être soit un score réel soit une classe de pertinence ordonnée (très pertinent > moyennement pertinent > peu pertinent) ou non ordonnée (non pertinent ou pertinent).

2.2.1.1 Score de pertinence réel

Dans ce cas, le problème d'apprentissage d'ordonnancement peut être ramené à un problème de régression linéaire. C'est notamment ce qu'ont proposé Cossock et Zhang [Cossock 2006] dans leur méthode intitulée *Subset ranking with Regression*. Considérant y_j le score de pertinence de référence et $\tilde{y}_j$ le score de pertinence prédit pour le document d_j, l'algorithme apprend la fonction qui minimise, pour chaque document d_j, la fonction de perte $L(y_j, \tilde{y}_j) = (y_j - \tilde{y}_j)^2$, qui traduit l'écart entre la valeur prédite et la valeur attendue.

Dans les deux autres cas, le problème d'apprentissage d'ordonnancement peut être ramené à un problème de discrimination. Les méthodes utilisées diffèrent sui-

vant que le jugement de pertinence correspond à une variable catégorielle non ordonnée ou ordonnée.

2.2.1.2 Classes de pertinence binaire ou multiple non ordonnées

Si la variable (jugement de pertinence) à prédire est binaire, alors des méthodes de discrimination connues en apprentissage automatique peuvent être utilisées pour résoudre le problème. De nombreux travaux se sont intéressés à l'apport de ces méthodes pour l'apprentissage d'ordonnancement.

Ainsi, Nallapati [Nallapati 2004] a évalué la performance des SVM linéaires dans ce contexte. Le principe général des SVM est de rechercher l'hyperplan qui sépare de façon optimale les documents non pertinents des documents pertinents dans l'espace des caractéristiques. Les fondements mathématiques de cette méthode sont présentés au chapitre 5.

Freund et Schapire ont proposé d'utiliser AdaBoost [Freund 1995], un algorithme basé sur le principe du boosting, pour apprendre la fonction d'ordonnancement. Il s'agit d'un algorithme itératif. L'objectif est de construire un ensemble de modèles dont la qualité de prédiction est légèrement meilleure que le hasard (le taux d'erreur est inférieur à 50 %) puis de les agréger afin d'avoir un modèle final meilleur que l'ensemble de ces classifieurs. A chaque itération, plusieurs classifieurs sont générés et celui qui présente la plus petite erreur de prédiction est sélectionné. Un vecteur de poids est alors défini pour l'ensemble des observations (ici des documents). Chaque document bien classé se trouve affecté d'un poids plus faible tandis que les documents mal classés sont affectés d'un poids plus élevé. De cette façon, le classifieur se concentrera sur ces derniers dans l'itération suivante, ce qui doit permettre de mieux les classer. Le modèle final est défini à partir de la moyenne pondérée de l'ensemble des modèles sélectionnés au cours des itérations successives.

Enfin, si le jugement de pertinence correspond à plus de deux classes, des méthodes de discrimination multiclasse peuvent être utilisées. Ainsi, Li *et al.* se sont intéressés à l'utilisation de méthodes de boosting [Li 2007]. Les méthodes d'apprentissage automatique autre que les SVM (boosting, réseaux de neurones, arbres de décision, etc) ne sont pas présentées en détail dans cette thèse. Le lecteur pourra se référer à l'ouvrage de référence de Hastie *et al.* [Hastie 2003].

2.2.1.3 Classes de pertinence ordonnées

Dans le cas où la variable à prédire est catégorielle et ordonnée, le problème d'apprentissage d'ordonnancement peut être ramené à un problème de régression ordinale. Contrairement aux méthodes d'apprentissage multiclasse, les méthodes de régression ordinale prennent en compte l'ordre relatif entre les classes pour apprendre le modèle. Soient N catégories de pertinence, l'objectif est de déterminer f la fonction d'ordonnancement qui retourne un score réel de pertinence et les $N-1$ seuils $s_1 < s_2 < \cdots < s_{N-1}$ qui déterminent les limites des classes et permettent d'associer la valeur de f à la catégorie de pertinence souhaitée. Ainsi, considérant

$\tilde{y}_j = f(x_j)$ le score de pertinence appris par la fonction f pour le document d_j, il suffit de comparer sa valeur aux différents seuils pour déterminer la catégorie prédite. Par exemple, si $\tilde{y}_j < s_1$ alors la classe correspondante est la plus petite classe de pertinence (par exemple non pertinent) tandis que si $\tilde{y}_j > s_{N-1}$, alors le document est associé à la plus grande classe de pertinence pour la requête considérée.

Plusieurs algorithmes ont été proposés pour résoudre ce type de problème. Le plus connu est probablement l'algorithme *PRanking* proposé par Crammer et Singer en 2001 [Crammer 2001]. Les seuils $s_1, \ldots, s_{N-1}$ et le modèle d'ordonnancement $\mathbf{w}$ sont déterminés itérativement. A chaque itération, un vecteur $\mathbf{x}_j$ représentant un couple requête-document dans l'espace des variables est aléatoirement sélectionné. L'algorithme prédit alors la valeur $\tilde{y}_j$ telle que $\tilde{y}_j = \arg\min_k\{\mathbf{w}^\top \mathbf{x}_j - s_k < 0\}$. Si $\tilde{y}_j \neq y_j$, alors l'algorithme a fait une erreur de prédiction. Les valeurs des seuils et le vecteur $\mathbf{w}$ sont alors modifiés simultanément pour corriger l'erreur. Le processus est itéré jusqu'à convergence.

L'approche par point est la plus simple à mettre en oeuvre. Néanmoins, elle est aussi considérée comme la moins performante, car elle ne prend pas en compte l'ordre relatif des documents [Liu 2011], contrairement à l'approche par paire, qui lui est généralement préférée.

2.2.2 Approche par paire

Dans l'approche par paire, des paires de documents sont considérées en entrée du système d'apprentissage. L'objectif n'est pas ici de déterminer le score de pertinence de chaque document vis-à-vis de la requête, mais quel document est plus pertinent qu'un autre. Le jugement de pertinence est ainsi exprimé comme une préférence d'un document par rapport à l'autre.

2.2.2.1 Principe général

Considérant une requête q, à chaque paire de documents est associée une préférence y_{ij} à valeur dans $\{-1, 1\}$. Si $y_{ij} = 1$, alors $\mathbf{x}_i$ est préféré à $\mathbf{x}_j$ pour la requête considérée. La relation de préférence est notée $\mathbf{x}_i \succ \mathbf{x}_j$. En pratique, cela signifie que le document initial d_i est plus pertinent que le document d_j pour la requête et qu'il devrait être classé au-dessus de ce dernier dans la liste de résultats. A l'inverse, si $y_{ij} = -1$, alors $\mathbf{x}_j$ est préféré à $\mathbf{x}_i$ ($\mathbf{x}_j \succ \mathbf{x}_i$) et le document d_j devrait être classé au-dessus de d_i dans la liste finale.

L'objectif est d'apprendre la fonction qui permet de discriminer au mieux les paires de documents et de leur affecter la classe correspondante. Le problème d'optimisation de la fonction d'ordonnancement est généralement ramené à un problème de discrimination binaire, les observations à discriminer étant des paires de documents et non plus les documents eux-mêmes. De nombreux algorithmes ont été proposés pour résoudre ce problème. Ils utilisent des méthodes d'apprentissage automatique connues (réseaux de neurones, boosting, arbre de décision, SVM, etc), qui ont été adaptées aux spécificités de l'ordonnancement de préférences. Dans cette

section, nous nous limitons à la présentation des algorithmes les plus connus et représentatifs, notamment RankNet, RankBoost et RankingSVM.

2.2.2.2 Algorithmes basés sur les réseaux de neurones

Plusieurs méthodes utilisant les réseaux de neurones ont été proposées, comme RankNet [Burges 2005] ou FRank [Tsai 2007]. RankNet est l'un des premiers algorithmes basé sur les réseaux de neurones. Il est réputé pour être utilisé sous une autre variante par le moteur de recherche commercial Bing.

Considérant une paire $(\mathbf{x}_i, \mathbf{x}_j)$ des documents associés à une requête q et représentés dans l'espace des caractéristiques, les auteurs définissent P^r_{ij}, la probabilité de référence que $\mathbf{x}_i$ soit préféré à $\mathbf{x}_j$. La probabilité P_{ij} que x_i soit préféré à x_j à prédire par l'algorithme est alors modélisée à partir de la fonction d'ordonnancement f de la façon suivante :

$$P_{ij}(f) = \frac{e^{f(\mathbf{x}_i)-f(\mathbf{x}_j)}}{1+e^{f(\mathbf{x}_i)-f(\mathbf{x}_j)}}$$

Un réseau de neurones est utilisé pour minimiser la fonction de perte $L(f, x_i, x_j, y_{ij})$ définie à partir de la probabilité de référence et de la probabilité modélisée, telle que :

$$L(f, \mathbf{x}_i, \mathbf{x}_j, y_{ij}) = -P^r_{ij} log(P_{ij}(f)) - (1 - P^r_{ij}) log(1 - P_{ij}(f))$$

et la fonction d'ordonnancement f est ainsi apprise par l'algorithme. Tsai *et al.* [Tsai 2007] ont avancé que la fonction de perte utilisée par RankNet présentait certains problèmes théoriques, comme la non existence d'un minimum global, pouvant dégrader fortement la qualité de l'ordonnancement. Ils ont proposé l'algorithme FRank pour résoudre ce problème. Leurs expérimentations ont montré que FRank était plus performant que RankNet sur un jeu de données issu d'un moteur de recherche commercial. Burges *et al.* ont proposé l'algorithme LambdaRank [Burges 2007], qui utilise également les réseaux de neurones et optimise une fonction de perte dépendant de la mesure de performance à maximiser. Les auteurs ont montré que cet algorithme était plus rapide que RankNet et fournissait un ordonnancement de meilleure qualité.

2.2.2.3 Algorithmes basés sur les approches de *boosting*

A notre connaissance, l'algorithme de référence en apprentissage d'ordonnancement utilisant les méthodes de boosting est RankBoost, proposé par Freund *et al.* [Freund 2003]. En pratique, il s'agit d'une adaptation de l'algorithme AdaBoost utilisée dans l'approche par point. La différence principale est que RankBoost considère les paires de documents et non plus les documents seuls. Les auteurs ont montré que l'algorithme est particulièrement performant sur le jeu de données de référence Digital Equipment Corporation [Freund 2003] et sur un jeu de données issu de [Cohen 1999]. D'autres travaux se sont intéressés à des algorithmes d'ordonnancement basés sur le boosting dans le cadre de données non étiquetées [Amini 2008].

2.2.2.4 Algorithmes basés sur les SVM

Enfin, d'autres méthodes d'apprentissage par paire particulièrement populaires sont basées sur les SVMs. Les deux algorithmes les plus connus sont RankSVM et RankSVM-Primal.

Considérant une paire de documents (d_i, d_j) associée à un requête q et représentée par le vecteur $\mathbf{x}_p = \mathbf{x}_i - \mathbf{x}_j \in \mathbb{R}^d$ dans l'espace des caractéristiques, ces méthodes apprennent la fonction d'ordonnancement linéaire optimale en résolvant le problème d'optimisation suivant :

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{p=1}^{P} l(\mathbf{w}^\top \mathbf{x}_p) \tag{2.1}$$

où $\mathbf{w}$ est le vecteur de poids représentant le modèle linéaire appris, $\mathbf{w}^\top$ son transposé, C est un paramètre permettant de contrôler les erreurs de prédiction et $l(.)$ est une fonction de perte telle que $l(\mathbf{w}^\top \mathbf{x}_p) = max(0, 1 - \mathbf{w}^\top \mathbf{x}_p)$ dans RankSVM et $l(\mathbf{w}^\top \mathbf{x}_p) = (\max(0, 1 - \mathbf{w}^\top \mathbf{x}_p))^2$ dans RankSVM-Primal.

Les deux algorithmes résolvent des problèmes légèrement différents. Par ailleurs, RankSVM-Primal considère directement la formulation 2.1 appelée formulation primale. RankSVM considère une autre formulation, dite duale, qui est présentée au chapitre 5. Ces différences entraînent des variations de la qualité d'ordonnancement entre les deux algorithmes. RankSVM-Primal peut être légèrement meilleur que RankSVM sur les jeux de données de référence. Par ailleurs, lors de la publication de RankSVM-Primal en 2010 [Chapelle 2010], celui-ci était bien plus rapide que son homologue dual. Des modifications ont depuis été apportées à l'algorithme RankSVM et les versions actuellement disponibles en ligne [1] présentent des temps d'exécution comparables sur les jeux de données de référence. D'autres algorithmes ont été proposés pour remédier à certaines limitations de RankSVM et RankSVM-Primal. Ainsi, l'algorithme IR-SVM introduit une normalisation dans la fonction de perte pour prendre en compte le fait que toutes les requêtes ne possèdent pas le même nombre de documents, ce qui peut dégrader la qualité d'ordonnancement [Cao 2006]. Sculley [Sculley 2009] propose une version stochastique de RankSVM, qui a pour but d'accélérer les temps d'exécution afin de pouvoir traiter des jeux de données très volumineux.

Les approches par paire permettent de prendre en compte des relations d'ordre entre documents deux à deux, mais pas sur la globalité de la liste. Des travaux [Cao 2007] [Xia 2008] ont proposé de considérer en entrée des systèmes d'apprentissage non pas des paires de documents, mais la liste complète de documents. Cette approche, appelée approche par liste (*listwise*), ainsi que les algorithmes représentatifs, sont présentés dans la section suivante.

1. www.joachims.org

2.2.3 Approche par liste

Dans l'approche par liste, à chaque requête q en entrée du système d'apprentissage est associée une liste de résultats ordonnés suivant leur pertinence. Cette liste est considérée comme la référence que les algorithmes cherchent à prédire. Deux grands types de méthodes ont été proposés dans ce but. Elles diffèrent sur les fonctions de perte utilisées pour résoudre le problème d'ordonnancement. Les méthodes de la première catégorie considèrent des fonctions de perte définies à partir de mesures de RI, contrairement aux méthodes de la deuxième catégorie qui utilisent des fonctions de perte indépendantes des mesures de RI.

2.2.3.1 Méthodes utilisant des fonctions de perte basées sur les mesures de RI

Ces méthodes reposent sur le principe suivant : la performance des algorithmes étant évaluée à partir de certaines mesures de RI, il est plus logique et efficace d'optimiser celles-ci directement dans l'algorithme d'ordonnancement plutôt qu'indirectement via une autre fonction de perte. La principale difficulté de ce type d'approches réside dans le fait que les mesures de performance utilisées, en l'occurrence le NDCG et la MAP, ne sont pas différentiables [Liu 2011]. Il est donc difficile de les optimiser directement.

Pour remédier à ce problème, certains travaux ont proposé de considérer des approximations différentiables des mesures au lieu des mesures elles-mêmes. Considérant une certaine mesure m, ces méthodes cherchent à minimiser une approximation de la quantité $1-m$. A notre connaissance, SoftRank [Taylor 2008] a été le premier algorithme proposé pour l'apprentissage d'ordonnancement par liste implémentant une telle approximation. L'idée sous-jacente de cet algorithme est de considérer le score de pertinence des documents comme une variable aléatoire. Plusieurs classements avec des probabilités distinctes d'apparition peuvent alors être générés. Les valeurs correspondantes de la mesure m peuvent être calculées. L'espérance de cette mesure, notée $\mathbb{E}(m)$, est utilisée comme approximation de m. L'algorithme optimise alors la quantité $1-\mathbb{E}(m)$. SoftRank a été initialement proposé pour optimiser le NDCG, l'optimisation de la MAP étant présentée dans [Yilmaz 2010]. D'autres algorithmes ont proposé l'utilisation de fonctions d'utilité [Zoeter 2008]. Les algorithmes ApproximateRank [Qin 2010] et SmoothRank réécrivent les mesures en faisant intervenir des fonctions indicatrices non différentiables et les approchent à l'aide de fonctions exponentielles ; l'approximation obtenue alors est différentiable. Les deux approches diffèrent sur l'approximation choisie et la méthode d'optimisation utilisée.

D'autres travaux ont proposé de résoudre le problème en optimisant non pasune approximation de la mesure m, mais la borne supérieure de la quantité $1-m$ qui est différentiable. L'algorithme le plus connu basé sur cette méthode est probablement SVM-MAP, qui définit une borne supérieure de la MAP en utilisant la somme des termes d'erreur d'ordonnancement [Yue 2007].

D'autres méthodes résolvent le problème d'ordonnancement en optimisant directement les mesures de RI. Des techniques spécifiquement conçues pour l'optimisation de mesures non différentiables sont utilisées par les algorithmes. Parmi ces derniers, nous trouvons notamment AdaRank [Xu 2007], basé sur l'algorithme AdaBoost et permettant l'optimisation de la MAP et du NDCG respectivement et RankGP [Yeh 2007]. Ce dernier utilise un algorithme génétique pour résoudre le problème d'optimisation. Selon [Liu 2011], si ces méthodes présentent l'avantage d'optimiser directement les mesures considérées, elles ne garantissent pas que la solution optimale a bien été atteinte.

D'autres méthodes de l'approche par liste existent et utilisent des fonctions de perte ne dépendant pas des mesures à optimiser. Deux exemples principaux sont présentés dans la section suivante.

2.2.3.2 Méthodes utilisant des fonctions de perte indépendantes des mesures de RI

Dans cette section, nous nous limitons à la présentation des algorithmes ListNet [Cao 2007] et ListMLE [Xia 2008], qui sont considérés comme les algorithmes de référence, bien que d'autres algorithmes [Huang 2008, Volkovs 2009] existent. Contrairement aux méthodes de la catégorie précédente, l'objectif ici n'est pas d'utiliser une fonction de perte construite à partir de la mesure de RI considérée, mais une fonction de perte qui mesure le nombre de permutations entre la liste de référence et la liste apprise.

A notre connaissance, ListNet est le premier algorithme proposé dans le cadre de cette approche. Les auteurs définissent dans un premier temps la probabilité d'obtention de la liste permutée π à partir de la liste de référence π_r, notée $P_y(\pi)$, de la façon suivante :

$$P_y(\pi) = \prod_{j=1}^{n} \frac{\phi(y_{\pi(d_j)})}{\sum_{k=j}^{n} \phi(y_{\pi(d_k)})}$$

où ϕ est une fonction croissante strictement positive et $y_{\pi(d_j)}$ est le score du document d_j dans la liste π. Cette probabilité pourrait être utilisée directement par la fonction de perte ; néanmoins, le nombre de permutations possibles entre deux listes étant égal à $n!$, la complexité du problème serait trop grande. Les auteurs proposent de considérer plutôt la probabilité de rang 1. Il s'agit de la probabilité qu'un document d_j soit classé en premier sachant les scores de tous les objets. Elle est définie pour la liste π de la façon suivante :

$$P_\pi(d_j) = \prod_{j=1}^{n} \frac{\phi(y_{d_j})}{\sum_{k=j}^{n} \phi(y_{d_k})}$$

Considérant deux listes π_1 et π_2 associées à une requête q, l'algorithme ListNet minimise alors la fonction de perte définie à partie de l'entropie croisée de la façon

suivante :

$$L(\pi_1, \pi_2) = \sum_{j=1}^{n} P_{\pi_1}(j) \log(P_{\pi_2(j)})$$

à l'aide d'une approche par réseaux de neurones. Les auteurs montrent que ListNet donne de meilleurs résultats que d'autres algorithmes de l'approche par paire sur trois jeux de données de référence, donc TREC2003 [Craswell 2003], Ohsumed [Hersh 1994] et un jeu de données commercial.

L'inconvénient majeur de ListNet est de présenter des temps d'exécution élevés. D'autres algorithmes utilisent des fonctions de perte différentes. Qin *et al.* ont proposé d'utiliser le cosinus de l'angle entre les vecteurs de scores des deux listes à comparer [Qin 2008]. L'algorithme correspondant est appelé RankCosine et a un taux de convergence plus rapide que ListNet. Xu *et al.* ont introduit l'algorithme ListMLE utilisant une fonction de perte basée sur la vraisemblance. Les deux algorithmes présentent les mêmes taux théoriques de convergence (donc a priori les mêmes vitesses d'exécution) mais la fonction de perte utilisée par ListMLE présente l'avantage d'être convexe [Xia 2008].

L'ensemble de ces algorithmes sont évalués en utilisant la méthodologie de type Cranfield présentée au chapitre précédent. La performance des différents algorithmes en matière d'ordonnancement est généralement comparée via la MAP et le NDCG. Plusieurs jeux de données de référence ont été construits et mis à disposition de la communauté afin de pouvoir comparer les approches et algorithmes sur les mêmes jeux de données. Dans la section suivante, nous présentons les principaux jeux de données de référence et leurs caractéristiques.

2.3 Collections de référence

Afin de comparer plus facilement les performances des différentes méthodes, des collections spécifiques à l'apprentissage d'ordonnancement ont été mises à disposition de la communauté. Dans cette partie, nous présentons les trois collections les plus connues en apprentissage d'ordonnancement : les collections LEarning TO Rank (LETOR), Yahoo ! Learning to Rank Challenge et Microsoft Learning to Rank. Nous détaillons plus particulièrement les collections LETOR, qui sont les plus utilisées dans la littérature et sur lesquelles nous avons mené nos expérimentations.

2.3.1 Les collections LETOR

Les premières collections LETOR ont été mises à disposition de la communauté scientifique en 2007 par Microsoft Research. Il s'agissait alors des toutes premières collections dédiées à l'évaluation des algorithmes d'apprentissage d'ordonnancement. Elles présentent de nombreux avantages : accessibilité, format de fichiers standardisé, bonne documentation des jeux de données et caractéristiques utilisés et publication en ligne des résultats des algorithmes de référence. Ceux-

ci expliquent qu'elles restent encore aujourd'hui les collections les plus populaires, malgré l'existence de jeux de données plus récents. Dans cette section, nous nous concentrons sur les collections LETOR 3.0 et 4.0, les plus récentes et les plus utilisées dans la littérature.

2.3.1.1 Description générale

Les collections LETOR 3.0 et 4.0 ont été mises en ligne par Microsoft Research Asia sur un site Internet dédié [2] en décembre 2008 et juillet 2009 respectivement. L'objectif déclaré était de faciliter la recherche en apprentissage d'ordonnancement. Les deux collections réunies regroupent au total neuf jeux de données, la plupart issus des campagnes TREC [3].

LETOR 3.0 est ainsi constitué du jeu de données *Ohsumed* et du corpus *Gov* tandis que LETOR 4.0 est constituée du seul corpus *Gov2*. Dans la suite de cette section, nous présentons en détail la composition de chacun de ces corpus ainsi que les caractéristiques extraites pour l'apprentissage d'ordonnancement. Nous présentons également le format de fichier et le mode de partitionnement des jeux de données, communs à l'ensemble des collections.

TABLE 2.1 – Description des jeux de données associés aux collections LETOR 3.0 et 4.0

Jeux de	**Nombre de**				**Nombre moyen par requête de**	
données	requêtes	paires	préférences	variables	documents	documents pertinents
OHSUMED	106	16140	582588	45	152	46
HP2003	150	157606	181302	64	984	1
HP2004	75	74409	80306	64	985	1
NP2003	150	148657	150495	64	991	1
NP2004	75	73834	75747	64	992	1
TD2003	50	49058	358562	64	981	8
TD2004	75	74146	1079810	64	989	15
MQ2007	1692	69623	404467	46	41	11
MQ2008	784	15211	80925	46	19	4

2.3.1.2 Le jeu de données Ohsumed

Le jeu de données Ohsumed est issu de la base de données documentaires MEDLINE, spécialisée dans les publications scientifiques du domaine médical et paramédical. Il est constitué de 348 566 enregistrements extraits de 270 journaux médicaux parus entre 1987 et 1991. Chaque enregistrement, ou document, est lui-même composé d'un titre, d'un résumé, de la liste des auteurs ou encore de termes d'indexation. Un sous-ensemble de 106 requêtes soumises sur ce corpus est utilisé, chaque

2. http ://research.microsoft.com/en-us/um/beijing/projects/letor//
3. http ://trec.nist.gov/

requête étant associée à des documents pertinents et non pertinents. Les caractéristiques exactes du jeu de données (nombre moyen de documents par requête, nombre moyen de documents pertinents par requête, ...) sont indiquées dans le tableau 2.1. Pour chaque couple requête-document, 45 caractéristiques ont été extraites, dont la liste complète est fournie dans le tableau 2.2. Il s'agit majoritairement des scores de pertinence calculés par les modèles présentés au chapitre 1. Ceux-ci sont calculés entre la requête et le titre du document, entre la requête et le résumé du document et enfin, entre la requête et la réunion du titre et du résumé. Parmi les 45 caractéristiques extraites, six présentent des valeurs nulles pour l'ensemble des couples requête-document et ne sont donc pas discriminantes. Elles ne seront pas considérées dans la suite de ces travaux.

TABLE 2.2 – Caractéristiques extraites pour le jeu de données Ohsumed [Liu 2011]

ID	Description
1	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le titre
2	$\sum_{t_i \in q \bigcap d} log(TF(t_i, d) + 1)$ sur le titre
3	$\sum_{t_i \in q \bigcap d} \frac{TF(t_i,d)}{length(d)}$ sur le titre
4	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} + 1)$ sur le titre
5	$\sum_{t_i \in q log(\lvert C \rvert . IDF(t_i))}$ sur le titre
6	$\sum_{t_i \in q log(log(\lvert C \rvert . IDF(t_i)))}$ sur le titre
7	$\sum_{t_i \in q} log(\frac{\lvert C \rvert}{TF(t_i,C)} + 1)$ sur le titre
8	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} . log(\lvert C \rvert) . IDF(t_i) + 1)$ sur le titre
9	$\sum_{t_i \in q \bigcap d} TF(t_i, d) . log(\lvert C \rvert . IDF(t_i))$ sur le titre
10	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} \cdot \frac{\lvert C \rvert}{TF(t_i,C)} + 1)$ sur le titre
11	$BM25$ sur le titre
12	$log(BM25)$ sur le titre
13	LMIR avec lissage Dirichlet sur le titre
14	LMIR avec lissage Jelinek-Mercer sur le titre
15	LMIR avec lissage Absolute Discounting sur le titre
16	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le résumé
17	$\sum_{t_i \in q \bigcap d} log(TF(t_i, d) + 1)$ sur le résumé
18	$\sum_{t_i \in q \bigcap d} \frac{TF(t_i,d)}{length(d)}$ sur le résumé
19	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} + 1)$ sur le résumé
20	$\sum_{t_i \in q log(\lvert C \rvert . IDF(t_i))}$ sur le résumé
21	$\sum_{t_i \in q log(log(\lvert C \rvert . IDF(t_i)))}$ sur le résumé
22	$\sum_{t_i \in q} log(\frac{\lvert C \rvert}{TF(t_i,C)} + 1)$ sur le résumé
23	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} . log(\lvert C \rvert) . IDF(t_i) + 1)$ sur le résumé
24	$\sum_{t_i \in q \bigcap d} TF(t_i, d) . log(\lvert C \rvert . IDF(t_i))$ sur le résumé
25	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} \cdot \frac{\lvert C \rvert}{TF(t_i,C)} + 1)$ sur le résumé

ID	**Description** (Suite table 2.2)
26	$BM25$ sur le résumé
27	$log(BM25)$ sur le résumé
28	LMIR avec lissage Dirichlet sur le résumé
29	LMIR avec lissage Jelinek-Mercer sur le résumé
30	LMIR avec lissage Absolute Discounting sur le résumé
31	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le titre et le résumé
32	$\sum_{t_i \in q \bigcap d} log(TF(t_i, d) + 1)$ sur le titre et le résumé
33	$\sum_{t_i \in q \bigcap d} \frac{TF(t_i,d)}{length(d)}$ sur le titre et le résumé
34	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} + 1)$ sur le titre et le résumé
35	$\sum_{t_i \in q log(\|C\|.IDF(t_i))}$ sur le titre et le résumé
36	$\sum_{t_i \in q log(log(\|C\|.IDF(t_i)))}$ sur le titre et le résumé
37	$\sum_{t_i \in q} log(\frac{\|C\|}{TF(t_i,C)} + 1)$ sur le titre et le résumé
38	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)}.log(\|C\|).IDF(t_i) + 1)$ sur le titre et le résumé
39	$\sum_{t_i \in q \bigcap d} TF(t_i, d).log(\|C\|.IDF(t_i))$ sur le titre et le résumé
40	$\sum_{t_i \in q \bigcap d} log(\frac{TF(t_i,d)}{length(d)} \cdot \frac{\|C\|}{TF(t_i,C)} + 1)$ sur le titre et le résumé
41	$BM25$ sur le titre et le résumé
42	$log(BM25)$ sur le titre et le résumé
43	LMIR avec lissage Dirichlet sur le titre et le résumé
44	LMIR avec lissage Jelinek-Mercer sur le titre et le résumé
45	LMIR avec lissage Absolute Discounting sur le titre et le résumé

2.3.1.3 Le corpus Gov et les six jeux de données associés

Le corpus *Gov* est constitué de 1 053 110 documents web extraits en janvier 2002 à partir du domaine .gov. Il est constitué de six jeux de données, dont trois issus de la campagne TREC 2003 et trois de la campagne TREC 2004. Chacun des jeux de données correspond à l'une des sous-tâches de la tâche *Web Track*, soient les tâches *Homepage Finding*, *Named Page Finding* et *Topic Distillation*, décrites ci-après.

L'objectif de la tâche *Homepage Finding* est de trouver la page d'accueil correspondant à la requête. La tâche *Named Page Finding* vise à restituer la page Internet dont le nom est exactement égal à la requête soumise. Enfin, l'objectif de la tâche *Topic Distillation* est de déterminer un ensemble de sites Internet traitant principalement du thème de la requête. Les jeux de données extraits sont nommés suivant la tâche et la campagne TREC considérées, soient respectivement, HP2003, HP2004, NP2003, NP2004, TD2003 et TD2004. Le nombre de requêtes constituant chaque jeu de données est indiqué dans le tableau 2.1, ainsi que le nombre de paires requête-document et les nombres moyens de documents et de documents pertinents par requête. Au total, 64 variables ont été extraites, correspondant soit à des caractéristiques de la requête, soit à des scores de similarité entre requête et document. Les similarités sont elles-mêmes calculées entre la requête et respectivement, le titre, le corps, l'ancre et l'URL du document. La liste détaillée des caractéristiques est présentée dans le tableau 2.3.

TABLE 2.3 – Caractéristiques extraites sur le corpus Gov [Liu 2011]

ID	Description
1	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le corps du document
2	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur l'ancre
3	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le titre
4	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur l'URL
5	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le document complet
6	$\sum_{t_i \in q} IDF(t_i)$ sur le corps du document
7	$\sum_{t_i \in q} IDF(t_i)$ sur l'ancre
8	$\sum_{t_i \in q} IDF(t_i)$ sur le titre
9	$\sum_{t_i \in q} IDF(t_i)$ sur l'URL
10	$\sum_{t_i \in q} IDF(t_i)$ sur le document complet
11	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur le corps du document
12	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur l'ancre
13	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur le titre
14	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur l'URL
15	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur le document complet
16	Nombre de termes dans le corps du document
17	Nombre de termes dans l'ancre
18	Nombre de termes dans le titre
19	Nombre de termes dans l'URL
20	Nombre de termes dans le document complet
21	$BM25$ sur le corps du document
22	$BM25$ sur l'ancre
23	$BM25$ sur le titre
24	$BM25$ sur l'URL
25	$BM25$ sur le document complet
26	LMIR avec lissage Absolute Discounting sur le corps du document
27	LMIR avec lissage Absolute Discounting sur l'ancre
28	LMIR avec lissage Absolute Discounting sur le titre
29	LMIR avec lissage Absolute Discounting sur l'URL
30	LMIR avec lissage Absolute Discounting sur le document complet
31	LMIR avec lissage Dirichlet sur le corps du document
32	LMIR avec lissage Dirichlet sur l'ancre
33	LMIR avec lissage Dirichlet sur le titre
34	LMIR avec lissage Dirichlet sur l'URL
35	LMIR avec lissage Dirichlet sur le document complet
36	LMIR avec lissage Jelinek-Mercer sur le corps du document
37	LMIR avec lissage Jelinek-Mercer sur l'ancre

ID	Description (Suite table 2.3)
38	LMIR avec lissage Jelinek-Mercer sur le titre
39	LMIR avec lissage Jelinek-Mercer sur l'URL
40	LMIR avec lissage Jelinek-Mercer sur le document complet
41	Sitemap based term propagation
42	Sitemap based score propagation
43	Hyperlink based score propagation : weighted in-link
44	Hyperlink based score propagation : weighted out-link
45	Hyperlink based score propagation : uniform out-link
46	Hyperlink based feature propagation : weighted in-link
47	Hyperlink based feature propagation : weighted out-link
48	Hyperlink based feature propagation : uniform out-link
49	Autorité HITS
50	Hub HITS
51	PageRank
52	HostRank
53	PageRank thématique
54	Autorité HITS thématique
55	Hub HITS thématique
56	Nombre de liens entrants
57	Nombre de liens sortants
58	Nombre de slash dans l'URL du document
59	Taille de l'URL
60	Nombre de pages filles du document
61	$BM25$ sur le titre extrait
62	LMIR avec lissage Absolute Discounting sur le titre extrait
63	LMIR avec lissage Dirchlet sur le titre extrait
64	LMIR avec lissage Jelinek-Mercer sur le titre extrait

2.3.1.4 Le corpus Gov2 et les deux jeux de données associés

Le corpus *Gov2* regroupe environ 25 millions de documents (pages web) extraits début 2004 à partir du domaine .gov. Deux jeux de données ont été construits en utilisant ce corpus. Les requêtes considérées sont issues des tâches *Million Query Track* de TREC 2007 et 2008, dont l'un des objectifs principaux est d'effectuer des recherches *ad-hoc* sur une collection de très grande dimension. Les jeux de données, nommés MQ2007 et MQ2008, contiennent respectivement 1692 et 784 requêtes. Leur composition est détaillée dans le tableau 2.1 page 38. Pour chaque couple requête-document, 46 caractéristiques, représentant soit les scores des modèles de recherche soit des propriétés des documents, sont extraites. Elles sont détaillées dans le tableau 2.4.

Table 2.4 – Caractéristiques extraites sur le corpus Gov2 [Liu 2011]

ID	Description
1	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le corps du document
2	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur l'ancre
3	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le titre
4	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur l'URL

ID	Description (Suite table 2.4)
5	$\sum_{t_i \in q \bigcap d} TF(t_i, d)$ sur le document complet
6	$\sum_{t_i \in q} IDF(t_i)$ sur le corps du document
7	$\sum_{t_i \in q} IDF(t_i)$ sur l'ancre
8	$\sum_{t_i \in q} IDF(t_i)$ sur le titre
9	$\sum_{t_i \in q} IDF(t_i)$ sur l'URL
10	$\sum_{t_i \in q} IDF(t_i)$ sur le document complet
11	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur le corps du document
12	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur l'ancre
13	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur le titre
14	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur l'URL
15	$\sum_{t_i \in q \bigcap d} TF(t_i, d).IDF(t_i)$ sur le document complet
16	Nombre de termes dans le corps du document
17	Nombre de termes dans l'ancre
18	Nombre de termes dans le titre
19	Nombre de termes dans l'URL
20	Nombre de termes dans le document complet
21	$BM25$ sur le corps du document
22	$BM25$ sur l'ancre
23	$BM25$ sur le titre
24	$BM25$ sur l'URL
25	$BM25$ sur le document complet
26	LMIR avec lissage Absolute Discounting sur le corps du document
27	LMIR avec lissage Absolute Discounting sur l'ancre
28	LMIR avec lissage Absolute Discounting sur le titre
29	LMIR avec lissage Absolute Discounting sur l'URL
30	LMIR avec lissage Absolute Discounting sur le document complet
31	LMIR avec lissage Dirichlet sur le corps du document
32	LMIR avec lissage Dirichlet sur l'ancre
33	LMIR avec lissage Dirichlet sur le titre
34	LMIR avec lissage Dirichlet sur l'URL
35	LMIR avec lissage Dirichlet sur le document complet
36	LMIR avec lissage Jelinek-Mercer sur le corps du document
37	LMIR avec lissage Jelinek-Mercer sur l'ancre
38	LMIR avec lissage Jelinek-Mercer sur le titre
39	LMIR avec lissage Jelinek-Mercer sur l'URL
40	LMIR avec lissage Jelinek-Mercer sur le document complet
41	PageRank
42	Nombre de liens entrants
43	Nombre de liens sortants

ID	Description (Suite table 2.4)
44	Nombre de slash dans l'URL du document
45	Nombre de termes de l'URL
46	Nombre de pages filles

2.3.1.5 Format de fichier et partitionnement des données

Les collections partagent le même format de fichiers, ce qui fournit une norme de lecture pour l'ensemble des travaux en apprentissage d'ordonnancement. Les jeux de données sont disponibles au format texte. Chaque ligne du fichier représente une paire requête-document structurée comme indiqué dans l'extrait 2.1. Le premier élément d'une ligne correspond au jugement de pertinence associé à la paire requête-document, traduit par un nombre entier. Le second élément correspond à l'identifiant de la requête, précédé de la balise `qid:`. Le reste de la ligne correspond aux valeurs des variables caractérisant la paire requête-document. Chaque valeur est précédée d'une balise correspondant à l'identifiant numérique de la caractéristique considérée. Le dernier élément est un commentaire contenant l'identifiant du document.

Extrait 2.1 – Structure d'une ligne du fichier.

```
<relevance> qid:<qid> <feature1>:<value> <feature2>:<value> ... <featureN>:<value> #<Docid>

avec
<relevance> : score de pertinence
<qid> : identifiant de la requete
<feature1> ... <featureN> : entier indiquant la variable
<value> : valeur variable
<Docid> : identifiant du document
```

Chaque jeu de données est décomposé en sous-échantillons de façon à effectuer une validation croisée de type 5-Folds. Le partitionnement est détaillé dans le tableau 2.5. Cinq sous-échantillons S_1, S_2, S_3, S_4, S_5 de tailles sensiblement équivalentes sont créés en tirant aléatoirement et sans remise des paires requête-document. Pour chacune des cinq répétitions, ou runs, de la validation croisée, un des sous-échantillons est choisi comme jeu de test, un comme jeu de validation et les trois restants sont fusionnés pour former l'échantillon d'apprentissage. Ce partitionnement est identique pour l'ensemble des neuf jeux de données présentés.

Table 2.5 – Partitionnement des jeux de données pour la validation croisée [Liu 2011]

Run	Apprentissage	Validation	Test
1	$\{S_1, S_2, S_3\}$	S_4	S_5
2	$\{S_2, S_3, S_4\}$	S_5	S_1
3	$\{S_3, S_4, S_5\}$	S_1	S_2
4	$\{S_4, S_5, S_1\}$	S_2	S_3
5	$\{S_5, S_1, S_2\}$	S_3	S_4

Enfin, les caractéristiques des différents jeux de données ont toutes subi une normalisation pour les ramener dans l'intervalle $[0, 1]$. Cette normalisation est ef-

fectuée au niveau de la requête. Considérant q une requête du jeu de données, N_q le nombre de documents associés à la requête q et $x_j^{(q)}$ la valeur de la caractéristique x pour la requête q et le document j, la valeur normalisée $\tilde{x}_j^{(q)}$ de $x_j^{(q)}$ est calculée de la façon suivante :

$$\tilde{x}_j^{(q)} = \frac{x_j^{(q)} - min\{x_k^{(q)}, k = 1, \ldots, N_{(q)}\}}{max\{x_k^{(q)}, k = 1, \ldots, N_{(q)}\} - min\{x_k^{(q)}, k = 1, \ldots, N_{(q)}\}} \tag{2.2}$$

Pour conclure cette section, les collections LETOR présentent l'avantage d'être libres d'accès, d'offrir un cadre expérimental reproductible (format de fichier de données standardisé, partitionnement des données pré-établi, ...) et surtout, de fournir les résultats en matière de MAP et NDCG pour les algorithmes de référence en apprentissage d'ordonnancement, ce qui facilite l'évaluation. Bien qu'elles soient utilisées pour expérimentation dans la grande majorité des travaux actuels en apprentissage d'ordonnancement, leur taille relativement restreinte (nombre de requêtes, nombre de documents par requête) peut constituer une limite. D'autres collections de plus grande volumétrie ont ainsi été mises à disposition par des moteurs commerciaux. Nous présentons deux de ces collections dans la section suivante.

2.3.2 Les autres collections dédiées à l'apprentissage d'ordonnancement

Dans cette section, nous présentons les deux collections dédiées au learning to rank les plus connues après les collections LETOR. De plus grande volumétrie que ces dernières, elles ont été respectivement mises à disposition de la communauté par les sociétés Yahoo ! et Microsoft. Nous présentons rapidement le contexte dans lequel elles ont été proposées et nous les décrivons.

2.3.2.1 Yahoo ! Learning to Rank Challenge

Deux jeux de données dédiés au learning to rank ont été mis à disposition par la société Yahoo ! dans le cadre du Yahoo ! Learning to Rank Challenge [Chapelle 2011a] qui s'est déroulé de mars à mai 2010, en association avec la conférence internationale "ICML" (*International Conference on Machine Learning)*. Les deux jeux de données constituent des extraits de la base d'apprentissage du moteur de recherche Yahoo ! et correspondent à deux pays différents, les Etats-Unis d'Amérique (Set1) et un pays d'Asie non précisé (Set2). Ils sont disponibles sur demande auprès de la librairie en ligne Yahoo !WebScope [4].

Le format des jeux de données est similaire à celui utilisé par les collections LETOR. Chaque ligne du jeu de données correspond à une paire requête-URL à laquelle est associé un degré de pertinence allant de 0 (*bad* ou non pertinent) à

4. http ://webscope.sandbox.yahoo.com/

TABLE 2.6 – Description of Yahoo ! Learning to Rank Datasets [Chapelle 2011a]

	SET1			SET2		
	Apprentissage	Validation	Test	Apprentissage	Validation	Test
Requêtes	19 944	2994	6983	1266	1266	3798
Documents	473 134	71 083	165 660	34 815	34 881	103 174
Variables		519			596	

4 (*excellent* ou parfaitement pertinent). Chaque jeu de données est composé d'un échantillon d'apprentissage, un échantillon de validation et un échantillon de test. Un seul run est disponible, contrairement aux collections LETOR qui en proposent cinq. Par ailleurs, les valeurs des mesures d'évaluation pour les algorithmes de référence ne sont pas fournies, ce qui oblige les chercheurs à implémenter ces algorithmes pour comparaison à l'état de l'art.

Pour des raisons de confidentialité, aucune description des données, qu'il s'agisse des requêtes, des documents ou des caractéristiques utilisés dans le jeu de données, n'a été divulguée. Seules les informations quantitatives sur le nombre de requêtes, de documents et de caractéristiques sont connues. Les deux collections regroupent environ 36 000 requêtes, 883 000 documents et de 500 à 600 caractéristiques. La répartition exacte des requêtes, documents et caractéristiques est indiquée dans le tableau 2.6. Si cette politique de confidentialité est légitime pour un moteur commercial, l'absence d'informations qualitatives sur les requêtes, documents et caractéristiques limitent l'utilisation des jeux de données dans des perspectives de recherche. Plus particulièrement, l'impact des requêtes sur la qualité de l'ordonnancement et l'importance respective de chaque caractéristique pour l'ordonnancement ne peuvent être analysés.

2.3.2.2 Microsoft Learning to Rank

Deux collections de grande dimension ont également été publiées par Microsoft Research en mai 2010 pour la recherche en learning to rank. Elles contiennent des requêtes et documents issus du moteur de recherche commercial Bing. De façon similaire aux jeux de données du "Yahoo ! Learning to Rank Challenge", aucune information qualitative n'est disponible sur les requêtes et les documents. A l'inverse, une liste des caractéristiques [5] est disponible, pour la simple raison que les caractéristiques fournies ne sont pas celles utilisées par le moteur de recherche, mais sont issues de la littérature.

Le premier jeu de données contient les paires requête-document annotées pour 31 531 requêtes et 3 771 126 documents, il est nommé MSLR-WEB30K. Le second jeu de données, nommé MSLR-WEB10K, est un sous-ensemble du premier. Il contient des paires sélectionnées aléatoirement et correspondant à 10 000 requêtes et

5. Nous ne reproduisons pas la liste ici. Elle est disponible sur le site Internet de Microsoft Research à l'adresse suivante : http ://research.microsoft.com/en-us/projects/mslr/

1 200 193 documents. Au total, 136 caractéristiques sont extraites. Chaque paire est annotée avec une classe de pertinence allant de 0 (non pertinent) à 4 (parfaitement pertinent).

Le format de fichier utilisé et le partitionnement des données sont les mêmes que ceux des collections LETOR. Néanmoins, les valeurs de MAP et NDCG@k, $k = 1, \ldots, 10$ des algorithmes de référence ne sont pas disponibles pour comparaison.

La plupart des algorithmes présentés à la section 2.2 ont fait l'objet d'une analyse comparative sur les jeux de données LETOR [Liu 2011]. Dans la section suivante, nous détaillons les avantages et inconvénients des différentes approches ainsi que les résultats d'analyses comparatives présentés dans plusieurs travaux.

2.4 Comparaison des approches

Les approches par point et par paire ont été les premières proposées, dès le début des années 2000, tandis que les premiers travaux traitant de l'approche par liste datent de 2007. Les approches par point et par paire sont généralement considérées comme plus aisées à mettre en place. En effet, dans le cas des approches par point, les algorithmes utilisent directement des méthodes de discrimination existantes. Les approches par paire permettent également d'utiliser des algorithmes existants de discrimination, moyennant une adaptation, pour prendre en compte les paires de documents et non plus les documents indépendamment. Les algorithmes de l'approche par liste requièrent des développements plus importants afin de résoudre le problème posé. Nous avons notamment vu qu'ils pouvaient induire de grands temps d'exécution. La relative facilité d'implémentation et d'adaptation des algorithmes des approches par point et paire constitue un avantage.

Néanmoins, plusieurs travaux [Xia 2008, Liu 2009b, Li 2011, Liu 2011] ont affirmé que l'approche par liste était la plus adaptée pour modéliser le problème d'ordonnancement. En effet, les fonctions de perte utilisées sont directement basées sur la comparaison et la minimisation des écarts entre une liste de référence et une liste candidate, qui est l'objet de l'apprentissage d'ordonnancement. Par ailleurs, les fonctions de perte sont définies au niveau de la requête (comparaison des deux listes de documents pour une requête q) ce qui n'est pas le cas des approches par point ou paire. L'approche par point, qui considère les documents indépendamment, ne tient compte ni de l'ordre entre documents ni de la structure entre requête et documents qui est perdue. L'approche par paire résout partiellement le premier problème en tenant compte de l'ordre relatif des documents deux à deux.

Plusieurs travaux se sont intéressés à la comparaison des approches et algorithmes [Verbene 2009, Zhang 2009, Chapelle 2011a, Liu 2011], dans le but de déterminer les méthodes les plus performantes. Néanmoins, tous les travaux n'arrivent pas nécessairement aux mêmes conclusions.

La majorité des travaux en apprentissage d'ordonnancement s'accordent sur le

fait que la performance d'un algorithme est variable suivant le jeu de données et la mesure d'évaluation considérée. Ainsi pour la même mesure, certains algorithmes seront très bons sur certains jeux de données et très mauvais sur d'autres. Si nous nous basons sur les résultats présentés dans l'étude de Liu [Liu 2011], RankBoost est le meilleur algorithme sur le jeu de données NP2003 et l'avant dernier sur NP2004, que nous considérions la MAP ou le NDCG@10.

Un grand nombre de travaux admettent que les approches par liste sont meilleures que les approches par paire, elles-mêmes meilleures que les approches par point. Les études comparatives présentées dans [Zhang 2009] et [Liu 2011] montrent ainsi que les algorithmes de l'approche par liste testés donnent globalement de meilleurs résultats que les algorithmes de l'approche par paire. Néanmoins, ces analyses ne sont effectuées que sur un nombre réduit d'algorithmes et de jeux de données, en l'occurrence la collection LETOR 3.0 pour [Zhang 2009] e [Liu 2011] et un jeu de données commercial pour [Zhang 2009]. Il est par ailleurs étonnant de constater que les valeurs de MAP présentées pour les mêmes algorithmes et jeux de données différent entre les deux études. Ceci peut être expliqué par une procédure de validation différente (choix du modèle qui maximise le NDCG au lieu de la MAP sur l'échantillon de validation par exemple), mais indique aussi que les performances des algorithmes peuvent être fortement influencées par le protocole expérimental. Par ailleurs, les analyses ne considèrent que certaines mesures d'évaluation. De plus, l'analyse effectuée dans [Liu 2011] se contente de comparer les valeurs des mesures de façon purement quantitative, sans prendre en compte des tests statistiques pour mesurer l'amélioration significative de la mesure d'un algorithme à l'autre. L'auteur utilise en particulier une mesure nommée *winning number* pour évaluer la performance globale d'un algorithme (*cf.* définition 12).

Définition 12 (*Winning Number*)**.**

Soient
- N le nombre de jeux de données,
- M le nombre d'algorithmes,
- i l'indice de l'algorithme considéré,
- $M_i(j)$ la valeur de la mesure M pour l'algorithme i sur le jeu de données j,

alors le *winning number* $S_i(M)$ de l'algorithme i pour la mesure M est défini de la façon suivante :

$$S_i(M) = \sum_{j=1}^{N} \sum_{k=1}^{M} \mathbb{1}_{\{M_i(j) > M_k(j)\}}$$

Selon cette mesure, l'algorithme est d'autant meilleur qu'il surpasse un grand nombre d'algorithmes sur l'ensemble des jeux de données pour une mesure de RI considérée. Le principe de la mesure peut paraître bon, mais ne tient pas compte de la significativité réelle de l'amélioration entre deux algorithmes. En effet, le *winning number* considère qu'un algorithme est meilleur qu'un autre si la valeur de la mesure de RI considérée est au-dessus de celle d'un autre algorithme, même si l'écart est faible ou non significatif (par exemple, ListNet est meilleur que RankSVM sur TD2003 lorsque le NDCG@10 est considéré, alors que la différence entre les deux valeurs est de 0.002, ce qui n'est pas une amélioration notable pour l'utilisateur).

Par ailleurs, une analyse similaire sur les jeux de données de LETOR 4.0, qui comportent plus de requêtes, montre que les résultats entre les deux approches sont comparables. Cette analyse seule ne suffit pas à généraliser, le nombre d'algorithmes testés étant limité. Néanmoins, ils rejoignent les constatations du Yahoo ! Learning to Rank Challenge [Chapelle 2011a] et de [Chapelle 2009a] qui mettent en évidence que les performances des algorithmes sont comparables lorsque les jeux de données considérés sont de grande dimension. Chapelle [Chapelle 2009a] indique ainsi que dans le cas d'algorithmes utilisant des fonctions non linéaires, les pourcentages d'amélioration obtenus par les approches par liste sont compris entre 0 et 0.5 %, comparativement aux approches par paire.

Il est donc difficile de déterminer quelle approche est la meilleure. Bien que les algorithmes de l'approche par liste soient généralement considérés comme plus performants, des travaux suggèrent que les performances des approches sont comparables sur les jeux de données de grande dimension. Par ailleurs, des travaux récents [Blanco 2011, Busa-Fekete 2012] indiquent qu'il est nécessaire d'être prudent lors des comparaisons des performances des algorithmes. En effet, [Busa-Fekete 2012] montre que les différents scripts permettant de calculer les valeurs des mesures peuvent renvoyer des résultats différents pour un même algorithme et un même jeu de données. Des différences d'implémentations peuvent ainsi engendrer des variations importantes dans les résultats obtenus. La présence de requêtes sans documents pertinents dans les jeux de données semble également avoir une influence sur les mesures d'évaluation. [Blanco 2011] met en évidence que de petites variations sur les scores de pertinence peuvent engendrer de grandes améliorations des mesures de RI, sans que ces améliorations aient du sens. Les résultats doivent donc être interprétés de façon prudente.

Conclusion

Dans ce chapitre, nous avons introduit le principe de l'apprentissage d'ordonnancement ainsi que les principales approches et algorithmes proposés. Nous avons également présenté les jeux de données de référence dédiés à l'apprentissage d'ordonnancement, dont les collections LETOR 3.0 et 4.0, très populaires dans la communauté et utilisées dans nos travaux. Nous avons également discuté de la comparaison des approches et indiqué qu'il pouvait être délicat de déterminer quel type d'ap-

proches était la meilleure parmi les trois existantes (*pointwise*, *pairwise*, *listwise*).

Dans le chapitre suivant, nous présentons certaines limites de l'apprentissage d'ordonnancement qui motivent les travaux réalisés au cours de cette thèse. Nous détaillons notre proposition et nos contributions.

Chapitre 3

Vers un nouveau système d'ordonnancement adaptatif : motivation et contributions

Sommaire

Dans les chapitres précédents, nous avons introduit des concepts et méthodes relatifs à la RI et à l'apprentissage d'ordonnancement. Nous présentons maintenant notre sujet de recherche, l'apprentissage d'ordonnancement adaptatif ainsi que nos contributions.

Dans une première partie, nous montrons les limites des méthodes d'ordonnancement actuelles. Nous introduisons l'apprentissage d'ordonnancement adaptatif comme une piste d'amélioration. L'apprentissage d'ordonnancement adaptatif constitue le cadre de nos travaux.

Dans une deuxième partie, nous dressons un panorama des contributions apportées au cours de la thèse dans ce contexte.

Dans une troisième partie, nous présentons la liste des publications, acceptées et soumises, ainsi que les présentations à des conférences et séminaires effectuées au cours de la thèse.

3.1 Vers un apprentissage d'ordonnancement adaptatif

Nos travaux de recherche portent sur la sélection de variables en apprentissage d'ordonnancement et l'apprentissage d'ordonnancement adaptatif, c'est-à-dire, l'apprentissage de fonctions d'ordonnancement spécifiques à un certain contexte de recherche. Dans cette section, nous présentons les limites de l'apprentissage d'ordonnancement "traditionnel". Nous motivons l'utilisation de la sélection de variables en apprentissage d'ordonnancement et la nécessité de concevoir des méthodes adaptatives. Nous illustrons nos propos dans le cadre de l'adaptation aux requêtes.

Les méthodes d'apprentissage d'ordonnancement présentées précédemment combinent un grand nombre de caractéristiques, de quelques centaines à plusieurs milliers. Cette volumétrie a un impact sur les temps de calcul et de traitement, mais aussi sur la mémoire nécessaire au stockage des données. Il paraît alors nécessaire de proposer de nouvelles approches permettant de réduire la dimension des problèmes traités [Geng 2007]. Des méthodes de sélection de variables ont été proposées et se sont avérées très efficaces pour résoudre ce problème. Nous en dressons un panorama au chapitre 4. Certaines méthodes considèrent la sélection de variables comme une étape préliminaire à l'apprentissage des fonctions d'ordonnancement [Geng 2007, Yu 2009, Dang 2010], tandis que d'autres réalisent sélection et apprentissage simultanément [Sun 2009, Lai 2013a, Lai 2013b]. Parmi ces dernières, les méthodes basées sur les SVM utilisant des normes parcimonieuses comme la norme ℓ_1 (*cf.* chapitre 5) se révèlent très efficaces [Lai 2013a]. Néanmoins, d'autres méthodes basées sur les SVM, utilisant des normes dites non convexes, sont connues dans le domaine de l'apprentissage pour supprimer beaucoup plus de caractéristiques tout en conservant de bonnes qualités de prédiction. A notre connaissance, ce type de méthodes n'a pas été étudié dans le contexte de l'apprentissage d'ordonnancement. Une contribution majeure de notre thèse est la proposition et l'évaluation de nouveaux algorithmes de sélection de variables dédiées à l'apprentissage d'ordonnancement et utilisant des normes non convexes.

Les méthodes d'apprentissage d'ordonnancement présentées au chapitre précédent présentent également une autre limite, liée à la prise en compte du contexte. Elles permettent d'apprendre une fonction d'ordonnancement unique utilisée pour classer l'ensemble des documents relatifs à l'ensemble des requêtes, sans distinction. Ainsi, les critères utilisés pour ordonner les documents sont les mêmes quel que soit le besoin d'information. Or les requêtes peuvent grandement différer tant du point de vue du besoin d'information que de leur sémantique et il paraît nécessaire d'adapter l'ordonnancement au contexte de la requête [Bai 2008]. Des travaux se sont ainsi intéressés à la mise en place de typologies de requêtes basées sur le but de l'utilisateur [Broder 2002, Rose 2004, Jansen 2008] ou spécifiques à des contextes particuliers, comme les moteurs de recherche de blogs [Mishne 2006] ou d'informations géoréférencées [Gan 2008]. D'autres études se sont plus concentrées sur la prédiction de la difficulté de la requête [Mothe 2005] ou ont préféré déterminer le meilleur système à utiliser en fonction du type de la requête [Bigot 2011].

L'idée sous-jacente de ces travaux est la suivante : les requêtes et besoins sont

différents par nature et ils devraient être traités différemment par les systèmes de recherche d'information. Une nouvelle branche de recherche en apprentissage d'ordonnancement a ainsi émergé : l'apprentissage d'ordonnancement dépendant des requêtes. Le principe général est d'adapter les fonctions d'ordonnancement aux classes de requêtes spécifiées, en apprenant une fonction d'ordonnancement propre à chacune de ces classes. Les caractéristiques utilisées pour apprendre les fonctions d'ordonnancement sont identiques pour chaque classe, seuls les poids sont modifiés. Un état de l'art en apprentissage d'ordonnancement dépendant des requêtes est proposé au chapitre 4. Nous proposons d'appliquer la sélection de variables à l'apprentissage dépendant des requêtes, afin de ne conserver que les caractéristiques pertinentes à chaque type de besoin.

Nos travaux se situent dans le cadre de la sélection de variables en apprentissage d'ordonnancement, vu comme un moyen de résoudre les problèmes de volumétrie rencontrés et d'améliorer l'adaptation au contexte en ne considérant que les caractéristiques pertinentes pour un type de requêtes donné. Nous proposons d'intégrer une étape de sélection de variables au processus d'apprentissage dépendant des requêtes afin d'en améliorer la performance. Nous introduisons brièvement les différentes propositions et contributions dans la section suivante. Nous les situons par rapport aux différents chapitres du mémoire.

3.2 Contributions

Dans nos travaux, nous nous intéressons à la sélection de variables en apprentissage d'ordonnancement et à son utilisation pour la prise en compte du besoin d'information et l'adaptation du processus d'ordonnancement sur les moteurs de recherche [Laporte 2011b]. Le besoin d'information peut être exprimé sous la forme d'une certaine tâche de recherche ou à partir des requêtes. Nous illustrons l'adaptation aux besoins d'information en traduisant ceux-ci par des groupes de requêtes, sans que cela limite la portée des méthodes proposées.

Nos travaux s'articulent autour de quatre contributions principales, qui sont présentées dans la suite de cette section.

3.2.1 Contribution 1 : Proposition d'un système d'ordonnancement adaptatif basé sur la sélection de variables

La première contribution est la proposition d'un système d'ordonnancement adapté aux différents besoins d'information. Celui-ci est basé sur trois éléments fondamentaux : la catégorisation des besoins d'information en différentes classes, la sélection des caractéristiques pertinentes et l'apprentissage des fonctions d'ordonnancement adaptées à ces catégories, n'utilisant que les caractéristiques pertinentes. L'utilisation de la sélection de variables à des fins d'adaptation au contexte dans le cadre de l'apprentissage d'ordonnancement est, à notre connaissance, une proposition nouvelle. Elle a été introduite dans un article publié en 2013 dans la revue francophone *Document Numérique* [Laporte 2013c]. Dans ces travaux, nous nous

sommes intéressés à l'adaptation aux types de requêtes, traduisant un type de besoin.

Le **chapitre 4** détaille nos propositions et notre positionnement. Dans un premier temps, nous dressons un état de l'art de l'apprentissage dépendant des requêtes, qui constitue le cadre de notre travail. Dans un second temps, nous introduisons la sélection de variables comme une piste d'amélioration. Nous présentons le principe de la sélection de variables et motivons son utilisation. Dans un troisième temps, nous décrivons le système d'ordonnancement adaptatif basé sur la sélection de variables et mettons en évidence la nécessité d'utiliser des algorithmes de sélection spécifiques. Enfin, nous dressons un état de l'art des approches de sélection de variables existantes et nous nous positionnons par rapport à elles.

3.2.2 Contribution 2 : Développement de nouveaux algorithmes de sélection de variables

Notre deuxième contribution correspond à la proposition, au développement et à l'évaluation d'algorithmes de sélection de caractéristiques dédiés à l'apprentissage d'ordonnancement. Ces algorithmes pourront être utilisés au sein du système d'ordonnancement adaptatif décrit dans la contribution 1.

Le **chapitre 5** décrit le principe des SVM et de l'utilisation de normes parcimonieuses, que nous utilisons dans nos algorithmes.

Dans le **chapitre 6**, nous proposons trois algorithmes de sélection utilisant les propriétés des normes mathématiques ℓ_1 et ℓ_0. Nous montrons comment ces algorithmes peuvent effectuer une sélection de variables par des méthodes de repondération et l'utilisation d'un seuil de nullité des coefficients.

Un article est en cours de préparation pour une soumission dans une conférence internationale [Laporte 2013b].

Dans le **chapitre 7**, nous proposons deux algorithmes pour effectuer la sélection de variables. Ces derniers ne nécessitent pas l'utilisation d'un seuil de nullité des variables, contrairement aux précédents. La suppression des variables est due uniquement à la propriété des normes et à l'utilisation du schéma de repondération. Le premier algorithme effectue la sélection par l'utilisation de SVM parcimonieux en norme ℓ_1 par le biais d'une méthode proximale. Le second algorithme effectue la sélection de variables par l'utilisation de normes non convexes. Il met en place une technique de repondération basée sur l'utilisation du premier algorithme.

Ces algorithmes et les résultats obtenus ont été acceptés en octobre 2013 pour publication dans la revue internationale *IEEE Transaction on Neural Networks and Learning Systems* (IF 3.766) [Laporte 2014].

3.2.3 Contribution 3 : Proposition d'un modèle d'évaluation implicite de la pertinence pour la création de jeux de données représentatifs

Un des verrous rencontrés au cours de la thèse est l'absence de jeux de données représentatifs pour l'évaluation d'un système adaptatif. En effet, les jeux de données de référence comme LETOR ne sont pas conçus pour l'évaluation de l'adaptation aux types de besoins ou de requêtes. Nous nous sommes intéressés à la construction de jeux de données représentatifs, basée sur l'utilisation des logs de connexion et l'extraction de jugements implicites de pertinence à partir des actions des utilisateurs. Le **chapitre 8** présente notre troisième contribution : la proposition d'un modèle d'évaluation de la pertinence basé sur les clics des utilisateurs extraits de fichiers de connexion. Un état de l'art des méthodes existantes est proposé [Laporte 2012c]. Le modèle que nous définissons est spécialement développé pour la modélisation de la pertinence sur des documents pouvant être cliqués plusieurs fois, ce que les approches existantes ne prennent pas en compte. L'objectif final de cette approche est de prédire la pertinence d'un grand nombre de documents vis-à-vis d'un grand nombre de requêtes, de façon automatique et rapide.

Cette approche a été présentée dans deux articles [Laporte 2012d, Laporte 2013a] et a été évaluée sur des données extraites du moteur de recherche d'informations géoréférencées Nomao.

La dernière contribution complète l'évaluation des algorithmes dans un cadre applicatif d'évaluation sur le moteur de recherche commercial Nomao.

3.2.4 Contribution 4 : Évaluation de la sélection de variables et de l'ordonnancement adaptatif sur le moteur de recherche d'informations géoréférencées Nomao

Notre dernière contribution correspond à l'évaluation des algorithmes de sélection de variables du système d'ordonnancement adaptatif sur un jeu de données réel issu du moteur de recherche Nomao.

Dans une première phase, un jeu de données de grande dimension a été extrait des fichiers de connexion de Nomao. La pertinence requête-document a été évaluée à l'aide du modèle d'évaluation implicite de la pertinence proposé à la contribution 3. Dans une deuxième phase, un système d'extraction du type de besoin à partir de la requête a été mis en place. Ce système permet d'associer à chaque requête sa catégorie de besoin. Les catégories de besoins utilisées sont celles définies par le moteur de recherche Nomao et sont détaillées au chapitre 9. Dans une troisième phase, les algorithmes de sélection de variables ont été évalués sur le jeu de données. Une première expérimentation a consisté à évaluer la performance des algorithmes lorsque le type de besoin n'est pas pris en compte. Les algorithmes proposés apparaissent alors comme plus efficaces que l'état de l'art pour supprimer un grand nombre de variables des modèles, tout en conservant une qualité d'ordonnancement

équivalente. Une autre expérimentation sur le même jeu de données est en cours pour évaluer l'apport de la sélection de variables dans le système d'apprentissage d'ordonnancement adaptatif. Les résultats sont attendus d'ici la fin d'année 2013.

Le **chapitre 9** présente le moteur Nomao et décrit le jeu de données utilisé pour l'évaluation.

Le **chapitre 10** rappelle la méthodologie d'évaluation et commente les résultats obtenus.

3.3 Publications et communications

Les contributions et travaux ont donné lieu à des publications, ainsi qu'à des présentations dans des conférences ou séminaires. Une liste détaillée des publications acceptées, des communications ainsi que des articles soumis ou en préparation est présentée ci-après. Les articles précédés d'un astérisque ont donné lieu à une présentation orale à la conférence par la doctorante.

3.3.1 Publications acceptées

3.3.1.1 Article de revue internationale

[Laporte 2014] Léa Laporte, Rémi Flamary, Stéphane Canu, Sébastien Déjean et Josiane Mothe. *Non-convex Regularizations for Feature Selection in Ranking with Sparse SVM.* IEEE Transactions on Neural Networks and Learning Systems, à paraître en 2014.

3.3.1.2 Article de revue nationale

[Laporte 2013c] Léa Laporte. *De l'apprentissage d'ordonnancement à l'adaptation au contexte : état de l'art et propositions.* Document numérique, Hermès, numéro spécial "Fertilisation croisée des contenus à l'ère du numérique", vol.16, n. 1, p. 97-121, 2013.

3.3.1.3 Article de conférence internationale avec actes et comités de lecture

*[Laporte 2013a] Léa Laporte, Sébastien Déjean et Josiane Mothe. *Multiple-clicks model for web search of multi-clickable documents.* International Conference on Enterprise Information Systems (ICEIS 2013), Angers, 4-7 juillet 2013.

3.3.1.4 Articles de conférence nationale avec actes et comité de lecture

– [Delpech 2013] Estelle Delpech, Laurent Candillier, Léa Laporte et Samuel Phan. *Identification de compatibilité entre descripteurs de lieux et apprentissage automatique.* Conférence nationale sur l'Extraction et la Gestion de Connaissances (EGC 2013), Toulouse, 29 janvier - 1er février 2013, p. 311-316.

- *[Laporte 2012d] Léa Laporte, Laurent Candillier, Sébastien Déjean, Laurent Candillier et Josiane Mothe. *Modélisation de la pertinence dans les moteurs de recherche géoréférencés*. INFormatique des Organisations et Systèmes d'Information et de Décision (INFORSID 2012), Montpellier, 29-31 mai 2012, p.281-297.
- *[Laporte 2012c] Léa Laporte. *Ordonnancement des résultats sur les moteurs de recherche : principes, limites et applications au géoréférencement*. Séminaire de Veille Stratégique, Scientifique et Technologique (VSST 2012), Ajaccio, 23-25 mai 2012.
- *[Laporte 2011b] Léa Laporte. *RI et contextualisation*. Forum Jeunes Chercheurs INFormatique des Organisations et Systèmes d'Information et de Décision (INFORSID 2011), Montpellier, 25 mai 2011, p. 443-444.

3.3.2 Communications dans des conférences et séminaires sans actes

- [Laporte 2012a] *Approche proximale pour la sélection de variables en apprentissage d'ordonnancement via des SVM parcimonieux*. Séminaire DocToMe, IRIT Université Paul Sabatier, Toulouse, 20 décembre 2012.
- [Laporte 2012b] *Evaluation de la pertinence dans les moteurs de recherche géoréférencés - Application des SVM*. Séminaire FREMIT, IRIT Université Paul Sabatier, Toulouse, 21 mars 2012.
- [Laporte 2011a] *Data Mining pour la recherche d'information contextuelle - Learning to rank dans les moteurs de recherche d'information géoréférencés*. Journées Fouile de données du GDR-I3, BU Lyon 1 Campus de la Doua, Lyon, 28 septembre 2011.
- [Laporte 2010b] *Sélection de variables et apprentissage de fonctions d'ordonnancement en recherche d'information*. Journées de jeunes statisticiens de la société française de Statistique, Aussois, 5-9 septembre 2011
- [Laporte 2010a] *Learning to rank : géolocalisation et personnalisation pour les systèmes de recherche d'information*. Séminaire FREMIT, IRIT Université Paul Sabatier, Toulouse, 20-21 septembre 2010.

3.3.3 Publications soumises ou en préparation

- Estelle Delpech, Léa Laporte. *Semantic relatedness measures for duplicate record detection*. Soumis dans Advances in Knowledge Discovery and Management, vol. 5 (AKDM5), Springer.
- Léa Laporte, Sébastien Déjean et Josiane Mothe. *Reweighted algorithms for feature selection in learning to rank*. Soumis.

Conclusion

Nos travaux de thèse s'intéressent particulièrement à la proposition d'algorithmes de sélection de variables dédiés à l'apprentissage d'ordonnancement. La

sélection de variables constitue un moyen efficace de réduction de la volumétrie des problèmes d'apprentissage. Par ailleurs, elle est vue comme une piste d'amélioration de l'apprentissage adaptatif. L'apprentissage d'ordonnancement non adaptatif ne prend pas en compte la grande diversité et la spécificité des requêtes et des besoins d'information. Les résultats obtenus peuvent ne pas être optimaux. L'intégration de la sélection de variables au sein d'un système d'apprentissage d'ordonnancement constitue un piste d'amélioration de l'apprentissage adaptatif et sert également de cadre à ces travaux de thèse.

Les contributions apportées par la thèse sont décrites et positionnées dans le cadre de l'apprentissage d'ordonnancement adaptatif. Elles sont abordées en détail dans la deuxième partie de ce manuscrit.

La partie II regroupe les propositions

- d'un système d'apprentissage d'ordonnancement adaptatif basé sur la sélection de variables (chapitre 4),
- de trois algorithmes de sélection de variables basés sur l'utilisation d'un schéma de repondération de SVM en norme ℓ_2 et d'un seuil de nullité des variables (chapitre 6),
- de deux algorithmes, l'un basé sur l'utilisation d'une approche proximale pour la résolution des SVM en norme ℓ_1, l'autre basé sur un schéma de repondération de l'algorithme norme ℓ_1. Ils ne nécessitent pas la définition d'un seuil de nullité des variables pour effectuer la sélection (*cf.* chapitre 7)
- d'un modèle de pertinence pour la construction de jeu de données nécessaires à l'évaluation dans un contexte d'exploitation commercial (chapitre 8) .

La partie III présente l'évaluation des algorithmes sur un jeu de données commercial, qui constitue une contribution applicative de nos travaux de thèse dans un contexte d'exploitation commercial (*cf.* chapitre 9).

Deuxième partie

De nouveaux algorithmes pour la sélection de variables en apprentissage d'ordonnancement

Résumé

Dans cette partie, nous présentons trois de nos contributions, soit les propositions :

- d'un système d'ordonnancement basé sur la sélection de variables,
- d'algorithmes de sélection de variables dédiés à l'apprentissage d'ordonnancement,
- d'un modèle d'évaluation implicite de la pertinence pour la construction de jeux de données d'apprentissage utilisables pour l'évaluation du système et des algorithmes proposés.

Dans le chapitre 4, nous introduisons le principe de la sélection de variables et motivons son utilisation en apprentissage d'ordonnancement. Nous proposons un système d'ordonnancement basé sur la sélection de variables afin d'apprendre des fonctions spécifiques à un groupe de requêtes. Nous dressons un état de l'art des approches de sélection pouvant être intégrées à ce système. Nous choisissons de considérer des approches de type *embedded* basées sur les SVM.

Dans le chapitre 5, nous présentons le principe des SVM en discrimination et en apprentissage d'ordonnancement. Nous nous intéressons plus particulièrement à l'utilisation de normes parcimonieuses, que nous définissons, pour effectuer la sélection de variables dans le cadre d'approches de types *embedded* . Nous justifions l'utilisation des SVM parcimonieux dans le cadre de ces travaux. Nous présentons les normes ℓ_1, ℓ_0, MCP, Log et ℓ_q, $q < 1$ que nous utilisons dans les algorithmes que nous proposons.

Dans le chapitre 6, nous proposons trois algorithmes de sélection de variables basés sur l'approximation des normes parcimonieuses ℓ_1 et ℓ_0 par repondération de la norme ℓ_2 et d'un seuil de nullité des variables. La norme ℓ_2 n'est pas parcimonieuse, mais le schéma de repondération permet de diminuer la valeur des poids des variables les moins pertinentes pour l'apprentissage, qui sont alors annulées grâce au seuil de nullité. Nous évaluons les algorithmes sur les cinq jeux de données issus de LETOR 3 et 4 : Ohsumed, HP2004, NP2004, TD2004 et MQ2008. Nous comparons leur performance en matière de MAP et de NDCG à trois algorithmes de l'état de l'art. Nous comparons les temps d'exécution des algorithmes proposés entre eux et analysons les ensembles de caractéristiques sélectionnés.

Dans le chapitre 7, nous proposons deux algorithmes que nous évaluons. Le premier algorithme utilise une approche proximale pour résoudre le problème en norme ℓ_1. Le second algorithme utilise une approche par repondération de la première méthode pour effectuer la sélection de variables via des régularisations non convexes (MCP, Log, et ℓ_q). Comparativement aux algorithmes du chapitre 6, ceux-ci sont directement parcimonieux. Il n'est pas nécessaire de fixer un seuil de nullité des variables. Nous évaluons ces algorithmes sur l'ensemble des jeux de données issus de LETOR 3 et 4. Nous comparons les valeurs de MAP et NDCG obtenues à celles de trois algorithmes de référence. Nous analysons également les ratio de parcimonie, c'est-à-dire la proportion de caractéristiques restantes après la sélection.

Enfin, dans le chapitre 8, nous proposons un modèle d'évaluation implicite de la pertinence à partir des logs de connexion et des actions des utilisateurs. Ce modèle permet d'évaluer la pertinence d'un document vis-à-vis d'une requête de façon automatique. Il est utilisé pour la création de jeux de données d'apprentissage issus d'un moteur commercial. Les jeux de données ainsi créés permettent l'évaluation des algorithmes proposés et du système dans le cadre d'un moteur commercial.

Chapitre 4

Un système d'ordonnancement adaptatif basé sur la sélection de variables

Sommaire

Dans ce chapitre, nous présentons notre première contribution : la proposition d'un système d'ordonnancement adaptatif, dépendant des requêtes et basé sur la sélection de variables.

Dans une première section, nous dressons un état de l'art des approches d'ordonnancement dépendant des requêtes, vis-à-vis desquelles nous nous positionnons. Nous introduisons la sélection de variables comme un moyen d'améliorer l'apprentissage adaptatif.

Dans une seconde section, nous définissons la sélection de variables, nous rappelons son principe et nous motivons son utilisation en apprentissage d'ordonnancement.

Dans une troisième section, nous décrivons le système d'ordonnancement adaptatif proposé, qui intègre des algorithmes de sélection de variables.

Enfin, dans une dernière section, nous dressons un état de l'art des algorithmes de sélection de variables en apprentissage d'ordonnancement. Nous positionnons nos travaux (*cf.* chapitres 6 et 7) par rapport à l'existant.

4.1 Apprentissage d'ordonnancement dépendant des requêtes

Comme nous l'avons indiqué au chapitre 3, l'apprentissage d'ordonnancement "traditionnel" présente l'inconvénient d'utiliser la même fonction d'ordonnancement pour tous les types de requêtes et besoins. L'adaptation de l'ordonnancement aux types de requêtes est vue comme une façon d'améliorer le classement des résultats et l'expérience de l'utilisateur.

Différents travaux se sont intéressés à l'adaptation de l'ordonnancement au type de besoins. Nous pouvons distinguer deux catégories d'approches. La première s'est concentrée sur la proposition et l'évaluation de fonctions de pertes dépendant de la requête. La deuxième s'est focalisée sur l'apprentissage de fonctions d'ordonnancement spécifiques à un groupe ou une classe de requêtes. Nous introduisons dans les sections suivantes quelques travaux représentatifs de chacune de ces catégories d'approches.

4.1.1 Méthodes basées sur la proposition d'une fonction de perte dépendant de la requête

Les méthodes de cette catégorie modifient directement les fonctions de pertes utilisées par les algorithmes d'apprentissage afin de prendre en compte chaque requête.

Zha *et al.* [Zha 2006] proposent une nouvelle fonction de perte définie à partir de la fonction initiale indépendante des requêtes et de certains paramètres propres à chaque requête.

Bian *et al.* [Bian 2010] proposent également de définir une fonction de perte spécifique à chaque requête et de l'utiliser pour adapter l'ordonnancement. Ils mettent en évidence que, bien que cette solution soit théoriquement intéressante, elle n'est pas réalisable en pratique, du fait du grand nombre de requêtes. Les auteurs proposent alors de considérer des catégories de requêtes. Celles-ci sont définies suivant la typologie de Bröder [Broder 2002].

Selon cette typologie, les requêtes peuvent être regroupées en trois catégories de besoin : informationnel, navigationnel et transactionnel. Une requête navigationnelle suppose la recherche d'une page particulière. Par exemple, "site internet lufthansa" semble indiquer que l'utilisateur recherche la page d'accueil du site de la compagnie aérienne. Une requête informationnelle suppose que l'utilisateur recherche plusieurs pages pertinentes sur un sujet donné. Par exemple, "sites touristiques Japon" indique que l'utilisateur recherche différents lieux touristiques à visiter au Japon. Le site de l'office du tourisme japonais, des pages d'accueil des musées, etc, sont po-

tentiellement des résultats pertinents. Enfin, une requête transactionnelle indique que l'utilisateur recherche une information dans le but d'effectuer une transaction. Par exemple, "achat billet avion Toulouse-Tokyo" indique que l'utilisateur souhaite acheter un billet pour le trajet Toulouse-Tokyo.

Bian *et al.* considèrent uniquement les requêtes informationnelles et transactionnelles dans leur travaux. La fonction de perte proposée est exprimée comme un modèle de mélange composé des fonctions spécifiques aux requêtes navigationnelles d'une part et aux requêtes informationnelles et transactionnelles d'autre part.

4.1.2 Méthodes basées sur la catégorisation des besoins et l'utilisation de fonctions d'ordonnancement spécifiques

Les travaux de cette catégorie proposent de créer des groupes de requêtes puis d'appliquer sur chacun d'eux une fonction d'ordonnancement spécifique.

Kang et Kim [Kang 2003] ont proposé de regrouper les requêtes en trois catégories correspondant chacune aux classes de Bröder : *topic relevance task* (requêtes informationnelles), *homepage finding task* (requêtes navigationnelles) et *service finding task* (requêtes transactionnelles).

Dans leurs expérimentations, Kang et Kim n'ont considéré que les deux premières catégories. Ils ont proposé différentes mesures de similarité qui, combinées par des algorithmes d'apprentissage supervisé, permettent d'affecter chaque requête à la catégorie adéquate. Enfin, ils ont utilisé une fonction d'ordonnancement spécifique pour chacune d'elles. Notons que les deux fonctions utilisées n'ont pas été déterminées par un algorithme d'apprentissage, mais manuellement.

D'autres travaux ont combiné l'utilisation de catégories de requêtes et d'algorithmes d'apprentissage d'ordonnancement.

Geng *et al.* [Geng 2008] ont ainsi proposé d'utiliser la méthode des k plus proches voisins afin de créer dynamiquement des groupes de requêtes sur lesquels des fonctions spécifiques sont apprises.

Dans la phase d'apprentissage, chaque requête est représentée dans l'espace des caractéristiques. Ses k plus proches voisines sont alors sélectionnées. Un modèle est appris sur l'extrait du jeu de données d'apprentissage contenant ces $k+1$ requêtes.

Dans la phase de test, chaque nouvelle requête soumise au système est représentée dans l'espace des caractéristiques. Sa plus proche voisine est recherchée parmi les requêtes connues et la fonction d'ordonnancement correspondante, précédemment apprise, est utilisée pour classer les résultats. Les expérimentations menées par les auteurs montrent qu'utiliser des fonctions spécifiques pour chaque groupe de requêtes améliore la qualité des listes de résultats.

Contrairement aux travaux de Geng *et al.* [Geng 2008] qui créent autant de modèles que de requêtes, Banerjee *et al.* [Banerjee 2009] ont proposé de regrouper celles-ci en C groupes et d'apprendre une fonction spécifique pour chacun d'eux.

Les auteurs définissent une nouvelle mesure de similarité afin de créer les groupes.

Chaque requête est représentée dans l'espace des caractéristiques par l'ensemble D_q des vecteurs des documents qui lui sont associés. Une analyse en composantes principales est effectuée et les P composantes principales $(u_1, \ldots, u_P)$ sont extraites. La similarité entre deux requêtes est définie à partir des composantes principales de la façon suivante :

$$sim(q, q') = \frac{1}{P} \sum_{p=1}^{P} |u_p^\top u'_p|$$

Cette similarité est utilisée par l'algorithme de classification non supervisée pour regrouper les requêtes en C groupes ; une fonction d'ordonnancement est alors apprise pour chacun d'eux.

Dans la phase de test, chaque nouvelle requête soumise au système est comparée à chacune des requêtes connues, puis affectée au groupe dont elle est la plus similaire. La fonction d'ordonnancement apprise pour ce groupe est alors utilisée pour ordonner les résultats.

Enfin, Liu *et al.* [Liu 2009c] ont également proposé de créer des groupes de requêtes en se basant sur la similarité des URL des documents restitués en réponse à la requête. Des fonctions d'ordonnancement sont alors apprises spécifiquement pour chacun des groupes de requêtes ainsi formés. Les auteurs montrent que cette méthode permet d'améliorer la qualité de la liste de résultats obtenue.

Nos travaux se situent dans la deuxième catégorie d'approches, qu'ils complètent en proposant d'apprendre non seulement une fonction d'ordonnancement pour chaque catégorie de requêtes, mais aussi de sélectionner uniquement les caractéristiques pertinentes pour chaque catégorie. Ce dernier point est possible grâce à l'utilisation d'algorithmes de sélection de variables. Dans la section suivante, nous rappelons le principe de la sélection de variables et motivons son utilisation en apprentissage d'ordonnancement adaptatif.

4.2 Sélection de variables : principe et motivation pour une utilisation en apprentissage d'ordonnancement

Dans un premier temps, nous rappelons le principe de la sélection de variables, initialement proposée dans le contexte de la discrimination, ainsi que les trois grands types d'approches existantes. Dans un second temps, nous motivons son utilisation dans le contexte de l'apprentissage d'ordonnancement.

4.2.1 Sélection de variables : principe et approches

La sélection de variables en discrimination est une thématique ancienne, qui a connu un essor depuis le début des années 90. L'augmentation continue du volume d'information disponible dans les bases de données et l'émergence de disciplines de recherche à forte volumétrie (RI, web, données omiques en biologie) ont fait apparaître de nouveaux enjeux de contrôle de la dimensionalité des données et de

la qualité de prédiction des modèles appris. La sélection de variables est définie comme la recherche du meilleur sous-ensemble de caractéristiques pertinentes utilisables par un algorithme pour apprendre le modèle ayant la meilleure qualité de prédiction. Les approches de sélection de variables, qui permettent de supprimer les caractéristiques non pertinentes pour la modélisation d'un problème, sont importantes à plusieurs niveaux : facilitation de l'interprétation des modèles, réduction des temps de calcul pour la préparation des données et l'apprentissage des modèles, réduction des espaces de stockage requis ou encore amélioration de la performance des prédicteurs [Guyon 2003].

Trois grands types d'approches ont été proposés : les approches *filter* (par filtre), *wrapper* (encapsulantes) et *embedded* (embarquées). Dans l'approche *filter*, la sélection de variables est une étape de pré-traitement au cours de laquelle un sous-ensemble de caractéristiques est sélectionné indépendamment du classifieur utilisé pour apprendre le modèle. Dans l'approche *wrapper*, la sélection de variables est également une étape de pré-traitement. Le classifieur est utilisé comme boîte noire pour évaluer, via un score, la qualité de prédiction des modèles appris sur chaque sous-ensemble de caractéristiques possibles. Celui qui obtient le score le plus élevé est sélectionné. Enfin, dans l'approche *embedded*, la sélection de variables est incorporée à la phase d'apprentissage du modèle. Elle est spécifique au classifieur utilisé. Les variables à sélectionner et le modèle sont ainsi déterminés simultanément.

De nombreux algorithmes de sélection de variables existent dans le contexte de la discrimination. Récemment, des travaux ont proposés des méthodes de sélection de variables spécifiques à l'apprentissage d'ordonnancement en RI. Dans ce chapitre, nous nous concentrons sur ces dernières. Le lecteur intéressé par les méthodes de sélection de variables en discrimination pourra se référer à l'article de Guyon et Elisseeff de 2003 [Guyon 2003].

4.2.2 Motivations de l'utilisation de la sélection de variables en apprentissage d'ordonnancement

Au cours des dernières décennies, de nombreux modèles et mesures de similarité tels que BM25, PageRank, TF-IDF ou les modèles de langue ont été proposés afin d'ordonner les documents vis-à-vis d'une requête soumise par un utilisateur. Les algorithmes d'apprentissage d'ordonnancement ont été développés afin de combiner les résultats de ces modèles pour apprendre des fonctions optimisant le classement des documents. Les résultats des modèles deviennent les caractéristiques utilisées par les fonctions d'ordonnancement.

L'ajout continu et massif de nouvelles caractéristiques décrivant les requêtes et les documents a abouti à l'émergence de problématiques liées à la volumétrie des données : la présence de caractéristiques non pertinentes et/ou bruitées ou encore l'augmentation des temps de calcul. Ces problématiques ne sont pas propres à l'apprentissage d'ordonnancement. En discrimination, de nombreux travaux [John 1994, Guyon 2003] ont montré que la sélection de variables est un moyen efficace pour résoudre ces problèmes. Plus récemment, la sélection de variables a été

étudiée pour résoudre trois enjeux principaux en apprentissage d'ordonnancement, que nous détaillons ci-après : le contrôle de la qualité des fonctions ou modèles d'ordonnancement, le contrôle des temps de calcul via la suppression des variables non pertinentes et l'adaptation des fonctions d'ordonnancement aux types de recherche ou de besoins.

4.2.2.1 Enjeu 1 : Contrôle de la qualité des modèles

L'ajout massif de caractéristiques au sein des modèles appris a deux conséquences directes. D'une part, les modèles deviennent d'autant plus difficiles à interpréter que le nombre de caractéristiques qu'ils utilisent augmente. D'autre part, la présence de caractéristiques obsolètes, non pertinentes et/ou bruitées conduit à la dégradation de la qualité de prédiction des modèles appris. Il apparaît ainsi nécessaire de réduire le nombre de variables utilisées par les modèles, en supprimant celles qui ne sont pas pertinentes. De fait, la sélection de variables, qui répond directement à ce besoin, est une approche naturelle pour la résolution de ce problème.

4.2.2.2 Enjeu 2 : Contrôle des temps de calcul

L'augmentation du nombre de variables utilisées par les algorithmes a un impact sur les temps de calcul au niveau de la création des jeux de données (d'apprentissage de test et de validation) et de l'apprentissage des modèles d'ordonnancement.

Lors de la création des jeux de données d'apprentissage, chaque caractéristique est calculée pour chacune des paires requête-document qui constituent le jeu de données. L'augmentation du nombre de caractéristiques entraîne nécessairement une augmentation du temps de calcul. De fait, la création des jeux de données devient plus coûteuse. Par ailleurs, les algorithmes d'apprentissage sont contraints de résoudre des problèmes d'optimisation de bien plus grande dimension, ce qui entraîne une augmentation des temps de calcul des fonctions d'ordonnancement. Le contrôle de ces temps est ainsi un enjeu important pour l'utilisation des algorithmes en situation réelle.

Pour résoudre ces problèmes, plusieurs solutions sont envisageables. Tout d'abord, il semble nécessaire de développer des algorithmes pouvant traiter de grands volumes de données [Sculley 2009, Chapelle 2011b, Liu 2011]. Ensuite, un effort peut également être fait pour réduire la dimension du problème en sélectionnant les exemples constituant le jeu de données [Long 2010, Chapelle 2011b, Liu 2011]. Ainsi, en sélectionnant des observations plus informatives, mais en moins grand nombre, pour la constitution des données d'apprentissage, il est possible d'agir sur la volumétrie du problème. Enfin, réduire le nombre de variables permet directement de réduire le nombre de calculs à effectuer. La sélection de variables est donc une méthode permettant de contrôler les temps de calcul. Plus largement, la sélection de variables, en temps qu'approche de réduction de la dimension des problèmes, permet une dimension des temps de traitement et de la mémoire nécessaire au stockage et au traitement.

Plus récemment, un nouvel enjeu est apparu en apprentissage d'ordonnancement : l'adaptation de l'ordonnancement au contexte et plus précisément, à la requête. Nous motivons l'utilisation de la sélection de variables dans ce cadre dans la section suivante.

4.2.2.3 Enjeu 3 : Adaptation des fonctions d'ordonnancement aux types de recherche ou de besoin

L'idée générale est de déterminer des groupes de requêtes similaires, soit par des méthodes de classification non supervisée, soit par l'évaluation d'une mesure de similarité, puis d'apprendre une fonction d'ordonnancement spécifique pour chaque groupe. C'est le principe de l'ordonnancement dépendant des requêtes. En pratique, les fonctions d'ordonnancement apprises utilisent le même ensemble de variables, seuls les poids affectés à chaque variable varient suivant le groupe considéré. Ces approches présentent, à notre sens, deux limites. D'une part, elles ne résolvent pas le problème de volumétrie présenté précédemment. D'autre part, les modèles appris, bien qu'adaptés aux requêtes, ne sont pas nécessairement optimaux. En effet, ils peuvent contenir des caractéristiques non pertinentes pour l'ensemble des requêtes, aucune sélection n'ayant était réalisée. Par ailleurs, les mêmes caractéristiques sont utilisées pour chaque type de besoins, or, toutes ne sont pas nécessairement pertinentes pour tous les types de besoins. Nous proposons d'utiliser la sélection de variables avec deux objectifs : supprimer les caractéristiques non pertinentes pour tous les types de requêtes et utiliser des caractéristiques spécifiques à chaque groupe. L'adaptation est effectuée à la fois par l'apprentissage de poids différents pour une même variable suivant les groupes et par l'utilisation de variables spécifiques à chaque groupe. Les requêtes elles-mêmes peuvent être regroupées sans limitation suivant différents critères : similarités sémantiques, similarités de besoin, similarités de contexte (moteurs géoréférencés vs non géoréférencés), ...

La sélection de variables apparaît comme une approche permettant non seulement de réduire le bruit et la volumétrie des données, mais aussi d'apprendre des fonctions d'ordonnancement réellement adaptées au contexte de la recherche. Dans la section suivante, nous proposons et décrivons un système d'ordonnancement adaptatif, basé sur la sélection de variables.

4.3 Proposition d'un système d'ordonnancement adaptatif basé sur la sélection de variables

Afin d'adapter l'ordonnancement au type de besoin considéré, nous proposons de combiner l'apprentissage de fonctions d'ordonnancement spécifiques à un type de besoin ou à un groupe de requête et la sélection de variables. Ce système a été introduit dans un article intitulé "De l'apprentissage d'ordonnancement à l'adaptation au contexte" publié dans la revue francophone Document Numérique [Laporte 2013c].

Nous nous plaçons dans le cadre de l'apprentissage d'ordonnancement hors ligne. Le système proposé est constitué de deux entités :

Entité 1 le système d'apprentissage d'ordonnancement hors ligne, qui, à partir d'un jeu de données étiqueté, apprend des fonctions d'ordonnancement propre à chaque catégorie de besoin,

Entité 2 le système d'ordonnancement en ligne, qui à partir de la requête soumise par l'utilisateur, identifie la catégorie de besoin, sélectionne la fonction d'ordonnancement correspondante et l'utilise pour ordonner les résultats de recherche.

Le système d'apprentissage d'ordonnancement hors ligne est lui-même constitué de deux blocs :

Bloc 1 l'outil de catégorisation des besoins ou des requêtes, qui crée les groupes de requêtes ou de besoins

Bloc 2 l'algorithme d'apprentissage des fonctions d'ordonnancement, qui apprend une fonction d'ordonnancement spécifique pour chacun des groupes.

Nous schématisons le système d'ordonnancement sans sélection de variables, utilisé par les méthodes de l'état de l'art, à la figure 4.1.

L'originalité de notre approche est d'intégrer la sélection de variables à l'apprentissage. Les méthodes de l'état de l'art utilisent les mêmes caractéristiques pour chaque fonction, qu'elles pondèrent différemment suivant les catégories. Notre approche va plus loin, en adaptant non seulement le poids de chaque caractéristique, mais aussi l'ensemble des caractéristiques utilisées. L'avantage est de n'utiliser que les caractéristiques pertinentes à chaque besoin, tout en modifiant la volumétrie du problème. Les algorithmes de sélection de variables nous permettent de ne conserver que les caractéristiques les plus pertinentes pour l'apprentissage de la fonction d'ordonnancement. Ceci présente deux avantages majeurs : la dimensionnalité du problème est réduite - ce qui peut entraîner un gain de temps de calcul - et les caractéristiques non pertinentes ou bruitées ne sont pas utilisées. Il a été montré que ce dernier point permet d'améliorer la qualité de prédiction des modèles [Geng 2007].

Le modèle que nous proposons diffère de celui de l'état de l'art sur la façon d'effectuer l'apprentissage des fonctions d'ordonnancement. Nous modifions le bloc 2 "Outil d'apprentissage pour l'ordonnancement" de l'entité 1 "Système hors ligne" pour intégrer la sélection de variables, tandis que l'entité "Système en ligne" n'est pas modifié. Les modifications apportées sont décrites à la figure 4.2 et détaillées ci-après.

Dans un premier temps et de façon similaire à l'existant, notre système constitue des groupes ou catégories de requêtes à partir du jeu de données d'apprentissage. Des typologies, des méthodes de classification supervisée ou non supervisée peuvent être utilisées dans ce but.

Dans un second temps, des fonctions d'ordonnancement spécifiques aux groupes précédemment constitués sont apprises en utilisant des approches de sélection de variables. Deux cas de figures sont possibles à ce niveau, selon que nous utilisons des approches de type *filter* ou *wrapper* ou des approches de types *embedded* .

Entité 1 (Système hors ligne).

Entrée : Jeu de données d'apprentissage $\{\mathbf{x}_i, y_i\}_{i=1,\dots,n}$

Bloc 1 (Outil de classification des requêtes).

Entrée : Requêtes et caractéristiques associées

Utilisation de :
- Typologies (de Bröder, ...)
- Méthodes de classification non supervisée (k plus proches voisins, Classification ascendante hiérarchique, ...)
- Méthodes de classification supervisée

pour la catégorisation des requêtes.

Sortie : Groupes de requêtes

Bloc 2 (Outil d'apprentissage pour l'ordonnancement).

Entrée : Sous-ensemble du jeu de données d'apprentissage correspondant à un groupe de requêtes

Apprentissage d'une fonction de d variables à l'aide d'un algorithme d'ordonnancement (RankSVM, RankBoost, ...)

Sortie : Fonction d'ordonnancement adapté au groupe considéré

Sortie : Groupes de requêtes et fonctions d'ordonnancement spécifiques à chacun.

Entité 2 (Système en ligne).

Entrée : Nouvelle requête q et documents associés

1. Détection de la catégorie / du groupe de la requête
2. Sélection de la fonction d'ordonnancement adaptée
3. Détermination du classement à l'aide de la fonction d'ordonnancement

Sortie : Liste ordonnée des documents pour la requête

FIGURE 4.1 – Système d'ordonnancement adaptatif sans sélection de variables

Dans le cas d'approches *filter* et *wrapper*, la sélection de variables constitue une étape de pré-traitement. L'apprentissage des fonctions s'effectue en deux étapes ; d'abord la détermination du sous-ensemble de caractéristiques pertinentes par l'utilisation de la méthode de sélection, puis l'apprentissage de la fonction d'ordonnancement n'utilisant que les caractéristiques sélectionnées.

Dans le cas d'approches *embedded*, la sélection de variables est incorporée à l'algorithme d'apprentissage d'ordonnancement et une seule étape est nécessaire pour apprendre une fonction spécifique. L'algorithme d'apprentissage d'ordonnancement est dit parcimonieux, puisqu'il réduit le nombre de caractéristiques utilisées par les fonctions d'ordonnancement.

Les fonctions obtenues en utilisant la sélection de variables sont spécifiques à un groupe ou une catégorie de requêtes, non seulement car elles affectent des poids différents aux variables, mais aussi car elles n'utilisent pas les mêmes variables pour tous les groupes.

Ces fonctions sont ensuite utilisées au sein du système d'apprentissage en ligne pour classer les documents vis-à-vis des requêtes des utilisateurs. Considérant une requête soumise, le système en ligne détermine la catégorie du besoin exprimé. La fonction d'ordonnancement correspondante est alors sélectionnée parmi l'ensemble des fonctions apprises. Elle est utilisée pour ordonner les résultats de recherche qui sont restitués et présentés à l'utilisateur.

Le système ainsi formalisé constitue une des contributions de nos travaux. Une contribution majeure de nos travaux est la proposition de méthodes de sélection de variables. Nous avons ainsi proposé plusieurs algorithmes de sélection que nous avons évalués sur des données académiques et industrielles. Ces algorithmes peuvent être utilisés dans le cadre du système adaptatif. Une évaluation dans un cadre d'exploitation commercial est en cours. Les résultats sont attendus pour la fin d'année 2013. Dans la suite de ce mémoire, nous nous concentrons sur la présentation et l'évaluation des algorithmes que nous proposons. Dans la section suivante, nous présentons les méthodes de sélection de variables existantes et nous positionnons nos travaux.

4.4 Algorithmes de sélection de variables en apprentissage d'ordonnancement : état de l'art et positionnement

Les méthodes de sélection de variables en apprentissage d'ordonnancement peuvent être regroupées en suivant les trois principales approches introduites à la section 4.2.1. Dans la suite de cette section, nous présentons successivement les algorithmes des approches de type *filter*, *wrapper* et *embedded* dédiées à l'apprentissage d'ordonnancement.

Bloc 2 (Outil d'apprentissage pour l'ordonnancement avec sélection).

Entrée : Sous-ensemble du jeu de données d'apprentissage correspondant à un groupe de requêtes

Cas 1 (Utilisation de méthodes *filter* ou *wrapper*).

Etape 1 Sélection du sous-ensemble caractéristiques pertinentes

Etape 2 Apprentissage d'une fonction d'ordonnancement n'utilisant que le sous-ensemble de caractéristiques à l'aide d'un algorithme (RankSVM, RankBoost, ...)

Cas 2 (Utilisation de méthodes *embedded*).

Sélection des caractéristiques pertinentes et apprentissage de la fonction d'ordonnancement *simultanés.*)

Proposition 1 (Algorithmes de repondération en norme ℓ_2).

Proposition de trois algorithmes de sélection de variables, avec repondération de la norme ℓ_2 et utilisation d'un seuil de nullité des variables (*cf.* chapitre 6).

Proposition 2 (Algorithmes de repondération en norme ℓ_1).

Proposition d'un algorithme proximal de résolution en norme ℓ_1 et d'un algorithme de repondération de la norme ℓ_2 pour l'approximation des normes non convexes (*cf.* chapitre 7).

Sortie : Fonction d'ordonnancement spécifique à un groupe de requêtes, n'utilisant qu'un sous-ensemble des caractéristiques.

FIGURE 4.2 – Module d'apprentissage d'ordonnancement hors ligne incorporant la sélection de variables. L'outil de classification des requêtes et le système en ligne sont identiques à ceux de la figure 4.1

4.4.1 Algorithmes de l'approche par filtre (*filter*)

A notre connaissance, l'algorithme Greedy Search for Feature Selection (GAS) proposé par Geng *et al.* [Geng 2007] constitue la première proposition d'une approche de sélection de variables dédiée à l'apprentissage d'ordonnancement. Pour cette raison, nous la décrivons de façon plus détaillée que les autres approches introduites ici.

Il s'agit d'une approche de type *filter* basée sur l'utilisation d'un score d'importance et d'un score de similarité entre variables pour effectuer la sélection. L'algorithme se décompose en trois étapes :

Etape 1 *Calcul du score d'importance pour chaque variable*
Pour chaque caractéristique i, les observations sont ordonnées suivant les valeurs prises par i. Le classement π_i ainsi obtenu est évalué soit à l'aide d'une mesure de RI comme la MAP ou le NDG, soit à l'aide d'une fonction d'erreur. C'est ce résultat d'évaluation qui servira de score d'importance pour la caractéristique considérée.

Etape 2 *Calcul du score de similarité pour chaque couple de variables*
Pour chaque couple de caractéristiques (i, j) et chaque requête q, les deux classements π_i et π_j sont comparés via le τ de Kendall, tel que

$$\tau_q(\pi_i, \pi_j) = \frac{\#\{(d_s, d_t) \in \mathcal{D} | d_t \succ_{\pi_i} d_s \text{ and } d_t \succ_{\pi_j} d_s\}}{\#\{(d_s, d_t) \in \mathcal{D}\}} \tag{4.1}$$

où $\mathcal{D}$ est l'ensemble des documents du jeu de données associés à la requête et $d_t \succ_{\pi_i} d_s$ indique que le rang du document d_t est supérieur à celui de d_s lorsque l'on considère le classement π_i. Cela signifie que d_t apparaît avant d_s dans la liste de résultats, il est préféré à ce dernier. La valeur du τ est ainsi utilisée comme mesure de similarité entre les deux caractéristiques.

Etape 3 *Résolution d'un problème d'optimisation sous contraintes via un algorithme "Greedy"*
Le problème est alors de rechercher le sous-ensemble de caractéristiques pour lequel la somme des scores d'importance des caractéristiques est maximale et la somme des scores de similarité des caractéristiques est minimale. Les auteurs proposent de résoudre le problème d'optimisation sous contraintes suivant :

$$\begin{cases} \max\limits_{\omega_i, e_{i,j}} \sum\limits_i \omega_i x_i - c \sum\limits_i \sum\limits_{j \neq i} e_{i,j} x_i x_j \\ \text{sous la contrainte } \sum\limits_i x_i = t \\ \text{et } x_i \in \{0, 1\},\ i = 1, \ldots, m \end{cases} \tag{4.2}$$

où x_i indique si la caractéristique i est considérée dans le modèle ou non, ω_i est le score d'importance de i et $e_{i,j} = e_{j,i}$ est le score de similarité entre i et j. Le problème est résolu à l'aide d'un algorithme de type *Greedy*.

L'algorithme GAS a été évalué par les auteurs sur les jeux de données Ohsumed

et *Gov*, en utilisant les algorithmes RankSVM et RankNet pour l'apprentissage des fonctions d'ordonnancement. Les expériences menées ont permis de montrer que la MAP et le NDCG@10 restaient stables voire augmentaient lorsque le nombre de caractéristiques des modèles diminuait. Néanmoins, les performances obtenues variaient suivant le jeu de données considéré.

L'algorithme GAS a inspiré d'autres travaux de l'approche *filter*. Hua *et al.* [Hua 2010] ont proposé une approche nommée Hierarchical Feature Selection for Ranking basée sur la classification non supervisée et une méthode de délégation pour effectuer la sélection de variables. Leur algorithme se décompose en deux étapes. Dans un premier temps, la mesure de similarité entre variables présentée dans GAS et une mesure de qualité définie par les auteurs sont utilisées pour créer des groupes de caractéristiques, à l'aide d'un algorithme des k-moyennes. Dans un second temps, pour chaque groupe de caractéristiques, les auteurs calculent le score d'importance de chaque variable de façon similaire à GAS. La caractéristique qui obtient le meilleur score est déclarée représentative du groupe puis est sélectionnée. Le nombre total de caractéristiques est ainsi égal au nombre de groupes formés (k). Le modèle d'ordonnancement final est alors appris en utilisant seulement ces k caractéristiques. La méthode a été évaluée sur la collection LETOR 4. Vingt-six caractéristiques ont été sélectionnées sur le jeu de données MQ2007 et utilisées pour l'apprentissage sur les jeux de données MQ2007 et MQ2008. Les expérimentations montrent que les valeurs de MAP et de NDCG sont comparables voire meilleures pour les modèles de dimension réduite.

Parmi les travaux de l'approche *filter* n'utilisant pas les scores d'importance et de similarité définis dans GAS, nous pouvons citer RankFilter [Yu 2009]. Il s'agit d'une méthode basée sur l'algorithme Relief, développé pour la discrimination multi-classes (par exemple non pertinent *vs* pertinent) en 1992 par Kira et Rendell [Kira 1992]. L'algorithme Relief affecte initialement à chaque caractéristique un score de pertinence qui indique qu'elle est pertinente pour la modélisation du problème. Puis, à chaque étape, une observation est aléatoirement sélectionnée dans le jeu de données. Elle est alors comparée à l'observation la plus proche appartenant à la même classe de pertinence (appelée *Near Hit* et notée NH) et à l'observation la plus proche appartenant à la classe opposée (appelée *Near Miss* et notée NM). Le poids de pertinence $\mathbf{w}_i$ de chaque caractéristique i est alors mis à jour de la façon suivante :

$$\mathbf{w}_i = \mathbf{w}_i - \text{diff}(\mathbf{x}_i, NH_i)^2 + \text{diff}(\mathbf{x}_i, NM_i)^2 \tag{4.3}$$

où $\text{diff}(x_i, x_j) = \mathbb{1}_{\{\mathbf{x}_i \neq \mathbf{x}_j\}}$. Le processus est répété jusqu'à ce que toutes les variables pertinentes aient été sélectionnées. Yu *et al.* [Yu 2009] ont directement adapté l'algorithme Relief pour l'utiliser dans le cadre de l'apprentissage d'ordonnancement pour des données (couples requête-document) étiquetées avec des classes de pertinence (non pertinent, peu pertinent, très pertinent). Les auteurs montrent que les

valeurs de MAP et de NDCG sont meilleures en utilisant cette approche comparativement à l'approche GAS. Par ailleurs, la méthode peut être utilisée sur de grands volumes de données.

Parallèlement au développement d'approches *filter*, plusieurs travaux se sont intéressés à l'utilisation d'approches *wrapper* pour la sélection de variables en apprentissage d'ordonnancement. Nous présentons ces méthodes dans la section suivante.

4.4.2 Algorithmes de l'approche encapsulante (*wrapper*)

Certaines méthodes de l'approche *wrapper* sont en réalité des variations des méthodes *filter* présentées au paragraphe précédent. Hua *et al.* [Hua 2010] ont proposé une variante *wrapper* de leur algorithme Hierarchical Feature Selection for Ranking. La méthodologie de création des groupes de requêtes est identique à l'approche présentée plus haut, seul le choix de la caractéristique représentative diffère. Plutôt que d'utiliser le score d'importance défini par GAS, les auteurs proposent d'apprendre, pour chaque groupe de variables, un modèle qui n'utilise que les variables de ce groupe. La variable associée au plus grand poids dans le modèle est alors choisie comme représentative. Yu *et al.* ont également proposé une variante *wrapper* de leur algorithme basé sur la méthode Relief [Yu 2009] et nommée RankWrapper. Tandis que RankFilter considère des classes de pertinence comme jugement de pertinence de chaque document, RankWrapper considère un ordre de pertinence relatif entre documents, ce qui constitue la différence principale entre les deux algorithmes.

D'autres travaux proposent d'adapter des algorithmes existants. Dang et Croft [Dang 2010] considèrent l'algorithme Best First Search [Kohavi 1997] pour effectuer la sélection de variables. Considérant $\mathbb{F}$ l'ensemble des caractéristiques, la première étape est de créer k partitions disjointes $\mathbb{F}_1, \ldots, \mathbb{F}_k$ de $\mathbb{F}$ de taille maximale s. Pour cela, les auteurs créent un graphe dont chaque sommet ou noeud est un sous-ensemble des caractéristiques de taille maximale s . Une arête lie deux noeuds s'il est possible de passer d'un noeud à l'autre en ajoutant ou en supprimant exactement une variable du noeud. Un algorithme de Best First Search est alors utilisé pour choisir le meilleur sous-ensemble de variables. Un premier noeud est sélectionné de façon aléatoire, un modèle est appris à l'aide d'un algorithme d'ordonnancement et sa performance est évaluée à l'aide d'une certaine mesure.
L'algorithme répète le processus sur le noeud voisin et ainsi de suite jusqu'à ce que, soit la mesure se stabilise, soit aucun ensemble ne permette d'améliorer la performance. Le sous-ensemble final est ainsi sélectionné. Un nouveau graphe contenant les variables restantes est créé et le processus est itéré jusqu'à ce que toutes les partitions soient créées.
Une fois les partitions obtenues, la deuxième étape consiste à définir le modèle appris sur chaque sous-ensemble comme une nouvelle caractéristique. Ainsi, k nouvelles caractéristiques sont définies. Elles seront utilisées pour l'apprentissage du modèle final, qui est effectué à l'aide d'algorithmes de l'état de l'art comme Rank-

Net, RankBoost ou AdaRank. Les auteurs ont montré que la qualité de prédiction obtenue avec le modèle réduit est équivalente à celle du modèle utilisant toutes les variables, tout en utilisant de 10 % à 44 % de caractéristiques en moins, sur les jeux de données MQ2007 et MQ2008, en considérant le NDCG@5.

Pahikkala *et al.* proposent l'algorithme Greedy RankRLS [Pahikkala 2010], qui est une adaptation de l'algorithme d'apprentissage d'ordonnancement RankRLS présenté dans [Pahikkala 2009]. L'idée générale est de créer différents ensembles de caractéristiques et d'apprendre pour chacun un modèle d'ordonnancement à l'aide de l'algorithme RankRLS. Le sous-ensemble de caractéristiques qui obtient la meilleure qualité de prédiction est alors utilisé pour apprendre le modèle final. Les expérimentations effectuées sur la collection LETOR 4 ont montré que les valeurs de MAP et NDCG@10 des méthodes proposées sont équivalentes à celles obtenues par les méthodes de référence, tout en utilisant moins de caractéristiques. D'après les résultats présentés, les modèles optimaux utilisent environ 32 % des caractéristiques sur MQ2007 et moins de 10 % sur MQ2008, pour une qualité d'ordonnancement équivalente aux approches de l'état de l'art qui n'effectuent pas de sélection. Il est difficile de comparer cette approche à la précédente, car les mesures d'évaluation utilisées diffèrent.

Enfin, d'autres méthodes combinent des procédures *wrapper* existantes, comme la méthode de sélection *forward* et *backward*, à des scores de similarités et d'importance inspiré de GAS pour effectuer la sélection de variables. La méthode de sélection *forward* part d'un modèle initial vide et ajoute les variables une à une. La variable à ajouter est celle qui améliore le plus la qualité de prédiction selon la mesure considérée. Lorsque l'ajout de nouvelles variables ne permet plus d'améliorer le modèle, l'algorithme s'arrête. La méthode *backward* fonctionne de façon inversée. Elle considère un modèle initial contenant toutes les caractéristiques et enlève tour à tour la variable la moins utile au modèle. Les caractéristiques sont retirées jusqu'à ce que leur suppression dégrade le modèle.

Pan *et al.* [Pan 2009, Pan 2011] proposent ainsi de combiner l'utilisation de boosting d'arbres de décision binaires, d'une procédure d'élimination *backward* et un score d'importance basé sur le NDCG. En pratique, les auteurs créent différents sous-ensembles de caractéristiques. Puis, ils apprennent des modèles à l'aide d'un algorithme de boosting d'arbres de décision binaires. Enfin ils suppriment les caractéristiques suivant les valeurs de NDCG@5 obtenues, qui est ici défini comme un score d'importance des variables. Le processus d'apprentissage des modèles et de suppression de caractéristiques est itératif. Les auteurs comparent la performance de cette approche à une méthode *Greedy* qui optimise un score de similarité entre variables et un score d'importance. Ils montrent que l'approche *backward* donne de meilleurs résultats. Ils constatent que la méthode offre des résultats comparables à l'état de l'art tout en conservant uniquement 7 % des caractéristiques initiales du jeu de données. Cet algorithme a été évalué sur des données issues de Wikipédia et non sur LETOR, il n'est donc pas possible de le comparer aux méthodes précédentes

sur la base des articles publiés.

Enfin, Lai *et al.* [Lai 2011] combinent un score de similarité entre variables et des méthodes de sélection *backward* et *forward* pour sélectionner les caractéristiques les plus pertinentes. Le score de similarité entre deux caractéristiques i et j est défini comme le coefficient de corrélation de Pearson entre les listes de résultats ordonnées suivant les valeurs de i et les valeurs de j.

Plus récemment, des méthodes de sélection de variables de type *embedded* ont été proposées. Nous les présentons dans la section suivante.

4.4.3 Algorithmes de l'approche embarquée (*embedded*)

Contrairement aux approches *filter* et *wrapper*, les méthodes *embedded* effectuent la sélection de variables et l'apprentissage des fonctions d'ordonnancement de façon simultanée. Ces méthodes promeuvent la parcimonie dans les modèles en modifiant directement le problème d'optimisation résolu par l'algorithme. En pratique, un terme de régularisation induisant de la parcimonie est introduit dans la formulation du problème. Ce procédé est détaillé au chapitre 5 dans le cas des SVM.

Bien que ces méthodes soient largement utilisées en apprentissage, elles ont été relativement peu étudiées dans le contexte de la sélection de variables en apprentissage d'ordonnancement. Sun *et al.* [Sun 2009] ont proposé l'algorithme RSRank qui utilise un terme de régularisation en norme ℓ_1 pour induire de la parcimonie. Cette méthode permet d'optimiser directement le NDCG. Le problème d'optimisation est résolu via une méthode de descente de gradient. Les expérimentations menées sur les jeux de données Ohsumed et TD2003 ont montré que la méthode ne conservait qu'environ un tiers des caractéristiques en dégradant les résultats d'au plus 3 %.

L'utilisation de la norme ℓ_1 a également été proposée pour introduire de la parcimonie dans le cas d'algorithmes basés sur les SVM. Lai *et al.* [Lai 2013a] ont ainsi développé un algorithme nommé FenchelRank pour la résolution des SVM parcimonieux en apprentissage d'ordonnancement de préférences. Le problème d'optimisation est résolu de façon itérative. Les expérimentations effectuées sur les collections LETOR ont montré que l'algorithme obtient des taux de parcimonie très faible, avec de 19 à 50 % de variables conservées dans les modèles. Par ailleurs les valeurs de MAP et NDCG sont comparables à celles des algorithmes non parcimonieux de référence.

Enfin, Lai *et al.* ont récemment proposé un nouvel algorithme basé sur les SVM parcimonieux et nommé FSMRank [Lai 2013b]. Cette méthode résout un problème d'optimisation jointe qui introduit de la parcimonie tout en minimisant l'erreur de prédiction. Une approche de gradient accéléré est utilisée pour la résolution. L'algorithme proposé présente ainsi l'avantage de converger très rapidement. Par ailleurs, les auteurs montrent que la qualité de prédiction de FSMRank est globalement meilleure que celle obtenue avec GAS sur les jeux de données de référence.

Conclusion

Dans ce chapitre, nous avons motivé l'utilisation de méthodes de sélection de variables en apprentissage d'ordonnancement. Nous avons proposé un système d'ordonnancement utilisant la sélection de variables afin d'apprendre des fonctions spécifiques à chaque groupe de requêtes. Nous avons dressé un état de l'art des approches de sélection existantes et susceptibles d'être intégrées dans ce système.

Les approches *embedded* nous paraissent prometteuses. L'utilisation de régularisations parcimonieuses permet de réduire le nombre de caractéristiques des modèles et de conserver une bonne qualité de prédiction. L'apprentissage des fonctions d'ordonnancement et la sélection de variables se font simultanément, sans poser d'hypothèses supplémentaires sur la similarité ou l'importance des variables. Ce dernier point constitue à notre sens un avantage majeur de ces méthodes. Nous avons donc choisi de considérer ce type d'approches.

Dans la suite de cette partie, nous étudions l'utilisation des méthodes *embedded* pour la sélection de variables en apprentissage d'ordonnancement. Plus particulièrement, nous nous intéressons à l'utilisation de SVM parcimonieux. Nous nous concentrons sur l'utilisation de normes non convexes qui sont généralement plus parcimonieuses que la norme ℓ_1 utilisée par les algorithmes de l'état de l'art. L'objectif est de proposer des méthodes capables de supprimer un très grand nombre de caractéristiques, sans dégrader les qualités de prédiction, dans un temps raisonnable (de l'ordre de quelques secondes ou dizaines de secondes sur un ordinateur personnel).

Nous présentons les SVM au chapitre 5 et les algorithmes de sélection de variables développés aux chapitres 6 et 7. Les algorithmes du chapitre 6 sont basés sur des techniques de repondération en norme ℓ_2 avec seuillage pour l'approximation des problèmes en norme ℓ_1 et ℓ_0. Les approches du chapitre 7 sont basées sur un algorithme proximal en norme ℓ_1 et des techniques de repondération de la norme ℓ_1 pour l'approximation de normes parcimonieuses non convexes. Contrairement aux méthodes proposées au chapitre 6, elles ne nécessitent pas l'utilisation d'un seuil de nullité des variables et sont "naturellement parcimonieuses".

Chapitre 5

Support Vector Machine pour la sélection de variables : principe, formalisation et justification

Sommaire

5.1 Des SVM en classification aux SVM pour l'apprentissage d'ordonnancement : principe et formalisation

5.1.1 Principe général des SVM

Les Machines à Support de Vecteurs ou Séparateurs à Vaste Marge (Support Vector Machines en anglais, SVM) constituent une classe de techniques d'apprentissage supervisé proposées pour la résolution de problèmes de discrimination binaire. Ils sont basés sur les travaux de Vapnik [Vapnik 1995] sur l'apprentissage statistique. Les SVM ont également été étendus à la régression [Burges 1997] et à l'apprentissage multi-classes [Weston 1999, Crammer 2002]. Dans cette partie, nous présentons les SVM linéaires dans le cadre de la discrimination d'une variable dichotomique binaire. Les SVM non linéaires à noyaux ne sont pas abordés dans cette thèse. Le lecteur intéressé par ces méthodes peut se référer à l'ouvrage de référence de Schölkopf et Smola [Scholkopf 2001].

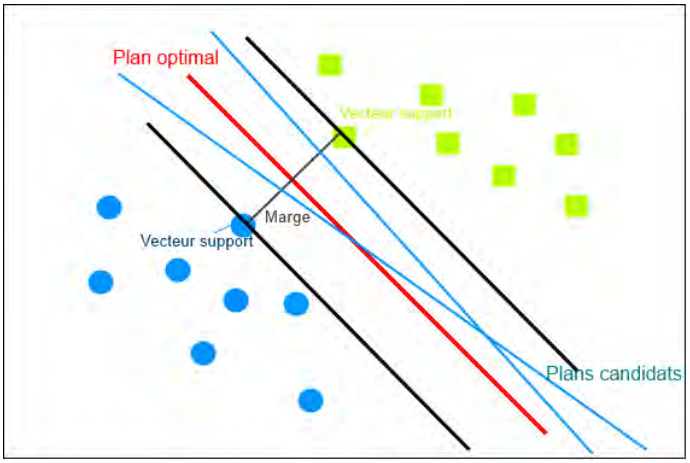


FIGURE 5.1 – Principe des SVM linéairement séparables. Les points colorés (ronds et carrés) représentent les deux classes à séparer. L'hyperplan séparateur optimal est représenté en rouge, tandis que d'autres hyperplans non optimaux sont indiqués en bleu.

Cas linéairement séparable

Considérons $[\mathbf{x}^{(1)} \dots \mathbf{x}^{(d)}]$ les variables prédictives, $\mathbf{x}_i = [\mathbf{x}_i^{(1)} \dots \mathbf{x}_i^{(d)}] \in \mathbb{R}^d$ la représentation de l'observation i dans l'espace des variables, $y \in \{-1, +1\}$ la variable dichotomique à prédire et un problème de discrimination linéairement séparable. Nous supposons l'existence de l'échantillon statistique $\{(\mathbf{x}_i, y_i)\}_{i=1,\dots,n}$ de taille n et de loi de probabilité inconnue. Le problème de discrimination linéaire par la méthode des SVM consiste à déterminer l'hyperplan séparateur de marge maximale, c'est-à-dire la fonction de décision de la forme $f(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$ où $\mathbf{w} \in \mathbb{R}^d$ et $b \in \mathbb{R}$, qui sépare de façon parfaite les deux classes (*cf.* figure 5.1). La recherche de cet hyperplan revient à résoudre un problème d'optimisation linéaire sous contrainte, appelé formulation *primale* des SVM linéairement séparables (*cf.* définition 13).

Définition 13 (Formulation primale des SVM linéairement séparables)**.**

Soient $\{(\mathbf{x}_i, y_i)\}_{i=1,\dots,n}$ un échantillon statistique de taille n et de loi de probabilité inconnue, $y_i \in \{-1, 1\}$ la classe à prédire, $\mathbf{w}$ le vecteur des poids, $b \in \mathbb{R}$ un biais, alors la formulation primale des SVM linéairement séparables est donnée par le problème d'optimisation sous contrainte suivant :

$$\begin{cases} \min\limits_{\mathbf{w},b} \dfrac{1}{2}\|\mathbf{w}\|_2^2 \\ \text{avec } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \ \forall i = 1, \dots, n \end{cases}$$

Le problème d'optimisation peut être résolu directement via la formulation primale. Néanmoins, la formulation duale lui est souvent préférée, d'une part, car elle permet d'écrire le problème sous une forme quadratique souvent plus facile à résoudre et d'autre part, car elle permet l'introduction de noyaux et donc la résolution de problèmes non linéaires. Le passage à la formulation duale s'effectue en écrivant le lagrangien du problème primal ainsi que les conditions de Karush, Kuhn et Tucker [Nocedal 2006]. L'objectif est alors de maximiser le lagrangien des SVM linéairement séparables, noté $\mathcal{L}(.,.)$. Celui-ci s'exprime de la façon suivante :

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|_2^2 - \sum_{i=1}^{n} \alpha_i (y_i(\mathbf{w}^\top \mathbf{x}_i + b) - 1) \tag{5.1}$$

où α est le vecteur des α_i multiplicateurs de Lagrange associés aux contraintes. Le problème dual peut alors être caractérisé à l'aide des cinq conditions de Karush, Kuhn et Tucker (conditions de stationnarité 5.2 et 5.3, condition de complémentarité 5.4, conditions d'admissibilité primale 5.5 et duale 5.6), où $(\mathbf{w}^*, b^*)$ est la solution optimale du problème et α^* les multiplicateurs de Lagrange correspondants :

$$\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}^*, b^*, \alpha^*) = 0 \Leftrightarrow \mathbf{w}^* - \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i = 0 \tag{5.2}$$

$$\frac{\partial \mathcal{L}(\mathbf{w}^*, b^*, \alpha^*)}{\partial b} = 0 \Leftrightarrow \sum_{i=1}^{n} \alpha_i^* y_i = 0 \tag{5.3}$$

$$\alpha_i^* (y_i(\mathbf{w}^\top \mathbf{x}_i + b^*) - 1) = 0 \ \forall i = 1, \ldots, n \tag{5.4}$$

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b^*) \geq 1 \ \forall i = 1, \ldots, n \tag{5.5}$$

$$\alpha_i^* \geq 0 \ \forall i = 1, \ldots, n \tag{5.6}$$

Les conditions de complémentarité (5.4) et d'admissibilité (5.5 et 5.6) permettent de définir les contraintes actives. Celles-ci correspondent aux coefficients de Lagrange strictement positifs c'est-à-dire pour lesquelles la relation $y_i(\mathbf{w}^\top \mathbf{x}_i + b^*) = 1$ est vérifiée. Les observations $\mathbf{x}_i$ associées sont appelées vecteurs supports. Les contraintes pour lesquelles les coefficients α_i sont nuls, c'est-à-dire qui vérifient la relation $y_i(\mathbf{w}^\top \mathbf{x}_i + b^*) > 1$, sont appelées des contraintes inactives. Résoudre la formulation primale des SVM équivaut alors à maximiser le lagrangien du problème dual, sous la contrainte que les solutions vérifient les conditions 5.2, 5.3 et 5.6. Cela équivaut à écrire le problème d'optimisation sous la forme suivante :

$$\begin{cases} \max\limits_{\mathbf{w},b} \dfrac{1}{2}\|\mathbf{w}\|_2^2 - \sum\limits_{i=1}^{n} \alpha_i (y_i(\mathbf{w}^\top \mathbf{x}_i + b) - 1) \\ \text{avec } \mathbf{w}^* - \sum\limits_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i = 0, \sum\limits_{i=1}^{n} \alpha_i^* y_i = 0 \text{ et } \alpha_i = 0 \ \forall i = 1, \ldots, n \end{cases} \tag{5.7}$$

En utilisant la condition de stationnarité (5.2), nous posons $\mathbf{w} = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x}_i = 0$. En remplaçant $\mathbf{w}$ par cette expression dans chacun des termes de l'équation précédente, nous pouvons réécrire le problème uniquement à partir de α_i, y_i et $\mathbf{x}_i$. Le problème obtenu s'appelle la formulation duale des SVM séparables. Elle est présentée à la définition 14.

Définition 14 (Formulation duale des SVM linéairement séparables)**.**

Soient $\{(\mathbf{x}_i, y_i)\}_{i=1,\dots,n}$ un échantillon statistique de taille n et de loi de probabilité inconnue, $y_i \in \{-1, 1\}$ la classe à prédire et α les multiplicateurs de Lagrange, alors la formulation duale des SVM linéairement séparables est donnée par le problème d'optimisation sous contrainte suivant :

$$\begin{cases} \max\limits_{\alpha} \dfrac{1}{2} \displaystyle\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_j \alpha_i y_i y_j \mathbf{x}_j^\top \mathbf{x}_i - \sum_{i=1}^{n} \alpha_i \\ \text{avec } \displaystyle\sum_{i=1}^{n} \alpha_i y_i = 0 \text{ et } \alpha_i \geq 0 \ \forall i = 1, \dots, n \end{cases}$$

Cas non linéairement séparable

Les formulations précédentes sont valables dans le cas de problèmes dont les observations sont effectivement séparables par un plan. Néanmoins, celles-ci peuvent être non séparables (*cf.* figure 5.2).

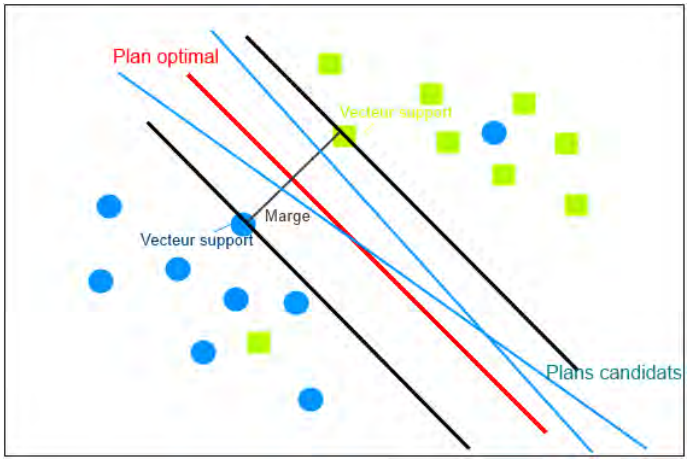


FIGURE 5.2 – Principe des SVM non linéairement séparables. Les points colorés représentent les deux classes à séparer. L'hyperplan séparateur optimal est représenté en rouge, tandis que d'autres hyperplans non optimaux sont indiqués en bleu. La présence d'un point bleu dans les carrés verts (resp. d'un carré vert dans les points bleus) rend les classes non linéairement séparables.

Dans ce cas, il est nécessaire de relâcher le problème en autorisant une erreur sur la discrimination, c'est-à-dire la présence d'observations mal classées. Cela est possible en introduisant des variables dites d'écart, notées ξ_i, qui vont permettre la présence d'erreurs tout en contrôlant l'éloignement entre l'hyperplan et les observations mal classées. La formulation primale (13) est ainsi modifiée en introduisant des variables d'erreur et un terme de pénalisation $C > 0$, comme indiqué dans l'équation 15. Le paramètre C permet de contrôler le compromis entre qualité d'ajustement et capacité de généralisation en agissant sur les erreurs autorisées.

Définition 15 (Formulation primale des SVM linéaires - Cas non séparable)**.**

Soient $\{(\mathbf{x}_i, y_i)\}_{i=1,\dots,n}$ un échantillon statistique de taille n et de loi de probabilité inconnue, $y_i \in \{-1, 1\}$ la classe à prédire, $\mathbf{w}$ le vecteur des poids, $b \in \mathbb{R}$ un biais, ξ_i, $i = 1, \dots, n$ les termes d'erreur et $C > 0$ un paramètre contrôlant les erreurs, alors la formulation primale des SVM non séparables est donnée par le problème d'optimisation sous contrainte suivant :

$$\begin{cases} \min\limits_{\mathbf{w},b} \dfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum\limits_{i=1}^{n} \xi_i \\ \text{avec } y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \ \forall i = 1, \dots, n \end{cases}$$

Les termes d'erreurs peuvent être ré-écrits pour ne faire intervenir que $y_i, \mathbf{w}, \mathbf{x}_i$ et b. Notamment, la régularisation Hinge $\xi_i = \max(0, y_i(\mathbf{w}^T\mathbf{x}_i + b))$ ou la régularisation Hinge au carré peuvent être utilisées. Dans cet état de l'art, nous avons considéré une régularisation Hinge au carré, ce qui revient à considérer le problème d'optimisation sans contrainte suivant :

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b))^2 \tag{5.8}$$

La formulation duale du problème 5.8 peut également être déduite en écrivant les conditions de Karush, Kuhn et Tucker. Le passage au cas non linéaire peut ensuite être effectué en introduisant des fonctions noyaux dans la formulation duale. En apprentissage d'ordonnancement, nous nous limitons à l'utilisation de SVM linéaires.

Dans la section suivante, nous présentons comment les SVM linéaires sont modifiés dans le cas de l'apprentissage d'ordonnancement par paire, appelé également apprentissage d'ordonnancement de préférences.

5.1.2 De la classification à l'apprentissage d'ordonnancement de préférences : formulation mathématique

Nous nous intéressons à l'utilisation des SVM dans le cadre de l'apprentissage d'ordonnancement et plus particulièrement, dans le contexte d'approches par paire.

Notre objectif est l'optimisation automatique de l'ordonnancement de paires de documents vis-à-vis de requêtes utilisateurs. Dans cette partie, nous montrons comment le problème d'apprentissage d'ordonnancement par paires peut être ramené à un problème de discrimination lorsque l'on considère les SVM. Nous partons du problème 16, proposé par Joachims [Joachims 2002] et montrons qu'il peut être ré-écrit sous la forme 5.8.

Définition 16 (Ordonnancement de préférences avec les SVM en RI)**.**

Soient :
- $\mathcal{R} = \{r_k\}_{k=1,\dots,R}$ l'ensemble des requêtes considérées, où R est le nombre total de requêtes,
- $\mathcal{D} = \{d_i\}_{i=1,\dots,N}$ l'ensemble des documents considérés, où N est le nombre total de documents,
- L'extracteur de caractéristiques ϕ qui à toute paire requête-document (r_k, d_i) associe un vecteur de caractéristiques $\phi(r_k, d_i) \in \mathbb{R}^d$ où d est le nombre de caractéristiques. $\phi(r_k, d_i)$ représente la paire dans l'espace des caractéristiques,
- $\mathcal{P}_{r_k} = \{(i,j)_{i,j=1,\dots,N} | d_i \succ_{r_k} d_j\}$ l'ensemble des couples d'indices $(i,j)_{i,j=1,\dots,N}$ tels que le document d_i est préféré au document d_j pour la requête r_k

Alors le problème d'apprentissage d'ordonnancement par paires avec les SVM est défini par le problème d'optimisation sous contraintes suivant :

$$\min_{\mathbf{w},\xi_{i,j,k}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i,j,k}\xi_{i,j,k} \tag{5.9}$$

sous les contraintes

$$\begin{cases} \forall (i,j) \in \mathcal{P}_{r_1}, \mathbf{w}^\top(\phi(r_1,d_i) - \phi(r_1,d_j)) > 1 - \xi_{i,j,1} \\ \vdots \\ \forall (i,j) \in \mathcal{P}_{r_R}, \mathbf{w}^\top(\phi(r_R,d_i) - \phi(r_R,d_j)) > 1 - \xi_{i,j,R} \end{cases} \tag{5.10}$$

et

$$\forall i, \forall j, \forall k, \xi_{i,j,k} \geq 0 \tag{5.11}$$

Les contraintes traduisent les préférences entre deux documents pour une même requête. Si le document d_i est préféré au document d_j pour la requête r_k, alors la valeur prédite pour d_i doit être supérieure à la valeur prédite pour d_j, soit $w^\top\phi(r_k,d_i) > w^\top\phi(r_k,d_j)_1 - \xi_{i,j,k}$. En faisant passer tous les termes contenant $\phi(.,.)$ à gauche de l'équation et en factorisant par $\mathbf{w}$, la formulation est équivalente à celle de la définition. En travaillant sur la ré-écriture de l'ensemble des préférences entre paires de documents, le problème 16 peut être ré-écrit sous la forme de la définition 15.

Soit $\mathcal{I}^k = \{i^k\}_{i=1,\ldots,N \text{ et}((i,.) \text{ ou } (.,i))\in\mathcal{P}_{r_k})}$ l'ensemble des indices des documents qui interviennent dans une relation de préférence pour la requête r_k. En posant

$$\mathbf{D} = [\{i \in \mathcal{I}^1\} \ldots \{i \in \mathcal{I}^R\}] \text{ , } \mathbf{D} \in \mathbb{N}^{\sum_{k=1}^{R} \text{card}\{\mathcal{I}^k\}}$$

$$\mathbf{R} = [\underbrace{1 \ldots 1}_{card\{\mathcal{I}^1\}} \cdots \underbrace{k \ldots k}_{card\{\mathcal{I}^k\}} \cdots \underbrace{R \ldots R}_{card\{\mathcal{I}^R\}}] \text{ , } \mathbf{R} \in \mathbb{N}^{\sum_{k=1}^{R} \text{card}\{\mathcal{I}^k\}}$$

et $n = \sum_{k=1}^{R} \text{card}\{\mathcal{I}^k\}$, nous pouvons définir l'ensemble $\mathcal{P}$ des relations de préférences (paires) de la façon suivante :

$$\mathcal{P} = \{(s,t)_{s=1,\ldots,n;t=1,\ldots,n} | (\mathbf{D}(s), \mathbf{D}(t)) \in \mathcal{P}_{r_{\mathbf{R}(s)=\mathbf{R}(t)}}\}$$

Chaque vecteur de caractéristiques $\phi(.,.)$ peut être écrit sous la forme $\phi(.,.) = \mathbf{x}_s, s = 1, \ldots, n$. Chaque préférence $p = (s,t) \in \mathcal{P}$ est alors représentée par le vecteur de caractéristiques $\tilde{\mathbf{x}}_p$ tel que $\tilde{\mathbf{x}}_p = \mathbf{x}_s - \mathbf{x}_t$. Soit P le nombre total de caractéristiques, le problème d'apprentissage d'ordonnancement par paires défini en 16 peut alors être ré-écrit sous la forme suivante :

$$\min_{w,\xi_p} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{p=1}^{P} \xi_p \tag{5.12}$$

sous les contraintes

$$\begin{cases} \tilde{\mathbf{x}}_p \mathbf{w} \geq 1 - \xi_p & \\ \xi_p \geq 0 & \forall p = 1, \ldots, P \end{cases} \tag{5.13}$$

Nous retrouvons donc bien le problème de discrimination dans le cas particulier ou $b = 0$ et $y_p = 1$ $\forall p$. En introduisant la régularisation Hinge au carré, le problème final à résoudre devient :

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{p=1}^{P} \max(0, 1 - \tilde{\mathbf{x}}_p^\top \mathbf{w})^2 \tag{5.14}$$

Ce dernier problème constitue notre reformulation de référence pour la suite de ces travaux. Des algorithmes d'apprentissage d'ordonnancement comme RankSVM-Primal et RankSVM-Dual permettent de résoudre efficacement ce type de problèmes. Dans la suite de nos travaux, nous nous concentrons sur le développement d'algorithmes de SVM parcimonieux, pour la sélection de variables en apprentissage d'ordonnancement. Nous présentons le principe des SVM parcimonieux dans la section suivante.

5.2 SVM parcimonieux pour la sélection de variables

De nombreux travaux se sont intéressés à l'utilisation de régularisations parcimonieuses pour la sélection de variables. Dans ce chapitre, nous présentons comment effectuer la sélection en introduisant des régularisations parcimonieuses. Nous utilisons les SVM comme méthode d'apprentissage. Dans un premier temps, nous ré-écrivons le problème d'apprentissage comme un problème de minimisation d'une fonction de perte et d'un terme de régularisation. Dans un second temps, nous expliquons comment adapter ce problème pour effectuer de la sélection de variables via des régularisations parcimonieuses. Nous présentons les régularisations convexes et non convexes qui peuvent être envisagées dans ce cadre. Une explication détaillée est disponible dans le livre de Hastie, Tibshirani et Friedman [Hastie 2003].

5.2.1 Définition du problème d'apprentissage avec régularisation

Les formulations utilisées dans cette partie sont issues de la communauté de l'apprentissage ou *Machine Learning* et plus précisément des travaux de Vapnik sur la théorie de l'apprentissage et le principe de minimisation du risque structurel [Vapnik 1995]. Ces travaux montrent que le problème d'apprentissage peut être défini de manière généralisée de la façon suivante :

$$\min_{\mathbf{w},b} \quad C\sum_{i=1}^{n} L(\mathbf{x}_i^\top \mathbf{w} + b, y_i) + \Omega(\mathbf{w}) \tag{5.15}$$

où C est une constante positive, $L(.,.)$ est une fonction de coût et $\Omega(.)$ est un terme de régularisation. La fonction de coût $L(.,.)$ est un terme mesurant l'erreur ou la divergence entre la valeur prédite par le modèle et la valeur de référence y_i. Le terme de régularisation $\Omega(.)$ contrôle la capacité de généralisation du modèle au travers de sa complexité. En pratique, différentes fonctions de perte peuvent être utilisées, suivant le cadre d'utilisation. En discrimination, les fonctions de perte Hinge, Hinge au carré et logistique sont généralement privilégiées. En régression, les fonctions de perte ℓ_2 ou de Hubert sont considérées. De façon similaire, différents termes de régularisation peuvent être utilisés, selon l'objectif ciblé. La norme ℓ_2, $\Omega(\mathbf{w}) = \|\mathbf{w}\|_2^2$ ou régularisation *ridge*, est la plus usitée dans le cadre général d'apprentissage. La norme ℓ_1, $\Omega(\mathbf{w}) = |\mathbf{w}|_1$, ou régularisation *Lasso* a été proposée par Tibshirani [Tibshirani 1996] pour sélectionner automatiquement les variables pertinentes. Le principe de la sélection de variables par utilisation de régularisations parcimonieuses est présenté dans la section suivante. Notons que le problème (5.14) est un cas particulier de (5.15) avec $b = 0$, $L(\tilde{\mathbf{x}}_i^\top \mathbf{w}, y_i) = \max(0, 1 - \tilde{\mathbf{x}}_p^\top \mathbf{w})^2$ et $\Omega(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_2^2$. Dans la suite de chapitre, nous considérons $b = 0$ sans perte de généralité.

5.2.2 Sélection de variables par utilisation de régularisations parcimonieuses

Apprendre le modèle le plus parcimonieux consiste à construire le modèle utilisant le moins de caractéristiques possibles. En pratique, dans le cas linéaire, il s'agit d'annuler le maximum de coefficients du modèle ou, de façon équivalente, de minimiser le nombre de coefficients non nuls. Une façon naturelle de résoudre ce problème est de considérer la norme ℓ_0, c'est-à-dire de poser $\Omega(\mathbf{w}) = \|\mathbf{w}\|_0$ et le problème suivant :

$$\min_{\mathbf{w}} \quad C \sum_{i=1}^{n} L(\tilde{\mathbf{x}}_i^\top \mathbf{w}, y_i) + \|\mathbf{w}\|_0 \tag{5.16}$$

où $\|\mathbf{w}\|_0 = \sum_{\mathbf{w}_i \neq 0} 1$. En effet, la norme ℓ_0 étant le nombre de coefficients non nuls de ce vecteur, minimiser cette norme revient à résoudre directement le problème de sélection. Néanmoins, la résolution n'est pas triviale. Le norme ℓ_0 présente trois inconvénients majeurs : elle n'est ni convexe, ni continue et n'est pas différentiable en zéro. Ainsi, le problème d'optimisation associé est de complexité NP. Il est donc particulièrement difficile à résoudre [Amaldi 1993].

Des méthodes existent pour l'approximation de la solution de ce problème, comme par exemple des approches par repondération [Weston 2003], pour lesquelles nous proposons une adaptation dans le cadre de l'apprentissage d'ordonnancement au chapitre 6. L'approche la plus populaire consiste à lui substituer un problème en norme ℓ_1, qui est alors vu comme une relaxation convexe du problème en norme ℓ_0. L'utilisation de la norme ℓ_1 à des fins de sélection de variables a été initialement proposée par Tibshirani [Tibshirani 1996].

Le problème à résoudre, dans le cadre généralisé, est alors le suivant :

$$\min_{\mathbf{w}} C \sum_{p=1}^{P} L(\tilde{\mathbf{x}}_p^\top \mathbf{w}, y_i) + \|\mathbf{w}\|_1 \tag{5.17}$$

où $\|\mathbf{w}\|_1 = \sum_{i=1}^{d} |\mathbf{w}_i|$. Le caractère parcimonieux de la norme ℓ_1 peut alors être observé en écrivant les équations 5.14 et 5.17 sous la forme du problème d'optimisation sous contraintes suivant :

$$\min_{\mathbf{w}} \sum_{p=1}^{P} \max(0, 1 - \tilde{\mathbf{x}}_p^\top \mathbf{w})^2 \text{ sous la contrainte } \|\mathbf{w}\|_1 < \lambda \tag{5.18}$$

où $\lambda = \frac{1}{C}$, C le paramètre de régularisation précédemment défini. Il suffit alors de comparer les contours des régions de faisabilité de la norme ℓ_2 et de la norme ℓ_1 pour comprendre le caractère parcimonieux de cette dernière (*cf.* figure 5.3). Nous observons ainsi que la norme ℓ_1, de contour plus "pointu", va avoir tendance à ramener les coefficients les plus faibles à 0 et à en fixer certains exactement à 0, introduisant ainsi la parcimonie. Comparativement à la norme ℓ_0, la norme ℓ_1 présente l'avantage d'être convexe, ce qui est une condition nécessaire à l'utilisation des algorithmes d'optimisation. Néanmoins, elle n'est pas différentiable en 0. Des

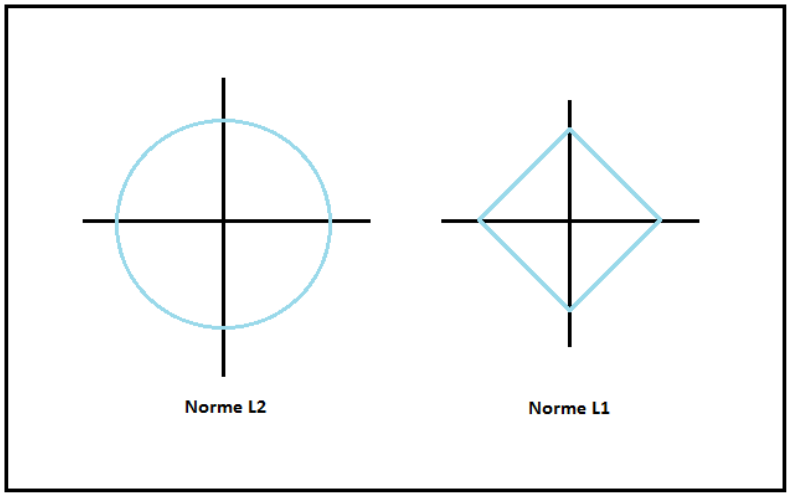


FIGURE 5.3 – Contours des régions de faisabilité pour les normes ℓ_2 et ℓ_1

méthodes spécifiques pour la résolution du problème d'optimisation ont ainsi été développées [Zhu 2003, Mangasarian 2006].

La norme ℓ_1 est un choix populaire pour introduire de la parcimonie dans les modèles, mais elle présente certains inconvénients. Zou [Zou 2006] a montré qu'elle pouvait être biaisée. Différentes méthodes et régularisations ont été proposées pour contrer ce problème. Une solution est d'utiliser une pénalité Lasso repondérée, qui affecte un poids à chaque élément du vecteur $\mathbf{w}$, tel que

$$\Omega(\mathbf{w}) = \sum_{i=1}^{d} \beta_i |\mathbf{w}_i| \tag{5.19}$$

avec $\beta_i > 0$. Zou a proposé des poids dans le contexte de la régression. D'autres méthodes utilisent cette approche pour intégrer des régularisations non convexes très parcimonieuses [Gasso 2009].

L'idée est alors d'exprimer le terme de régularisation de la façon suivante :

$$\Omega(\mathbf{w}) = \sum_{i=1}^{d} g(|\mathbf{w}_i|) \tag{5.20}$$

où $g(.)$ est une fonction non convexe. L'utilisation des fonctions non convexes assure de bonnes propriétés (parcimonie, qualité de prédiction, absence de biais). Le problème d'optimisation posé peut être résolu à l'aide de l'approche en norme ℓ_1 repondérée, comme l'ont montré Gasso *et al.* [Gasso 2009]. Les poids β_i sont alors définis tels que $\beta_i = g'(|\mathbf{w}_i^*|)$ où $g'(.)$ est la dérivée de la fonction convexe $g(.)$ et $\mathbf{w}_i^*$ est la solution trouvée à l'itération précédente. L'utilisation du schéma en norme ℓ_1 repondéré assure de bonnes propriétés en matière de temps d'exécution.

Plusieurs types de régularisations non convexes peuvent être utilisées dans ce contexte. La pseudo-norme $\ell_q, q < 1$ a ainsi été proposée par Fu [Fu 1998]. Elle

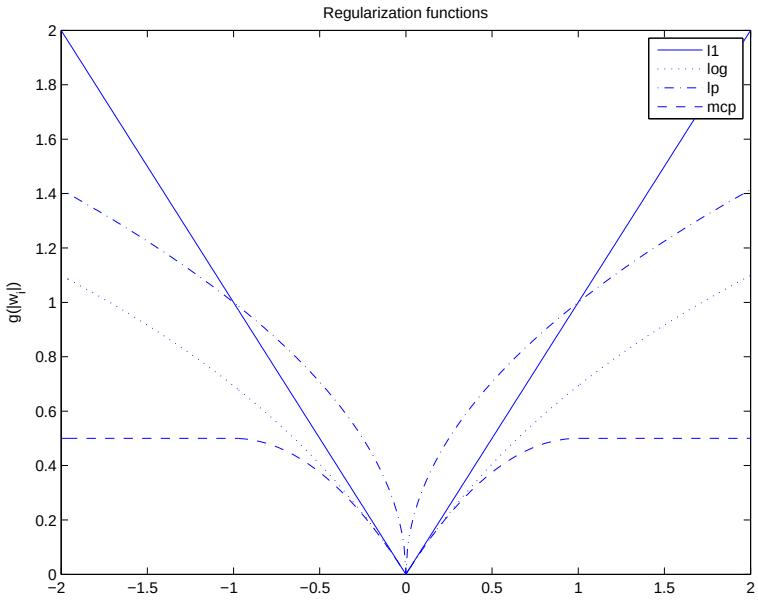


FIGURE 5.4 – Régularisations non convexes

est populaire dans de nombreux domaines, comme par exemple les applications de *compressed sensing* [Candes 2008]. Une approximation populaire de la norme ℓ_0 est la régularisation log proposée dans [Weston 2003]. Ces régularisations sont particulièrement efficaces pour introduire de la parcimonie. D'autres régularisations, comme les pénalités SCAD (Smoothly Clipped Absolute Deviation) [Fan 2001] ou MCP (Minimax Concave Penalty) [Zhang 2010] ont été proposées pour minimiser le biais de prédiction tout en introduisant de la parcimonie. Les expressions des régularisations non convexes sont présentées au tableau 5.1. Les régularisations sont tracées dans la figure 5.4.

Utiliser la norme ℓ_0 est le moyen le plus naturel de traiter le problème de sélection de variables. Nous souhaitons pouvoir évaluer son apport comparativement à des algorithmes utilisant la norme ℓ_1. Nous nous intéressons particulièrement à l'approximation des normes ℓ_0 et ℓ_1 par repondération de la norme ℓ_2 et nous proposons trois algorithmes de résolution dans ce cadre (*cf.* chapitre 6). Nous étudions l'utilisation de régularisations non convexes dans le cadre du schéma repondéré en norme ℓ_1. Nous proposons deux algorithmes de résolution (*cf.* chapitre 7). Les régularisations non convexes sont généralement plus parcimonieuses que la norme ℓ_1 et constituent de bonnes candidates pour la sélection de variables en apprentissage d'ordonnancement. A notre connaissance, l'utilisation de ces normes n'a pas été étudié dans le contexte de l'apprentissage d'ordonnancement. Nous proposons d'analyser l'apport des méthodes non convexes dans le cadre de la sélection de variables en apprentissage d'ordonnancement.

TABLE 5.1 – Expressions des régularisations parcimonieuses non convexes

Régularisation	**Expression de $g(\|\mathbf{w}_i\|)$**
ℓ_1	$\|\mathbf{w}_i\|$
$\ell_p, p < 1$	$\|\mathbf{w}_i\|^p$
log	$log(\varepsilon + \|\mathbf{w}_i\|)$
MCP	$\begin{cases} \lambda\|\mathbf{w}_i\| - \frac{\|\mathbf{w}_i\|^2}{2\gamma} \text{ si } \|\mathbf{w}_i\| \leq \gamma\lambda \\ \frac{\gamma\lambda^2}{2} \text{ si } \|\mathbf{w}_i\| > \gamma\lambda \end{cases}$

5.3 Justification de l'utilisation de SVM parcimonieux de l'approche par paire

Dans ces travaux, nous nous intéressons exclusivement aux méthodes de sélection de variables utilisant les SVM parcimonieux dans le cadre d'approches par paire, pour plusieurs raisons.

Tout d'abord, utiliser les SVM dans le cadre d'une approche par paire présente l'avantage de réduire le problème d'ordonnancement à un problème de discrimination, comme nous l'avons vu au début de ce chapitre. La différence entre discrimination et apprentissage d'ordonnancement est liée à la mesure utilisée pour choisir le modèle final. Dans le premier cas, nous cherchons à minimiser l'erreur de prédiction, tandis que dans le second, nous cherchons à maximiser une des mesures de performance comme la MAP ou le NDCG. De cette manière, les méthodes de sélection de variables proposées dans le cadre des SVM pour la classification peuvent être adaptées et évaluées plus facilement.

Ensuite, les algorithmes de l'approche par paire utilisant les SVM donnent généralement de bons résultats sur les jeux de données de référence comme LETOR 3 et 4 [Liu 2011]. En considérant l'étude de Liu [Liu 2011], nous observons que RankSVM obtient, pour l'ensemble des mesures considérées, un meilleur *winning number* que les autres algorithmes de l'approche par paires considérées. Néanmoins, comme nous l'avons mentionné au chapitre 2, cette comparaison doit être interprétée avec prudence, car tous les algorithmes ne sont pas considérés et aucun test de significativité n'est effectué.

Enfin, nous avons choisi de considérer des SVM parcimonieux car ces méthodes permettent de traiter le problème de la sélection de variables par la simple utilisation de normes. Ainsi, contrairement aux approches de types *filter* et *wrapper*, il n'est pas nécessaire de définir des hypothèses supplémentaires, comme l'importance de chaque variable ou la similarité des variables deux à deux. La sélection est réalisée grâce aux propriétés de parcimonie des normes uniquement. Par ailleurs, cette sélection est effectuée simultanément à la phase d'apprentissage du modèle et spécifiquement pour le classifieur, ce qui permet à la fois d'économiser une étape de calcul et de conserver les propriétés du classifieur pour le choix des variables. Les méthodes de types *embedded*, dont font partie les SVM parcimonieux, sont gé-

néralement considérées comme plus efficaces que les autres approches, notamment lorsque les temps de calcul sont considérés [Guyon 2003].

Nous proposons différents algorithmes basés sur l'approximation des normes ℓ_1 et ℓ_0 par repondération de la norme ℓ_2, présentés au chapitre 6. Ces algorithmes utilisent le schéma de repondération pour diminuer le poids des caractéristiques les moins pertinentes. Un seuil de nullité est alors utilisé pour retirer du modèle les caractéristiques associées à un poids trop faible. Nous proposons également deux algorithmes, un basé sur une approche proximale en norme ℓ_1 et un utilisant un schéma de repondération en norme ℓ_1 pour la résolution de problèmes avec régularisation non-convexes, présentés au chapitre 7. Comparativement aux méthodes de repondération de la norme ℓ_2, ces algorithmes sont naturellement parcimonieux et il n'est donc pas nécessaire de fixer un seuil de nullité des variables pour retirer celles-ci du modèle.

Chapitre 6

Proposition de méthodes de repondération pour la sélection de variables

Sommaire

Dans ce chapitre, nous présentons les techniques d'approximation des SVM parcimonieux par repondération en discrimination. Nous proposons une méthodologie d'adaptation de ces méthodes pour la sélection de variables en apprentissage d'ordonnancement. Nous évaluons les approches proposées sur des jeux de données de référence. Nous montrons que les approches considérées donnent d'excellents résultats tant du point de vue de la performance que de la qualité des modèles appris. Les algorithmes proposés et les résultats obtenus font l'objet d'un article soumis dans une conférence internationale [Laporte 2013b].

6.1 Repondération pour la résolution de problèmes parcimonieux

Nous nous intéressons à l'adaptation en apprentissage d'ordonnancement de méthodes et d'algorithmes de type IRLS (Iteratively Reweighted Least Squares). Les algorithmes IRLS constituent une famille de méthodes dont le principe consiste à substituer à un problème parcimonieux difficile à résoudre, une succession de problèmes d'optimisation de résolution plus aisée. La solution, obtenue par itérations successives de problèmes standards, est ainsi une approximation de la solution réelle du problème parcimonieux. Les méthodes existantes se sont surtout concentrées sur l'approximation des normes ℓ_0 et ℓ_1 par la norme ℓ_2 et de la norme ℓ_0 par la norme ℓ_1. Nous présentons une revue rapide de ces approches.

Weston [Weston 2003] a proposé un algorithme, nommé ℓ_2-AROM pour ℓ_2 Approximation to zeRO-norm Minimization, permettant la résolution de SVM parcimonieux en norme ℓ_0 par une succession de sous-problèmes de SVM en norme ℓ_2. Le principe général est détaillé dans l'algorithme 6.1.

Algorithme 6.1 Algorithme ℓ_2-AROM [Weston 2003]

Entrée : jeu de données d'apprentissage $\{(\mathbf{x}^j, \mathbf{y}^j)\}_{j=1}^m$
Sortie : Vecteur de poids $\mathbf{w}$
  Initialisation $\mathbf{v} = [1...1] \in \mathbb{R}^d$
  Tant que Non convergence **Faire**
    $\mathbf{w} \leftarrow$ SVMSolver$(\mathbf{v} * \mathbf{x}^j, y^j)$ %Solution des SVM en norme ℓ_2 dans le dual
    $\mathbf{v} \leftarrow \mathbf{v} * \mathbf{w}$
  Fin Tant que
  Retourner $\mathbf{w}$

D'autres travaux dans le domaine de la reconstruction de signaux se sont intéressés à la ré-écriture de la norme ℓ_0 faisant intervenir la norme ℓ_2. Ainsi, Gorodnitsky et Rao [Gorodnitsky 1997] ont proposé la ré-écriture'approximation suivante :

$$\|\mathbf{w}\|_0 = \sum_{i=1}^{d} \mathbb{1}_{\{\mathbf{w}_i \neq 0\}} = \sum_{i=1, \mathbf{w}_i \neq 0}^{d} 1 = \sum_{i=1, \mathbf{w}_i \neq 0}^{d} \frac{\mathbf{w}_i^2}{\mathbf{w}_i^2} \tag{6.1}$$

Résoudre le problème en norme ℓ_0 équivaut alors à résoudre une succession de problèmes en norme ℓ_2 pour lesquels chaque variable i est repondérée par $\mathbf{v}_i = \sqrt{\mathbf{w}_i^2} = |\mathbf{w}_i|$. Dans le même domaine d'application, Chartrand et Yin [Chartrand 2008] ont proposé l'approximation suivante :

$$\|\mathbf{w}\|_0 = \sum_{i=1, \mathbf{w}_i \neq 0}^{d} 1 \approx \sum_{i=1}^{d} \frac{\mathbf{w}_i^2}{\mathbf{w}_i^2 + \varepsilon} \text{ avec } \varepsilon > 0 \text{ petit} \tag{6.2}$$

Le terme ε est ici introduit comme terme de régularisation afin de s'affranchir de la contrainte $\mathbf{w}_i \neq 0$ dans l'équation 6.1. Zhang et Kingsbury [Zhang 2010] ont

également proposé une approximation similaire à 6.2 mais en posant le terme ε au carré :

$$\|\mathbf{w}\|_0 = \sum_{i=1,\mathbf{w}_i\neq 0}^{d} 1 \approx \sum_{i=1}^{d} \frac{\mathbf{w}_i^2}{\mathbf{w}_i^2+\varepsilon^2} \text{ avec } \varepsilon > 0 \text{ petit} \tag{6.3}$$

Ces deux approches conduisent à résoudre une succession de problèmes en norme ℓ_2 pour lesquels chaque variable i est repondérée respectivement par $\sqrt{\mathbf{w}_i^2+\varepsilon}$ ou $\sqrt{\mathbf{w}_i^2+\varepsilon^2}$.

D'autres travaux se sont concentrés sur l'approximation des problèmes parcimonieux en norme ℓ_1 par une succession de problèmes en norme ℓ_2. Ainsi, Figueirido [Figueiredo 2003] a proposé une modification de l'algorithme Expectation-Maximization (EM) pour l'approximation du problème en norme ℓ_1 par des problèmes en norme ℓ_2. L'algorithme est basé sur l'égalité suivante :

$$\forall \mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\|_1 = \sum_{i=1}^{d} |\mathbf{w}_i| = \sum_{i=1,\mathbf{w}_i\neq 0}^{d} \frac{\mathbf{w}_i^2}{|\mathbf{w}_i|} \tag{6.4}$$

A chaque itération, l'algorithme met ainsi à jour le vecteur $\mathbf{w}$ de la façon suivante :

$$\mathbf{w}^{(t+1)} = \arg\min_{w} \frac{1}{2} \sum_{i=1,\mathbf{w}_i\neq 0}^{d} \frac{\mathbf{w}_i^2}{|\mathbf{w}_i^{(t)}|} + C \sum_{j=1}^{n} \max(0, 1 - y_j \mathbf{x}_j^\top \mathbf{w})^2 \tag{6.5}$$

ce qui équivaut à résoudre un problème en norme ℓ_2 en repondérant chaque variable i par $\sqrt{|w_i^{(t)}|}$.

Kujala *et al.* ont proposé un algorithme de SVM parcimonieux en norme ℓ_1 nommé RW pour Re-Weighted [Kujala 2009], mais utilisant une re-pondération différente de la précédente. A chaque itération, chaque caractéristique i est ainsi pondérée par le poids $\mathbf{v}_i^{(t+1)}$ tel que $\mathbf{v}_i^{(t)} = \sqrt{\mathbf{w}_i^{(t)}\mathbf{v}_i^{(t)}}$. Contrairement à la pondération précédente, celle-ci conserve la mémoire des différents modèles appris au cours du temps. Les auteurs ont également montré que cette approche converge plus rapidement que la précédente.

Les propositions d'approximation de la norme ℓ_0 sont assez récentes. Une des proposition majeure est l'article de Candès et al. [Candes 2008] qui approche la norme ℓ_0 par la norme ℓ_1 repondérée. L'approximation utilisée est la suivante :

$$\|\mathbf{w}\|_0 = \sum_{i=1,\mathbf{w}_i\neq 0}^{d} 1 \approx \sum_{i=1}^{d} \frac{|\mathbf{w}_i|}{|\mathbf{w}_i|+\varepsilon} \text{ avec } \varepsilon > 0 \text{ petit} \tag{6.6}$$

A chaque itération, l'algorithme résout donc un problème en norme ℓ_1 où chaque variable i est pondéré par $\sqrt{|\mathbf{w}_i|+\varepsilon}$.

Concernant les repondérations en norme ℓ_2, il est important de noter que le problème en norme ℓ_2 n'étant pas parcimonieux, il est nécessaire de fixer un seuil de nullité des poids des caractéristiques : quand le poids affecté à une caractéristique

Algorithme 6.2 Algorithme RW [Kujala 2009]

Entrée : Jeu de données d'apprentissage $(\mathbf{x}^j, \mathbf{y}^j)_{j=1}^m$ et nombre maximal d'itérations N

Sortie : vecteur de poids $\mathbf{w}$

```
Initialisation v = [1...1] ∈ R^d
Pour t = 1 → N Faire
    Pour chaque observation x^j Faire
        Pour i = 1 → d Faire
            x'^j_i ← x^j_i v^(t)_i
        Fin Pour
    Fin Pour
    w^(t) ← solution des ℓ2-SVM en discrimination sur l'échantillon(x'^j, y^j)_{j=1}^m
    Pour i = 1 → d Faire
        v^(t+1)_i ← sqrt(|w^(t)_i v^(t)_i|)
    Fin Pour
Fin Pour
Pour i = 1 → d Faire
    w_i ← w^(N)_i v^(N)_i
Fin Pour
Retourner w
```

est inférieur au seuil, celle-ci est annulée.

Les algorithmes présentés ici ne sont pas adaptés pour effectuer la sélection de variables en apprentissage d'ordonnancement. En effet, il s'agit de méthodes d'approximations des normes ℓ_1 et ℓ_0 et non de sélection de variables. Par ailleurs, ils ne sont pas adaptés à l'apprentissage d'ordonnancement. Dans la suite de ces travaux, nous nous intéressons plus spécifiquement à l'adaptation des algorithmes ℓ_2-AROM et RW, nous ne considérons pas de terme ε et ne nous intéressons pas à son ajustement. Par ailleurs, nous souhaitons adapter la règle de mise à jour des poids de l'algorithme RW afin qu'il approche non pas la norme ℓ_0 mais la norme ℓ_1. Les points d'adaptations ainsi que les algorithmes que nous proposons sont présentés dans la section suivante.

6.2 Proposition d'algorithmes de résolution pour la sélection de variables en ordonnancement

Nous adaptons les algorithmes ℓ_2-AROM et RW présentés précédemment à la sélection de variables pour l'apprentissage d'ordonnancement. Nous proposons ainsi trois nouveaux algorithmes basés, pour l'un sur ℓ_2-AROM et pour les deux autres, sur l'algorithme RW. Ces derniers utilisent les ré-écritures des normes ℓ_0 et ℓ_1 via la norme ℓ_2 présentées respectivement en 6.1 et 6.4 au sein d'une structure contrôlant le nombre maximal de variables sélectionnées. Ces deux derniers algorithmes constituent ainsi deux variantes d'une même méthodes, capables d'approcher aussi

bien le problème en norme ℓ_1 que le problème en norme ℓ_0. Dans une première section, nous présentons les problèmes liés à l'adaptation et les solutions adoptées. Dans une seconde section, nous décrivons les algorithmes proposés.

6.2.1 Adaptation à la sélection de variables en apprentissage d'ordonnancement

L'adaptation des approches d'approximation des normes parcimonieuses présentées précédemment posent deux problèmes principaux : le passage de méthodes de discrimination à des méthodes d'apprentissage d'ordonnancement et le passage de méthodes d'approximation/estimation à des méthodes de sélection de variables. Nous détaillons ces problèmes et indiquons les solutions retenues dans la suite de cette section.

Problème 1 *Passage de la discrimination à l'apprentissage d'ordonnancement*
Les méthodes présentées précédemment ont été définies dans le cas de problème de discrimination et non d'apprentissage d'ordonnancement. Bien que nous ayons montré que le problème de l'apprentissage d'ordonnancement pouvait être réduit à un problème de discrimination, il est nécessaire de procéder à des modifications. En effet, les méthodes de repondération utilisent un algorithme de résolution en norme ℓ_2 qui doit être spécifique à l'apprentissage d'ordonnancement.

Solution *Utilisation de l'algorithme RankSVM-Primal pour la repondération*
Nous avons choisi de travailler sur les algorithmes ℓ_2-AROM et RW. Ces méthodes présentent l'avantage de pouvoir utiliser n'importe quel algorithme de résolution des SVM en norme ℓ_2 pour la mise à jour du vecteur de poids $\mathbf{w}_i$ à chaque itération. Ainsi, l'adaptation à l'apprentissage d'ordonnancement est effectuée en remplaçant l'algorithme de discrimination par un algorithme d'apprentissage d'ordonnancement utilisant les SVM, comme RankSVM-Primal ou RankSVM. En pratique, nous avons sélectionné l'algorithme RankSVM-Primal.

Problème 2 *Passage de l'approximation de la norme à la sélection de variables*
Les méthodes présentées n'ont pas été publiées en tant que méthodes de sélection de variables, mais comme méthodes d'approximation de normes parcimonieuses, que ce soit dans un contexte généraliste (algorithmes ℓ_2-AROM, RW) ou dans le contexte de la reconstruction de signaux. A priori, elles pourraient être utilisées en l'état comme méthodes de sélection parcimonieuse. Néanmoins, Weston [Weston 2003] indique dans le cas de l'algorithme ℓ_2-AROM que les modèles obtenus ainsi pourraient ne pas être optimaux. En effet, l'objectif de ces algorithmes est d'obtenir le modèle contenant le moins de variables possibles, sans se soucier de la capacité de généralisation de celui-ci. Autrement dit, les méthodes seront performantes en matière de parcimonie, mais de qualité de prédiction potentiellement médiocres.

Solution *Structure itérative et contrôle du nombre de variables* Pour résoudre ce problème, nous proposons, en nous inspirant de [Weston 2003], d'imposer comme contrainte d'arrêt de l'algorithme ℓ_2-AROM non pas un critère de convergence, mais un seuil r correspondant au nombre maximal de variables à conserver. Les itérations ℓ_2 de l'algorithme sont ainsi stoppées dès que la contrainte $\|\mathbf{w}\|_0 \leq r$ est vérifiée. Les variables correspondant à des poids $\mathbf{w}_i$ non nuls sont ensuite utilisées pour apprendre un nouveau modèle à l'aide d'un algorithme non parcimonieux de résolution en norme ℓ_2. Nous proposons d'utiliser le même critère d'arrêt, à la fois pour l'algorithme ℓ_2-AROM modifié et pour l'algorithme RW modifié.

Algorithme 6.3 Algorithme Rank-ℓ_2-AROM

Entrée : Jeu de données d'apprentissage $\{(\mathbf{x}^j, \mathbf{y}^j)\}_{j=1}^m$ et nombre maximal de variables à conserver r
Sortie : Vecteur de poids $\mathbf{w}$
%% Etape de sélection des variables
Initialisation $\mathbf{v} = [1...1] \in \mathbb{R}^d$
Tant que $\|\mathbf{w}\|_0 > r$ **Faire**
  $\mathbf{w} \leftarrow$ solution de RankSVM-Primal$(\{(\mathbf{v} * \mathbf{x}^j, \mathbf{y}^j)\}_{j=1}^m)$
  $\mathbf{v} \leftarrow \mathbf{v} * \mathbf{w}$
Fin Tant que
%% Récupération des variables pertinentes
$I \leftarrow \{i | \mathbf{w}_i \neq 0\}$
%% Apprentissage du modèle final
Pour $j = 1 \rightarrow m$ **Faire**
  Pour $i \in I$ **Faire**
    $\mathbf{z}_i^j \leftarrow \mathbf{x}_i^j$
  Fin Pour
Fin Pour
$\mathbf{w} \leftarrow$ solution de RankSVM-Primal$(\{(\mathbf{z}^j, \mathbf{y}^j)\}_{j=1}^m)$
Retourner $\mathbf{w}$

Ces modifications prises en compte, nous sommes en mesure d'utiliser ces algorithmes pour la sélection de variables en apprentissage d'ordonnancement. L'adaptation de l'algorithme RW pour la prise en compte des repondérations de le norme ℓ_0 par la norme ℓ_2 nécessite le changement de la règle de mise à jour du vecteur de poids $\mathbf{w}$. Nous décrivons chacun des algorithmes dans la section suivante.

6.2.2 Proposition de trois algorithmes pour la sélection de variables

Nous proposons ainsi de trois nouveaux algorithmes de type IRLS pour la sélection de variables en apprentissage d'ordonnancement : Rankℓ_2-AROM (*cf.* algorithme 6.3), RankRWFS-ℓ_1 et RankRWFS-ℓ_0 (*cf.* algorithme 6.4). RankRWFS-ℓ_1 et RankRWFS-ℓ_0 partagent la même structure et sont présentés au sein d'un même

Algorithme 6.4 Algorithme Rank-RWFS-$\ell_2\ell_.$

Entrée : jeu de données d'apprentissage $(\mathbf{x}^j, \mathbf{y}^j)_{j=1}^m$ et nombre maximal de variables r à conserver
Sortie : vecteur de poids $\mathbf{w}$

%% Etape de sélection des variables
Initialisation $\mathbf{v} = [1...1] \in \mathbb{R}^d$, $t = 1$
Tant que $\|\mathbf{w}\|_0 > r$ **Faire**
  Pour chaque observation $\mathbf{x}^j$ **Faire**
    Pour $i = 1 \rightarrow d$ **Faire**
      $\mathbf{x}_i^{'j} \leftarrow \mathbf{x}_i^j \mathbf{v}_i^{(t)}$
    Fin Pour
  Fin Pour
  $\mathbf{w}^{(t)} \leftarrow$ solution de RankSVM-Primal$(\{(\mathbf{x}^{'j}, \mathbf{y}^j)\}_{j=1}^m)$
  Pour $i = 1 \rightarrow d$ **Faire**
    if Algorithme $\ell_2\ell_1$ **then**
      %%%% Mise à jour pour l'approximation $\ell_2\ell_1$ %%%%
      $\mathbf{v}_i^{(t+1)} \leftarrow \sqrt{|\mathbf{w}_i^{(t)} \mathbf{v}_i^{(t)}|}$
    else if Algorithme $\ell_2\ell_0$ **then**
      %%%% Mise à jour pour l'approximation $\ell_2\ell_0$ %%%%
      $\mathbf{v}_i^{(t+1)} \leftarrow |\mathbf{w}_i^{(t)} \mathbf{v}_i^{(t)}|$
    Fin if
  Fin Pour
  Pour $i = 1 \rightarrow d$ **Faire**
    $\mathbf{w}_i \leftarrow \mathbf{w}_i^{(t)} \mathbf{v}_i^{(t)}$
  Fin Pour
  $t = t + 1$
Fin Tant que
%% Récupération des variables pertinentes
$I \leftarrow \{i | \mathbf{w}_i \neq 0\}$
%% Apprentissage du modèle final
Pour $j = 1 \rightarrow m$ **Faire**
  Pour $i \in I$ **Faire**
    $\mathbf{z}_i^j \leftarrow \mathbf{x}_i^j$
  Fin Pour
Fin Pour
$\mathbf{w} \leftarrow$ solution de RankSVM-Primal$(\{(\mathbf{z}^j, \mathbf{y}^j)\}_{j=1}^m)$
Retourner $\mathbf{w}$

paragraphe.

Rankℓ_2-AROM s'inspire de la méthode proposée dans [Weston 2003]. L'utilisateur fixe un seuil maximal de caractéristiques à conserver. Tant que ce seuil n'est pas atteint, l'algorithme en norme ℓ_2 est repondéré et les variables dont le poids est suffisamment faible (relativement à un seuil prédéterminé) sont retirées du modèle. Enfin, la fonction d'ordonnancement finale, n'utilisant que les caractéristiques restantes, est apprise à l'aide de l'algorithme en norme ℓ_2 utilisé pour les étapes de repondération.

RankRWFS-ℓ_1 et RankRWFS-ℓ_0 partagent la même structure, seule la pondération utilisée change. L'utilisateur fixe le seuil maximal de caractéristiques désirées : l'algorithme s'arrête lorsqu'un nombre inférieur ou égal à ce seuil est atteint. La repondération et le calcul du vecteur de poids $\mathbf{w}$ s'effectuent de la façon suivante : à chaque itération t, le vecteur de poids solution $\mathbf{w}^{(t)}$ est déterminé par l'algorithme en norme ℓ_2. Le vecteur $\mathbf{v} \in \mathbb{R}^d$ tel que $\mathbf{v}^{(1)} = [1 \dots 1]$ est mis à jour. Si la norme ℓ_1 est considérée, alors $\mathbf{v}_i^{(t+1)} = \sqrt{|\mathbf{w}_i^{(t)} \mathbf{v}_i^{(t)}|}$. Si la norme ℓ_0 est considérée, alors $\mathbf{v}_i^{(t+1)} = |\mathbf{w}_i^{(t)} \mathbf{v}_i^{(t)}|$. Enfin le vecteur de poids final $\mathbf{w}$ est calculé à partir de $\mathbf{v}^{(t)}$ tel que pour toute caractéristique i, $\mathbf{w}_i = \mathbf{w}_i^{(t)} \mathbf{v}_i^{(t)}$. Le nombre d'éléments non nuls de $\mathbf{w}_i$ indique le nombre de variables restantes. S'il est supérieur au seuil maximal autorisé, chaque valeur de caractéristique de chaque observation est pondérée par le vecteur $\mathbf{v}$, telle que $\mathbf{x}_i = \mathbf{x}_i \mathbf{v}_i^{(t)}$.
Une fois la sélection effectuée, l'algorithme en norme ℓ_2 est utilisé pour apprendre le modèle final.

Dans la suite de ce chapitre, nous présentons les expérimentations menées afin d'évaluer les algorithmes proposés. Les résultats obtenus mettent en évidence que les méthodes proposées sont efficaces pour la sélection de variables en apprentissage d'ordonnancement, tant en matière de qualité d'apprentissage que de vitesse d'exécution.

6.3 Protocole expérimental

Dans cette partie, nous présentons les différentes expérimentations menées ainsi que leurs objectifs. Nous détaillons également les mesures d'évaluation, jeux de données, algorithmes de référence et paramètres utilisés lors de ces expériences.

6.3.1 Jeux de données et algorithmes de référence

Nous avons choisi d'évaluer nos approches sur cinq jeux de données issus des collections de référence LETOR 3 et 4, à savoir : Ohsumed, HP2004, NP2004, TD2004 et MQ2008. Les caractéristiques de ces jeux de données ainsi que les variables qui les constituent ont été présentées au chapitre 2, tableaux 2.1, 2.2, 2.3 et 2.4. Les

algorithmes de référence considérés sont RankSVM-Primal, FenchelRank et FSMRank. RankSVM-Primal est l'algorithme non parcimonieux utilisé en tant que boîte noire par nos approches. FenchelRank et FSMRank sont choisis car il s'agit des algorithmes de sélection de variables pour l'apprentissage d'ordonnancement les plus récents [Lai 2013a, Lai 2013b].

Nous évaluons la performance de nos algorithmes en matière d'ordonnancement à l'aide de la MAP et du NDCG@10. Nous nous intéressons également au temps d'exécution des phases de sélection de variables et de prédiction. La capacité des algorithmes à sélectionner un faible nombre de caractéristiques est mesurée à l'aide des ratios de parcimonie, qui correspondent au pourcentage moyen de variables restantes à la fin de l'étape de sélection (*cf.* définition 17).

Définition 17 (Ratio de parcimonie)**.**

Soient :
– i un indice représentant l'algorithme considéré,
– j un indice représentant le jeu de données,
alors le ratio de parcimonie de l'algorithme i sur le jeu de données j, noté $RP_{i,j}$ correspond à la fraction de caractéristiques restantes à la fin de l'étape de sélection, soit :

$$RP_{i,j} = \frac{\text{Nombre de caractéristiques restantes}}{\text{Nombre de caractéristiques initiales}}$$

Les ratios de parcimonie sont liés au seuil de variables maximal imposé. Ainsi, si nous désirons conserver au plus 30 % des caractéristiques, le ratio de parcimonie théorique est égal $\frac{30}{100} = 0.3$. Néanmoins, la valeur réellement obtenue peut être inférieure.

6.3.2 Choix des paramètres expérimentaux

Les trois algorithmes proposés présentent la même structure en deux étapes. La première étape correspond à la phase de sélection des variables pertinentes et la deuxième à l'apprentissage du modèle final. Pour chacun d'eux, plusieurs seuils de variables sont analysés afin de sélectionner celui présentant le meilleur compromis entre parcimonie, qualité d'ordonnancement et temps d'exécution. Pour ces expériences, nous avons testé des seuils à 90 %, 70 %, 50%, 30 % et 10 % de caractéristiques restantes. Par ailleurs, nous avons fixé à quarante le nombre maximal d'itérations, afin d'éviter des temps de calcul trop importants.

Pour chacune des deux phases des algorithmes, la valeur optimale du paramètre de contrôle des erreurs C (*cf.* chapitre 5 page 85) doit être déterminée pour chaque sous-échantillon de chaque jeu de données. Le C choisi est celui qui maximise la MAP sur l'échantillon de validation. Lors de la phase de sélection, nous testons les valeurs du paramètre sur la grille [1 10 100 1000 10000]. Les bornes, supérieure et

inférieure, ont été choisies ici d'un point de vue pratique. De façon générale, l'algorithme est d'autant plus parcimonieux que le C considéré est petit. Pour des valeurs inférieures à 1, la sélection est généralement trop brutale : toutes les variables sont supprimées en une ou deux itérations. Pour des valeurs supérieures à 10000, les temps d'exécution nécessaires pour atteindre le seuil désiré sont très longs et les valeurs de MAP ne sont pas nécessairement améliorées. Lors de la phase d'apprentissage du modèle final, nous avons utilisé la grille usuelle de RankSVM-Primal où $C \in [0.0001\ 0.001\ 0.01\ 0.1\ 1\ 10\ 100\ 1000\ 10000\ 100000]$.

6.3.3 Description des expérimentations

Dans ces expérimentations, nous procédons à des analyses quantitatives et qualitatives des algorithmes. Nous souhaitons non seulement étudier le comportement et les propriétés de chaque algorithme, mais aussi comparer la performance des algorithmes entre eux et avec les références. Plus précisément, nous considérons indépendamment chaque algorithme afin de :

1. Connaître le ratio de parcimonie effectif pour chaque seuil et analyser son évolution ;
2. Connaître les temps d'exécution des phases de sélection et d'apprentissage pour chaque seuil et analyser leur évolution ;
3. Déterminer l'impact, s'il existe, de la sélection de variables sur les mesures d'évaluation (MAP et NDCG@10).

Nous comparons également les algorithmes proposés entre eux et aux références afin de déterminer si :

1. Pour un seuil fixé, un des algorithmes est plus parcimonieux que les autres ;
2. Sur l'ensemble des seuils, un algorithme est plus performant du point de vue des mesures d'évaluation (MAP et NDCG@10) ;
3. Sur l'ensemble des seuils, un algorithme est plus performant du point de vue des temps d'exécution.

Enfin, nous effectuons une analyse qualitative des modèles appris pour le seuil de sélection à 10 %. Nous étudions dans ce cas l'accord inter-algorithmes, c'est-à-dire la capacité des algorithmes à sélectionner les mêmes caractéristiques. Nous analysons les modèles finaux appris afin d'extraire des sous-ensembles de caractéristiques pertinentes et importantes pour l'apprentissage d'ordonnancement.

6.4 Résultats

Dans cette section, nous présentons les résultats obtenus lors des expérimentations et les analysons. Dans un premier temps, nous nous intéressons aux ratios de parcimonies effectifs des différents algorithmes. Dans un second temps, nous évaluons l'impact du seuil de sélection sur la MAP pour chaque algorithme. Nous poursuivons cette analyse quantitative en comparant les MAP et valeurs de NDCG

obtenues aux valeurs de référence. Puis, nous comparons les temps d'exécution de chacun des algorithmes. Enfin, nous procédons à l'analyse qualitative des modèles appris.

6.4.1 Ratio de parcimonie : des résultats globalement équivalents

Nous analysons dans cette partie les ratios de parcimonie effectivement obtenus à la fin de l'étape de sélection. La figure 6.1 présente ces ratios pour l'ensemble des seuils, des algorithmes et des jeux de données considérés. Les ratios de parcimonie théoriques sont représentés par la ligne pointillée bleue.

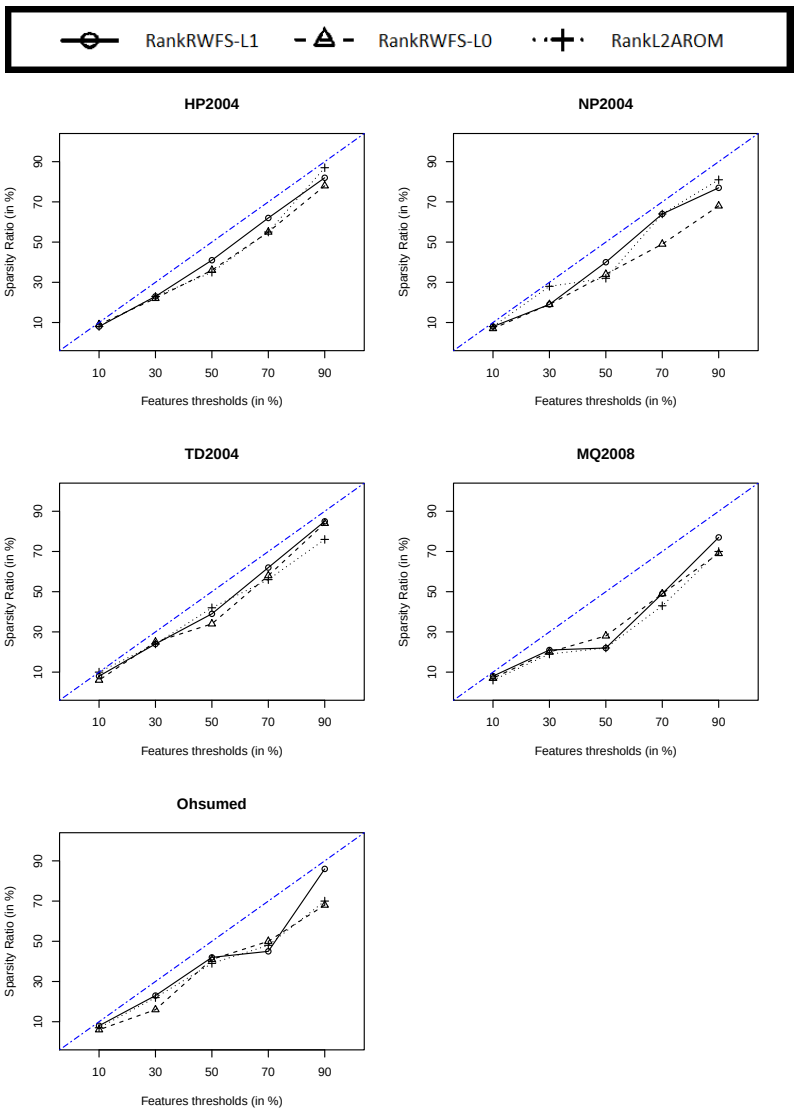


FIGURE 6.1 – Comparaison entre ratios de parcimonie effectifs et seuil de sélection pour les trois algorithmes sur chaque jeu de donnée.

Les ratios de parcimonie des algorithmes sont contrôlés par le taux de sélection imposé. Néanmoins, ce taux ne fixe que la limite maximale du nombre de variables autorisées. En pratique, les algorithmes ne suppriment pas les variables une par une à chaque itération, mais par blocs et le ratio de parcimonie final est généralement plus faible que le seuil imposé (*cf.* figure 6.1). Nous observons que les trois algorithmes présentent des ratios de parcimonie similaires pour les plus faibles seuils de sélection (10 % et 30 %). A l'inverse, les algorithmes en norme ℓ_0 sont généralement plus parcimonieux pour le seuil de 90 %. Ainsi, sur MQ2008 et Ohsumed, le ratio de parcimonie obtenu est de l'ordre de 70 % pour RankRWFS-ℓ_0 et Rankℓ_2-AROM contre environ 80 % pour RankRWFS-ℓ_1. Sur TD2004, Rankℓ_2-AROM présente un taux plus faible que les algorithmes de l'approche RankRWFS. A l'inverse, RankRWFS-ℓ_0 présente les plus faibles taux au seuil de 90 % pour HP2004 et NP2004.

De façon générale, les ratios de parcimonie diminuent de façon régulière pour l'ensemble des algorithmes sur les jeux de données, à l'exception de MQ2008 et Ohsumed. Sur MQ2008, les ratios de parcimonie des algorithmes diminuent de façon brutale pour atteindre une valeur moyenne inférieure à 30 % au seuil maximal imposé de 50 %. Sur Ohsumed, le ratio de parcimonie effectif de RankRWFS-ℓ_1 passe de 80% pour le seuil imposé de 90 %, à moins de 50 % pour le seuil imposé de 70 %, puis stagne. Les algorithmes semblent donc plus "agressifs" dans la sélection de variables sur ces deux jeux de données, ce qui pourrait traduire une proportion plus élevée de caractéristiques bruitées ou non pertinentes. Sur une majorité des jeux de données, RankRWFS-ℓ_0 semble avoir tendance à présenter des ratios de parcimonie effectifs légèrement plus faibles que les deux autres algorithmes. Néanmoins, la différence n'est pas très nette. Nous pouvons conclure que bien que de légères différences existent suivant les jeux de données considérés, les algorithmes ont des comportements globalement équivalents et sont capables d'atteindre de faibles ratios de parcimonie.

6.4.2 Impact limité de la sélection de variables sur les mesures d'évaluation

L'objectif de la sélection de variables est la suppression des caractéristiques bruitées ou non pertinentes, afin de ne conserver que celles réellement utiles pour l'apprentissage de modèles. Un algorithme de sélection efficace devra ainsi être capable de sélectionner un sous-ensemble restreint de caractéristiques sans pour autant dégrader la qualité des modèles appris. Les mesures d'évaluation comme la MAP et le NDCG devraient donc rester stables voire augmenter lorsque les caractéristiques sont supprimées. Ces mesures ont été définies au chapitre 1 section 1.2.2.

Dans cette section, nous proposons d'étudier l'évolution des mesures d'évaluation en fonction du seuil de sélection imposé. L'objectif est de vérifier que les algorithmes conservent un bon comportement quel que soit le niveau de parcimonie atteint. Nous portons une attention particulière aux très faibles seuils de sélection.

En effet, nous avons vu à la section 6.4.1 que les algorithmes proposés sont capables d'apprendre des modèles contenant un très faible nombre de variables (moins de 10 % des caractéristiques initiales). Or, nous pouvons supposer que plus le taux de sélection maximal imposé est faible, plus le risque d'avoir supprimé des caractéristiques pertinentes est important. Il est légitime de vérifier que la suppression des variables n'entraîne pas de diminution de la qualité au sens des mesures considérées.

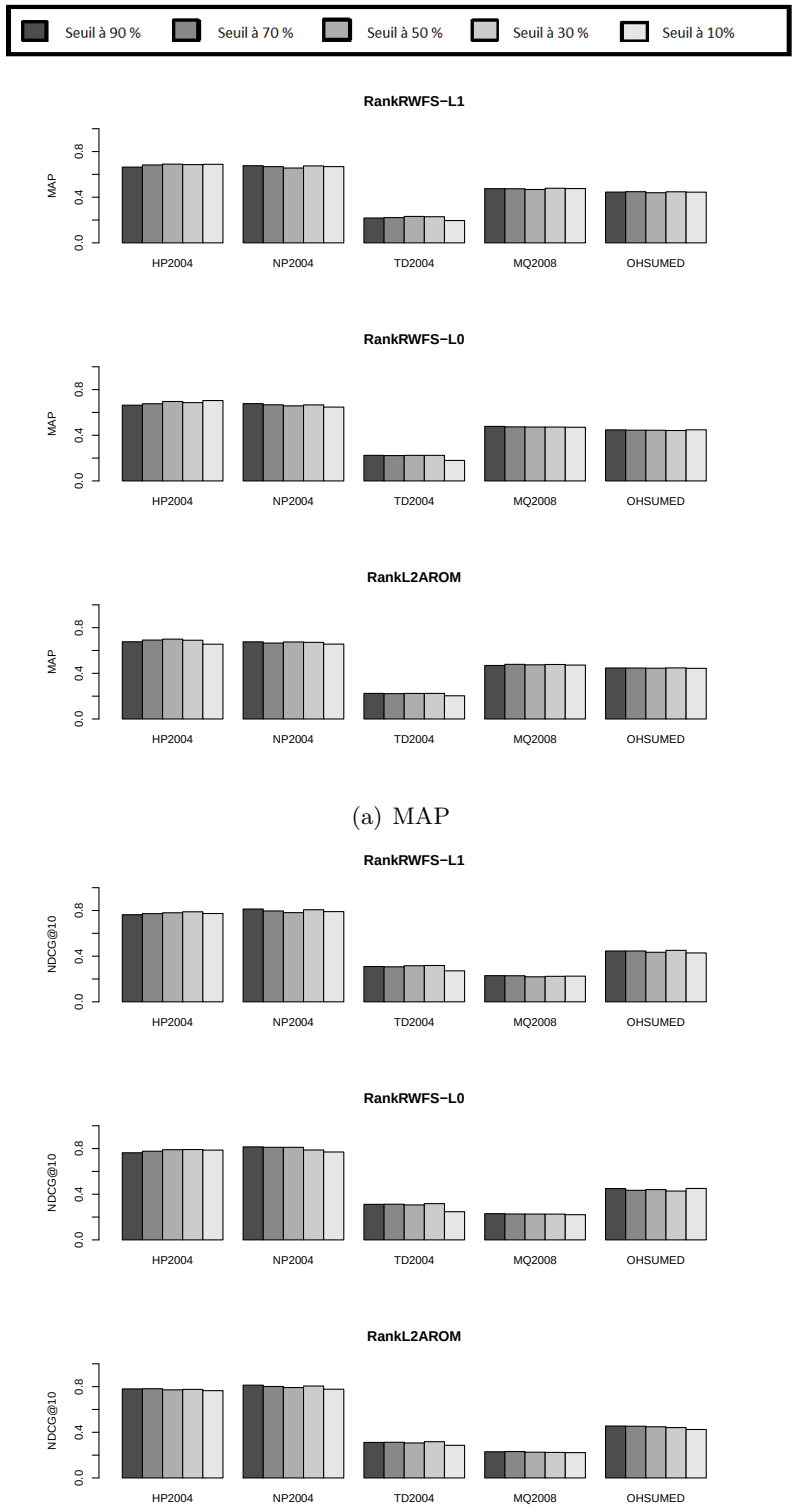
La figure 6.2 présente les valeurs moyennes de MAP (en haut) et de NDCG@10 (en bas) pour chaque seuil, chaque algorithme et chaque jeu de données. Notre première constatation est que l'allure des graphes est identique pour l'ensemble des algorithmes, sur chaque jeu de données et quelle que soit la mesure considérée. Les trois algorithmes que nous proposons ont donc un comportement similaire en ce qui concerne la qualité d'ordonnancement des modèles prédits. Par ailleurs, nous observons que la MAP et le NDCG restent stables pour l'ensemble des seuils sur les quatre jeux de données NP2004, HP2004, MQ2008 et Ohsumed. Sur ces jeux de données, la sélection ne dégrade pas les mesures d'évaluation. Sur TD2004, nous observons que les valeurs de MAP et de NDCG restent stables pour des seuils de caractéristiques maximaux allant de 90 % à 30 %. Pour le seuil de 10 %, nous notons une dégradation des mesures pour tous les algorithmes considérés. Sur ce jeu de données spécifique, les algorithmes ne sont pas capables de maintenir une qualité d'ordonnancement constante lorsque le nombre de caractéristiques devient très faible.

En conclusion, l'impact de la sélection sur la qualité des modèles appris reste limité. Les valeurs de MAP et de NDCG@10 restent stables pour l'ensemble des seuils de sélection pour tous les algorithmes et les jeux de données, à l'exception du jeu de données TD2004, pour lequel une légère dégradation des mesures est observée au seuil de sélection maximale à 10 %. Les algorithmes sont donc globalement capables de sélectionner un faible nombre de caractéristiques, tout en conservant la même qualité d'ordonnancement des modèles appris.

6.4.3 Des qualités d'ordonnancement comparables à l'état de l'art

Les analyses précédentes ont montré que les algorithmes proposés étaient efficaces pour effectuer la sélection de variables. En effet, ils sont capables d'apprendre des modèles comportant un nombre restreint de caractéristiques, tout en conservant la même qualité d'ordonnancement. Nous souhaitons à présent comparer la performance en matière d'ordonnancement des algorithmes proposés à celle des algorithmes de référence.

Les figures 6.3, 6.4 et 6.5 comparent les valeurs de MAP et de NDCG@10 des algorithmes Rankℓ_2-AROM, RankRWFS-ℓ_0 et RankRWFS-ℓ_1 respectivement, à celles des algorithmes de référence RankSVM-Primal et FenchelRank. La figure 6.6 compare les valeurs de MAP obtenues par les algorithmes de références et les algorithmes proposés au seuil de 10 %. Cette représentation permet d'avoir une vision globale du comportement des différents algorithmes et de les comparer plus aisément. Les graphes présentés sont appelés des *spider plots*, en référence à leur allure en toile



(a) MAP

(b) NDCG@10

FIGURE 6.2 – Evolution de la MAP (haut) et du NDCG@10 (bas) suivant le seuil de sélection pour chaque algorithme sur tous les jeux de données.

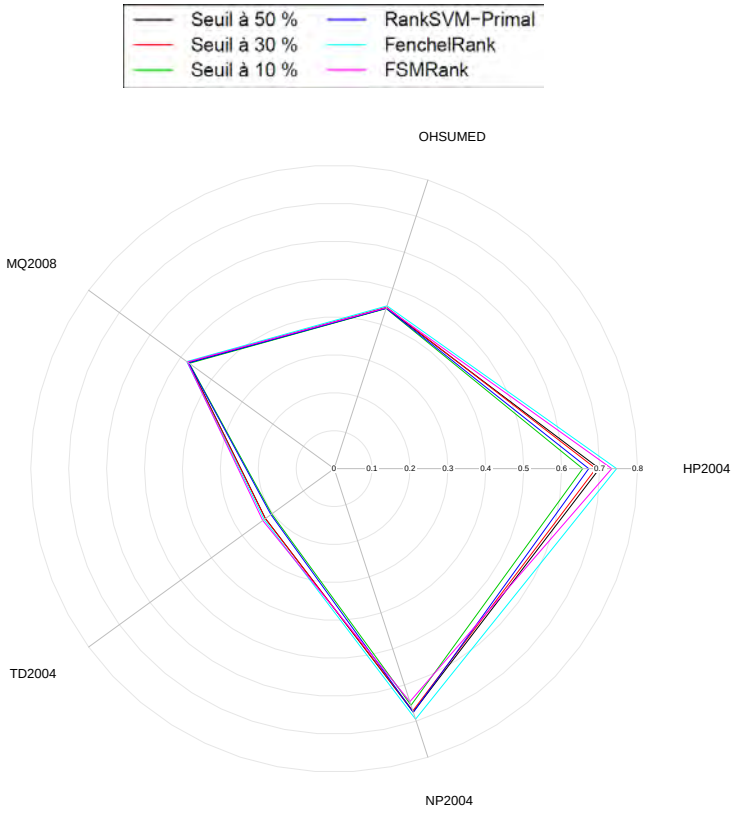
d'araignée. Ils permettent de visualiser simultanément les résultats des différents algorithmes sur tous les jeux de données pour lesquels ils ont été testés. Chaque jeu de données est représenté par un axe de la toile, tandis que chaque ligne pointillée présente la valeur de la mesure considérée pour un algorithme donné.

Nous constatons que tous les algorithmes, FenchelRank et RankSVM-Primal inclus, présentent la même allure, quelle que soit la mesure considérée. Nous pouvons directement en déduire que les algorithmes fournissent des résultats globalement comparables, malgré quelques légères variations. Dans la suite, nous commentons plus en détails ces résultats.

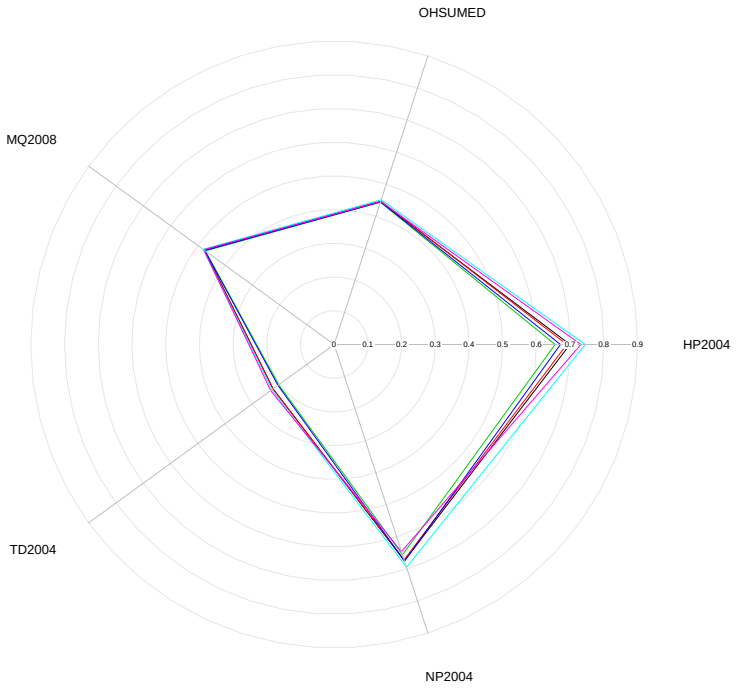
Considérons dans un premier temps les graphes 6.3(a), 6.4(a) et 6.5(a) représentant les valeurs de MAP. Nous observons que les courbes des différents algorithmes se superposent exactement pour les jeux de données Ohsumed et MQ2008. Les résultats des algorithmes sont donc parfaitement équivalents sur ces deux jeux de données. Sur TD2004, nous observons deux groupes d'algorithmes. Le premier, présentant les valeurs les plus faibles de MAP, est composé de l'algorithme de référence RankSVM-Primal et des algorithmes proposés avec le seuil de 10 %. Le deuxième, présentant les valeurs de MAP les plus élevées, est composé de l'algorithme de référence FenchelRank et des algorithmes proposés avec les seuils supérieurs à 10 %. Les algorithmes proposés obtiennent ainsi des résultats équivalents à FenchelRank et sont meilleurs que RankSVM-Primal pour des seuils supérieurs à 10 %. Pour le seuil de 10%, les algorithmes proposés fournissent des résultats légèrement moins bons ou comparables à RankSVM-Primal. Ce dernier point n'est pas étonnant puisque nous avons vu que pour ce jeu de données particulier, les mesures d'évaluation étaient dégradées lorsque le taux de sélection imposé était le plus faible. Si la qualité générale des algorithmes n'est pas remise en question, cela met en évidence que le seuil de 10 % n'est pas adéquat sur ce jeu de données particulier. Enfin, nous observons que les algorithmes proposés fournissent des résultats équivalents à RankSVM-Primal et légèrement inférieurs à FenchelRank sur NP2004 et HP2004.

Les résultats obtenus avec le NDCG@10 sont globalement similaires à ceux obtenus avec la MAP. Les valeurs de NDCG@10 sont identiques à celles des algorithmes de référence sur MQ2008. Sur NP2004 et Ohsumed, les valeurs sont comparables. Sur TD2004, les valeurs de NDCG@10 sont comparables à celles de FenchelRank et supérieures à celles de RankSVM-Primal, à l'exception des modèles appris avec le seuil de 10 %, pour lesquels le NDCG@10 est dégradé. Enfin, les valeurs observées sur HP2004 sont comparables à celles de RankSVM-Primal et légèrement inférieures à celles de FenchelRank.

En conclusion, les algorithmes proposés présentent des performances d'ordonnancement similaires aux références, quelle que soit la mesure considérée. Les algorithmes fournissent ponctuellement des résultats meilleurs que RankSVM-Primal et des résultats inférieurs à FenchelRank. Nous pouvons conclure que les algorithmes sont globalement comparables à l'existant du point de vue de la qualité de l'ordonnancement.

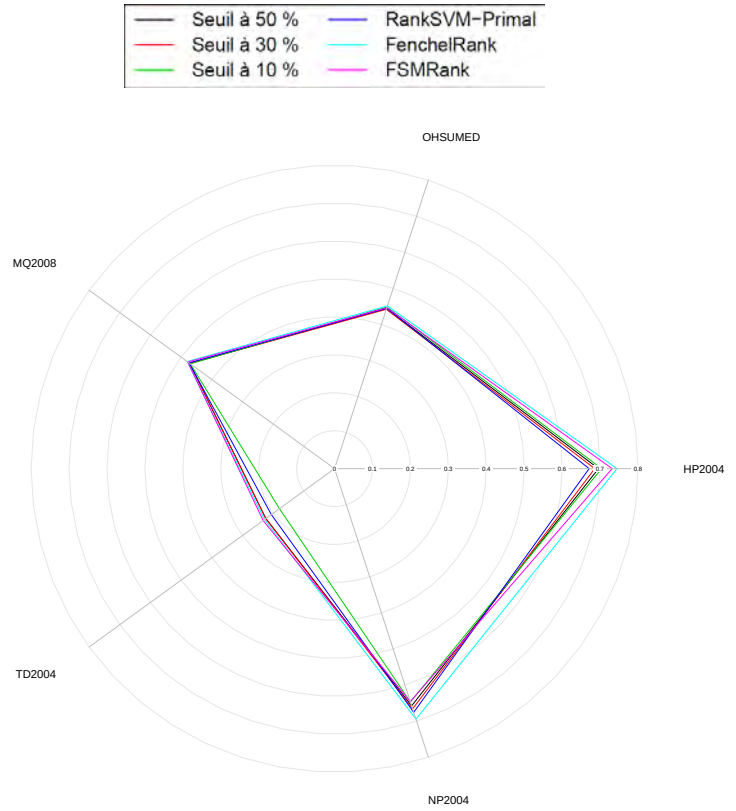


(a) MAP

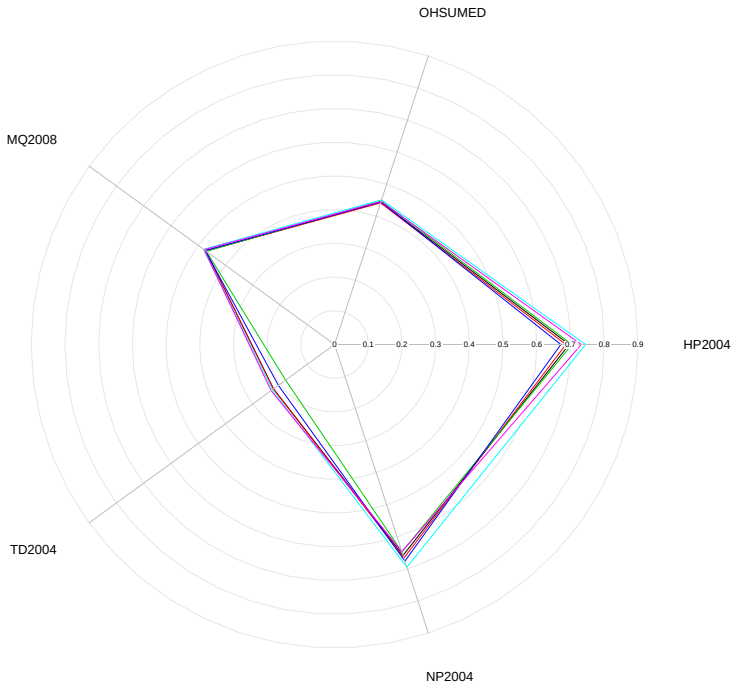


(b) NDCG@10

FIGURE 6.3 – Comparaison des MAP (haut) et NDCG@10 (bas) entre Rankℓ_2-AROM et les algorithmes de référence, pour tous les seuils de sélection.

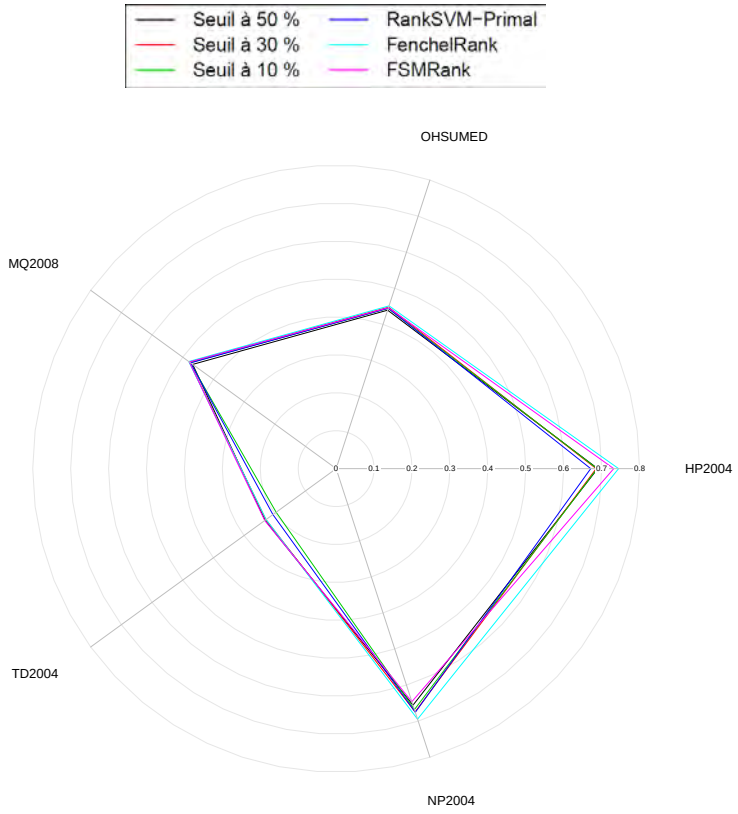


(a) MAP

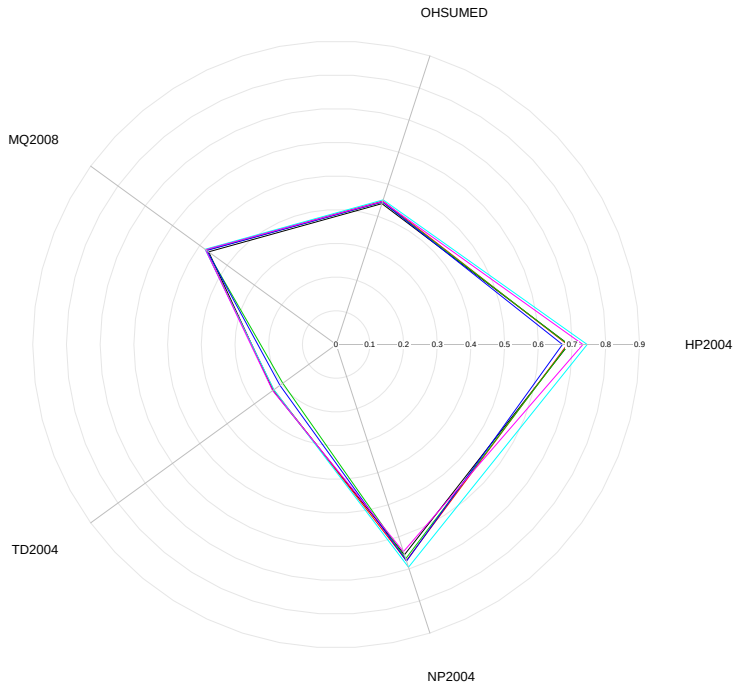


(b) NDCG@10

FIGURE 6.4 – Comparaison des MAP (haut) et NDCG@10 (bas) entre RankRWFS-ℓ_0 et les algorithmes de référence, pour tous les seuils de sélection.

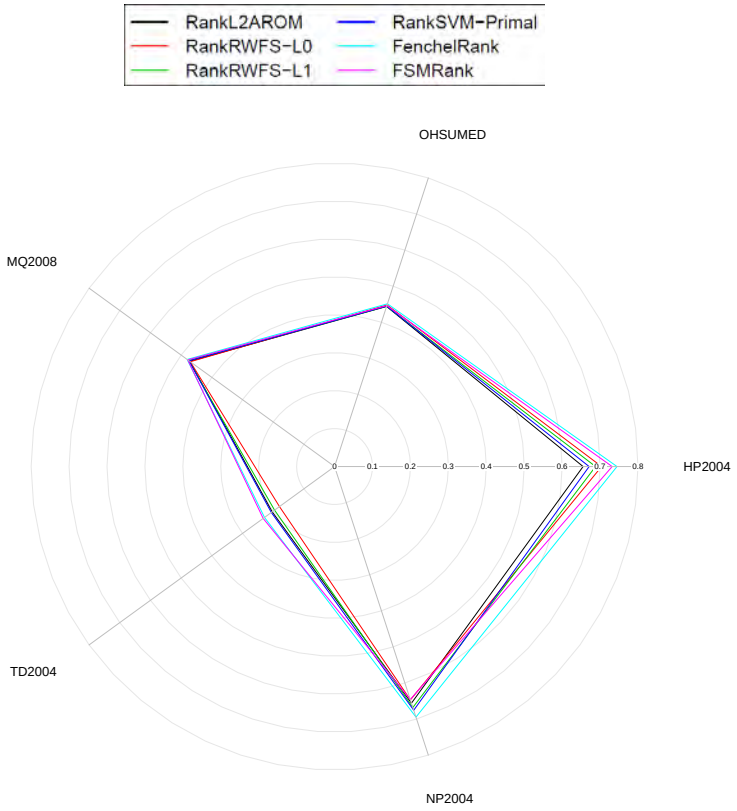


(a) MAP

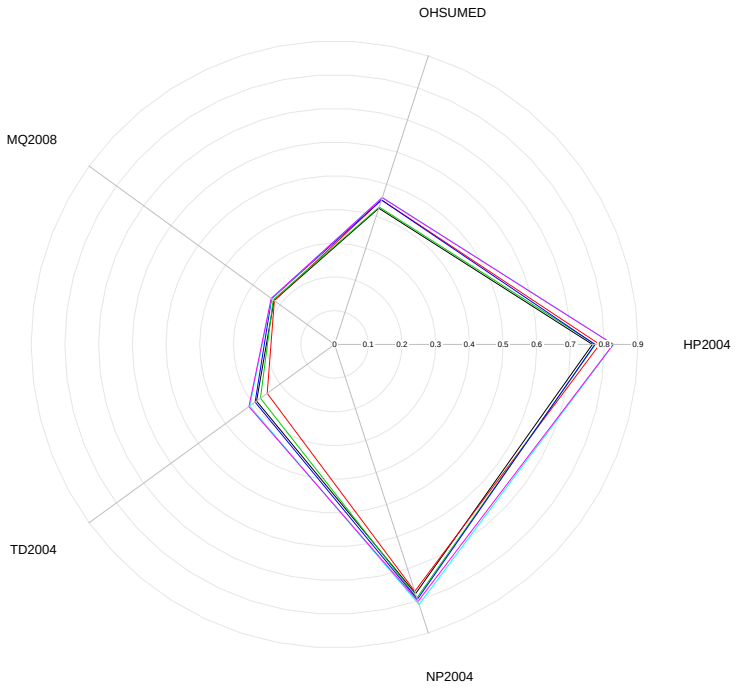


(b) NDCG@10

FIGURE 6.5 – Comparaison des MAP (haut) et NDCG@10 (bas) entre RankRWFS-ℓ_1 et les algorithmes de référence, pour tous les seuils de sélection.



(a) MAP



(b) NDCG@10

FIGURE 6.6 – Comparaison des MAP (haut) et NDCG@10 (bas) entre les algorithmes de référence et nos propositions (seuil de 10% de sélection).

6.4.4 Des temps d'exécution raisonnables, mais influencés par les seuils de sélection

Dans les sections 6.4.1 à 6.4.3, nous avons étudié la qualité d'ordonnancement des méthodes de repondération, que nous avons mis en relation avec leur ratio de parcimonie. Nous avons notamment montré que les algorithmes proposés étaient efficaces en sélection et compétitifs par rapport à l'existant. Dans cette partie, nous nous intéressons plus spécifiquement à la vitesse d'exécution des algorithmes. Notamment, nous souhaitons déterminer si :

1. Les temps d'exécution varient suivant le seuil de sélection imposé ;
2. Les temps d'exécution sont raisonnables [1] ;
3. Un algorithme est globalement plus ou moins rapide que les autres.

Les expérimentations ont été réalisées sur un MacBook Pro, utilisant Mac OS X Snow Leopard, un processeur Intel Core 2 Duo cadencé à 2.4 GHz et 4 Go de RAM (2 fois 2 Go 1067 MHz DDR3). Nous considérons que des temps de calcul raisonnables vont de quelques secondes à une ou deux minutes. Les figures 6.7 et 6.8 présentent les temps d'exécution des phases de sélection et d'apprentissage du modèle final, pour l'ensemble des algorithmes, sur chaque jeu de données.

Nous constatons que le taux de sélection a une influence sur le temps d'exécution de la phase de sélection. Ce dernier augmente quand le ratio de parcimonie demandé diminue. Ce résultat était attendu. En effet, les algorithmes doivent normalement réaliser plus d'itérations pour atteindre des taux de parcimonie plus faibles, ce qui augmente nécessairement le temps de calcul. Cette augmentation est variable suivant le jeu de données et l'algorithme considérés. Nous notons des augmentations brutales des temps de calcul pour les faibles seuils de caractéristiques maximaux imposés, en particulier pour le seuil à 10%. Nous observons notamment que :

- Sur HP2004, les temps de calcul des différents algorithmes augmentent progressivement d'une seconde pour le seuil de 90 % à environ deux secondes pour le seuil de 30 %, puis brutalement pour le seuil de 10 % pour atteindre cinq à six secondes suivant l'algorithme considéré.
- Sur NP2004, les temps d'exécution augmentent très légèrement entre les seuils de 90 % et 30 %, puis augmentent brutalement au seuil de 10 %. Ils sont multipliés par deux pour RankRWFS-ℓ_0, par trois pour Rankℓ_2-AROM et par quatre pour RankRWFS-ℓ_1.

Ce phénomène est ponctuellement observé sur :

- Rankℓ_2-AROM sur TD2004, avec une augmentation brutale de 40 à 140 secondes pour des seuils inférieurs à 50 %,
- Rankℓ_2-AROM et RankRWFS-ℓ_1 sur MQ2008, avec une multiplication par 1.5 et 2 respectivement entre les seuils de 30 % et de 10 %,
- Rankℓ_2-AROM et RankRWFS-ℓ_1 sur Ohsumed, avec une multiplication par 3 des temps de calcul entre les seuils de 50 % et 10 % pour le premier et les seuils de 70 % à 10 % pour le second.

1. de l'ordre de quelques secondes à moins de cinq minutes

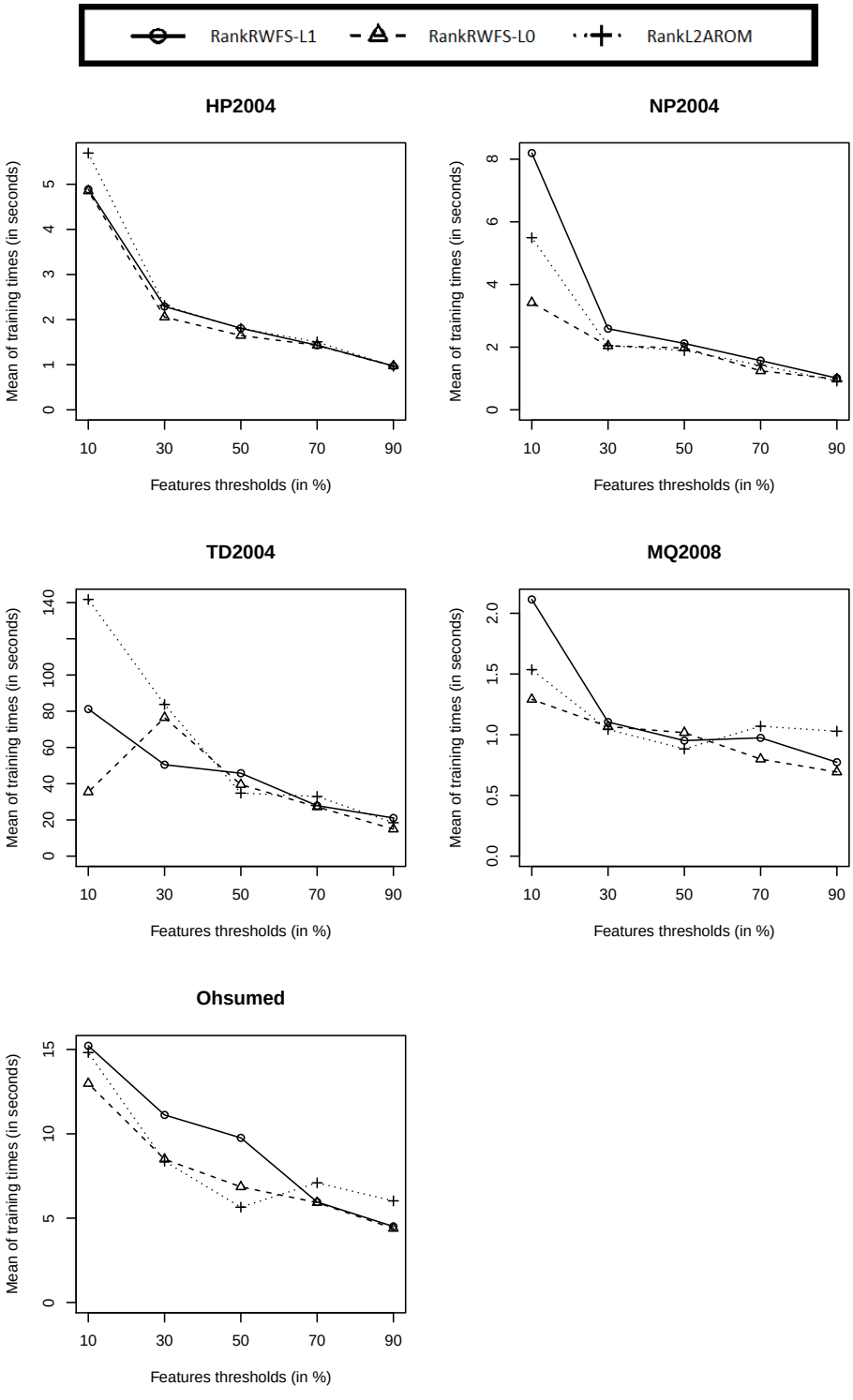


FIGURE 6.7 – Comparaison des temps d'exécution pour la phase de sélection et pour l'ensemble des algorithmes proposés.

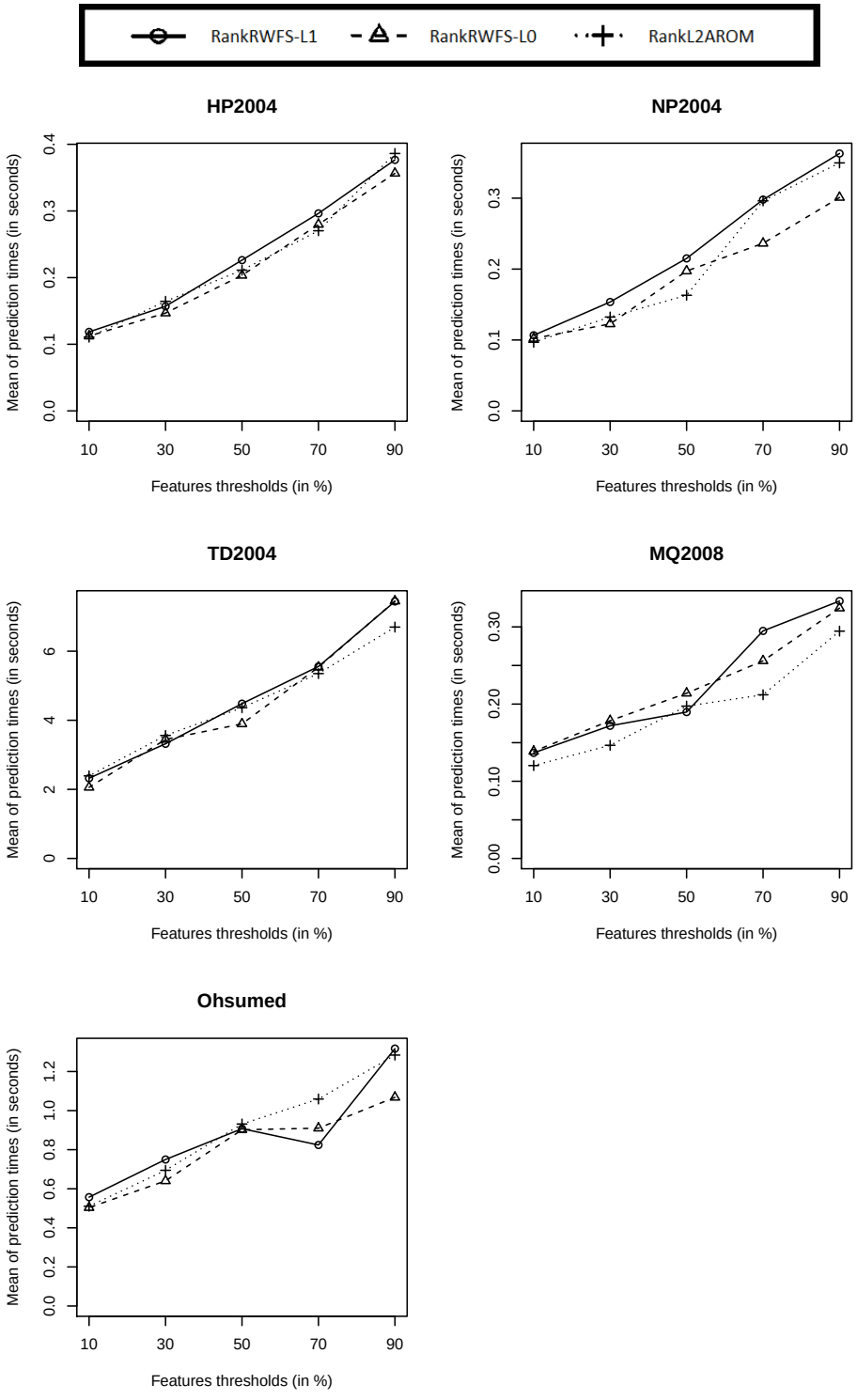


FIGURE 6.8 – Comparaison des temps d'exécution pour la phase d'apprentissage du modèle final, pour l'ensemble des algorithmes proposés.

De façon générale, le processus de sélection est d'autant plus long que le nombre maximal de caractéristiques requises est faible. L'augmentation importante des temps de calcul pour les petits seuils de sélection pourrait traduire la difficulté de sélectionner des sous-ensembles de caractéristiques de taille très réduite. Les algorithmes doivent en effet réaliser de plus en plus d'itérations, sans pour autant atteindre le seuil requis.

Nous vérifions par ailleurs que les temps de calcul de la phase d'apprentissage des modèles finaux diminuent avec les seuils de sélection. Cela est logique puisque le nombre de variables, donc la complexité du problème à résoudre, diminue.

Nous observons que les temps de calcul restent raisonnables. Ainsi, pour la phase de sélection, ils sont généralement de deux à quinze secondes maximum pour l'ensemble des jeux de données, à l'exception de TD2004. Sur ce dernier, les temps d'exécution varient de 40 à 140 secondes pour Rankℓ_2-AROM et de 40 à 80 secondes pour les algorithmes de l'approche RWFS. La phase d'apprentissage des modèles finaux dure généralement entre 0.1 et 1.2 secondes suivant les jeux de données, à l'exception TD2004. Pour ce dernier, les temps d'exécution sont globalement dix fois plus longs que sur les autres jeux de données. Ces différences peuvent être expliquées par le fait que TD2004 est de plus grande dimension que les autres jeux de données, puisqu'il contient de deux à dix fois plus de préférences que ces derniers.

Enfin, nous observons que l'algorithme RankRWFS-ℓ_1 présente généralement des temps d'exécution plus longs que les algorithmes résolvant le problème des SVM en norme ℓ_0. Ces derniers apparaissent donc comme prometteurs pour la sélection de variables en apprentissage d'ordonnancement, puisqu'il sont performants non seulement du point de vue de la qualité d'apprentissage, mais aussi du point de vue des temps de calcul. Par ailleurs, nous observons généralement que RankRWFS-ℓ_0 est légèrement plus rapide que Rankℓ_2-AROM, ce qui montre l'intérêt de l'utilisation de cette approche.

En conclusion, l'étude des temps d'exécution a montré que :

- Les temps d'exécution de la phase de sélection augmentent lorsque le nombre de caractéristiques requises diminue. Pour le seuil à 10 %, cette augmentation peut être brutale.
- Les temps d'exécution sont globalement raisonnables, de l'ordre de la quinzaine de secondes en sélection.
- Les approches en norme ℓ_0 sont légèrement plus rapides que l'approche en norme ℓ_1. Par ailleurs, l'approche RWFS en norme ℓ_0 semble également légèrement plus rapide que l'approche ℓ_2-AROM.

6.4.5 Des algorithmes de repondération offrant globalement de meilleures performances que l'état de l'art

Dans cette section, nous comparons les algorithmes que nous proposons aux méthodes de l'état de l'art FenchelRank et FSMRank, en prenant en compte l'ensemble des critères d'évaluation simultanément. Plus précisément, nous comparons les différents algorithmes sur leur capacité à supprimer un grand nombre de variables

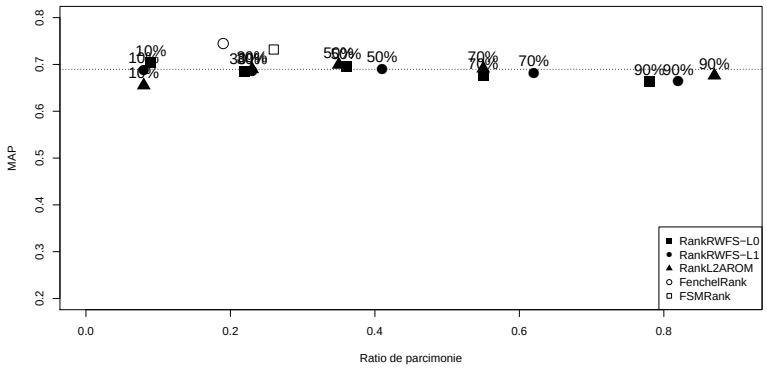
dans un temps raisonnable et en conservant une bonne qualité d'ordonnancement.

Les figure 6.9 et 6.10 (pages 119 et 120) représentent les valeurs de MAP obtenues par chaque algorithme en fonction du ratio de parcimonie atteint, sur chaque jeu de données. Les méthodes les plus performantes seront celles dont le ratio de parcimonie est le plus faible possible, sans dégradation de la MAP. Nous observons que sur tous les jeux de données à l'exception de HP2004, nos méthodes présentent un meilleur rapport MAP *vs.* ratio de parcimonie que les méthodes de l'état de l'art lorsque les seuils de sélection théoriques sont inférieurs ou égaux à 50 %. En effet, dans ces cas, les ratios de parcimonie sont plus faibles que ceux des méthodes de référence, tandis que les valeurs de MAP restent équivalentes. Dans le cas de HP2004, nous observons que nos méthodes obtiennent des valeurs de MAP légèrement plus faibles que celles des références, mais qu'elles parviennent néanmoins à atteindre de plus petits ratios de parcimonie. Globalement, nos méthodes sont donc plus performantes que celles de l'état de l'art pour supprimer un grand nombre de variables, sans dégrader la qualité de l'ordonnancement.

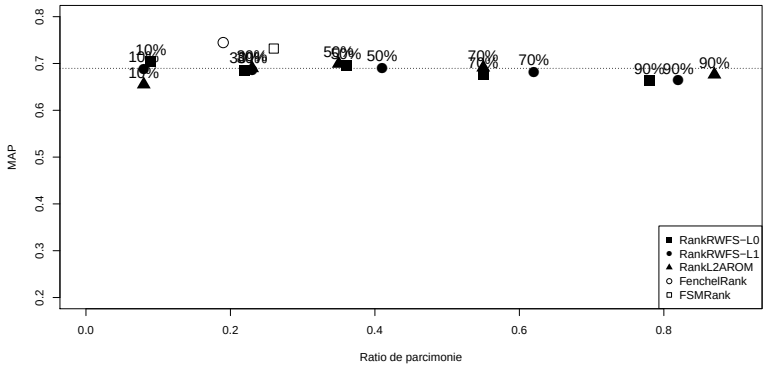
La figure 6.11 et 6.12 (pages 121 et 122) représentent le temps d'exécution total des algorithmes (sélection et apprentissage) en fonction du ratio de parcimonie atteint. Les meilleurs méthodes seront celles qui offrent le meilleurs compromis entre faible ratio de parcimonie et faible temps d'exécution. Notons que seul FSMRank est considéré comme algorithme de référence. En effet, nous ne disposons pas des temps d'exécution de FenchelRank. Par ailleurs, il est intéressant de nous comparer à FSMRank, qui a été présenté comme un algorithme rapide par ses auteurs [Lai 2013b].

Nous observons que nos approches sont globalement bien plus rapides que FSMRank, pour tous les seuils de sélection et sur tous les jeux de données à l'exception de MQ2008 pour lequel FSMRank est plus performant. Sur Ohsumed, nos approches sont environ 4.5 fois plus rapide que FSMRank pour des ratios de parcimonie similaires ou meilleurs. Sur TD2004, nos algorithmes sont généralement 2 à 3 fois plus rapides, tout en étant plus parcimonieux. Ils sont jusqu'à 4 et 5 fois plus rapides sur NP2004 et HP2004. Nos approches sont ainsi plus efficaces que l'état de l'art pour supprimer un grand nombre de variables dans des temps de calcul raisonnables.

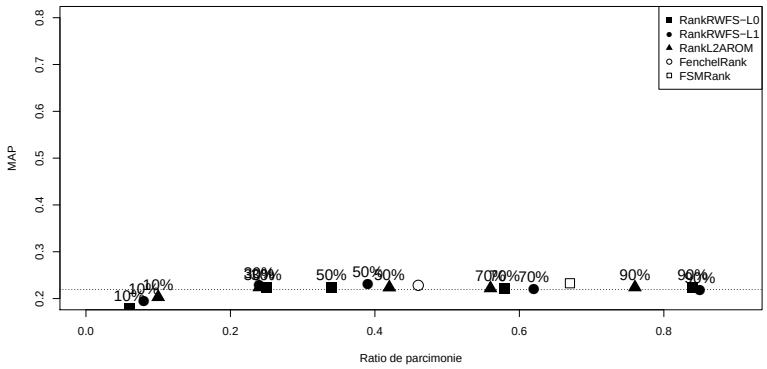
Les algorithmes que nous proposons constituent donc des approches performantes et compétitives pour sélectionner un grand nombre de variables dans des temps compétitifs, tout en maintenant une bonne qualité d'ordonnancement.



(a) HP2004

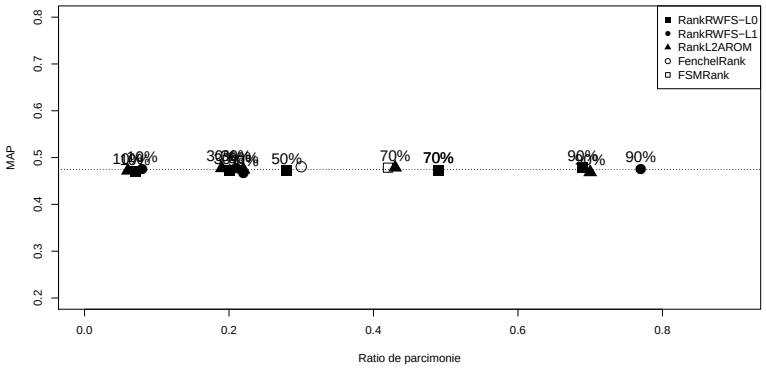


(b) NP2004

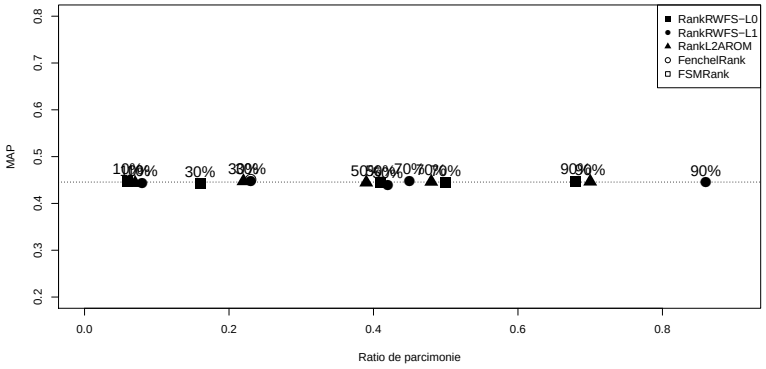


(c) TD2004

FIGURE 6.9 – Lien entre ratio de parcimonie et MAP pour les différents algorithmes sur les jeux de données HP2004, NP2004 et TD2004. Nos méthodes sont généralement les meilleures pour des seuils de sélection théoriques inférieurs à 50 %.

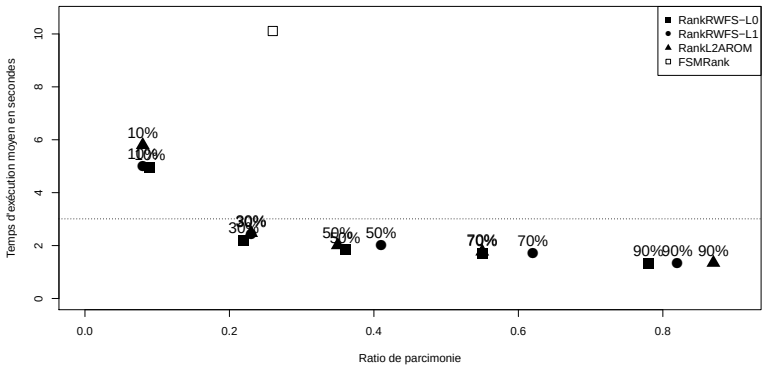


(a) MQ2008

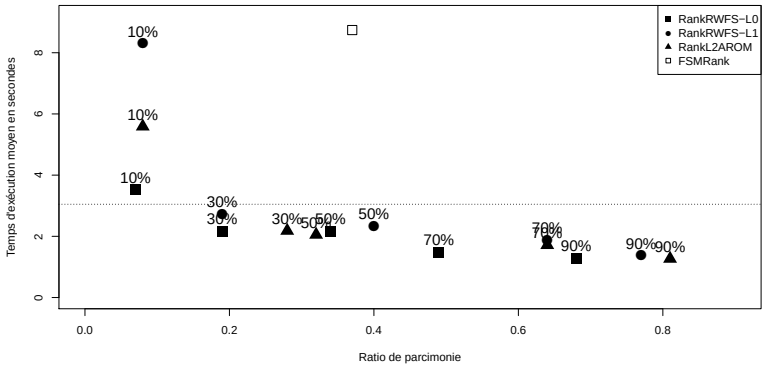


(b) Ohsumed

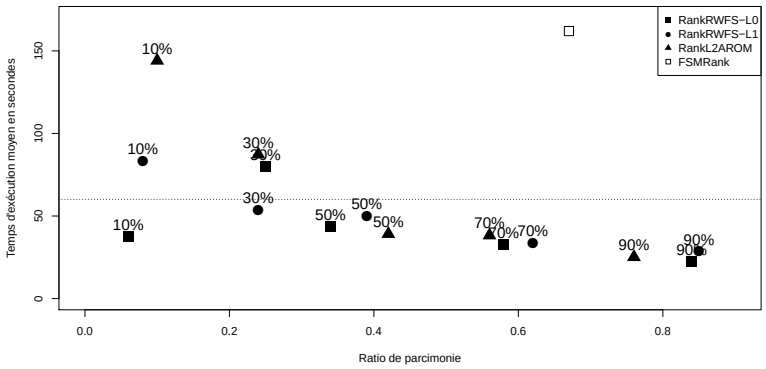
FIGURE 6.10 – Lien entre ratio de parcimonie et MAP pour les différents algorithmes sur les jeux de données MQ2008 et Ohsumed. Nos méthodes sont généralement les meilleures pour des seuils de sélection théoriques inférieurs à 50 %.



(a) HP2004

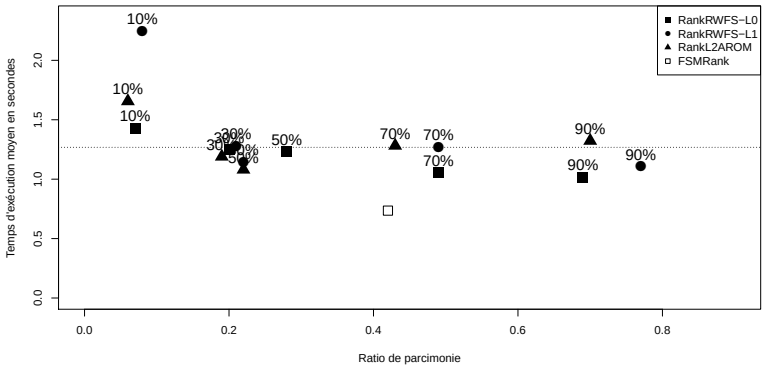


(b) NP2004

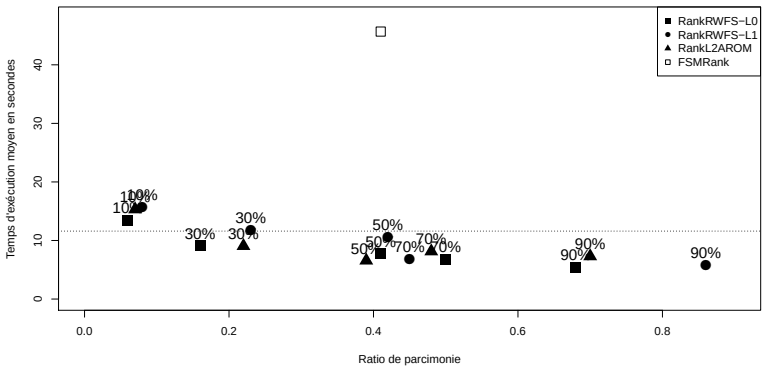


(c) TD2004

FIGURE 6.11 – Lien entre ratio de parcimonie et temps d'exécution pour les différents algorithmes sur les jeux de données HP2004, NP2004 et TD2004. Nos méthodes sont jusqu'à 5 fois plus rapides que l'état de l'art.



(a) MQ2008



(b) Ohsumed

FIGURE 6.12 – Lien entre ratio de parcimonie et temps d'exécution pour les différents algorithmes sur les jeux de données MQ2008 et Ohsumed. Nos méthodes sont généralement les meilleures pour des seuils de sélection théoriques inférieurs à 50 %.

6.4.6 Des sous-ensembles de variables sélectionnées cohérents et informatifs

Les analyses précédentes portaient sur une évaluation quantitative des approches proposées : analyse des ratios de parcimonie, comparaison des mesures d'évaluation et analyse des temps d'exécution. Nous proposons maintenant une étude orientée vers l'évaluation de la qualité des approches de sélection. Dans ce contexte, nous cherchons à évaluer deux aspects de la qualité des approches :

1. La capacité des trois algorithmes à sélectionner des sous-ensembles similaires sur un même jeu de données ;

2. La capacité des algorithmes à sélectionner des sous-ensembles de caractéristiques reconnues comme informatives.

Par souci de simplicité, nous nous sommes limités à l'analyse des modèles comprenant moins de 10 % des variables initiales.

Notons $\mathcal{S}_{\text{RankRWFS-}\ell_0}$ (respectivement $\mathcal{S}_{\text{RankRWFS-}\ell_1}$ et $\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$), la réunion des sous-ensembles de caractéristiques sélectionnées sur chaque sous-échantillon par l'algorithme RankRWFS-ℓ_0 (respectivement RankRWFS-ℓ_1 et Rankℓ_2-AROM). Les tableaux 6.1 à 6.5 présentent, pour chaque couple d'ensembles (X,Y), le ratio du nombre d'éléments de X présents dans Y et du nombre total d'éléments de X, pour le jeu de données considéré. Nous observons que, sur l'ensemble des jeux de données, les algorithmes sélectionnent des sous-ensembles de caractéristiques similaires voire identiques. Ainsi, le pourcentage minimal de variables communes entre deux sous-ensembles, sur tous les jeux de données, est de 50 %. Les modèles partagent fréquemment plus de 75 % de leur caractéristiques avec un autre modèle. Nous constatons par ailleurs que :

- sur HP2004, $\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}} \subset \mathcal{S}_{\text{RankRWFS-}\ell_0}$, tandis que $\mathcal{S}_{\text{RankRWFS-}\ell_0}$ et $\mathcal{S}_{\text{RankRWFS-}\ell_1}$ partagent environ trois caractéristiques sur quatre ;
- sur NP2004, $\mathcal{S}_{\text{RankRWFS}\ell_0} \subset \mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$ et ses deux sous-ensembles partagent respectivement 89 % et 80 % de leurs éléments avec $\mathcal{S}_{\text{RankRWFS-}\ell_1}$;
- sur TD2004, les sous-ensembles sont emboités tel que $\mathcal{S}_{\text{RankRWFS}\ell_0} \subset \mathcal{S}_{\text{RankRWFS-}\ell_1} \subset \mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$;
- sur MQ2008, $\mathcal{S}_{\text{RankRWFS-}\ell_0} = \mathcal{S}_{\text{RankRWFS-}\ell_1}$ et $\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$ est inclus dans ces deux sous-ensembles ;
- sur Ohsumed, $\mathcal{S}_{\text{RankRWFS-}\ell_0}$ est inclus dans les deux autres sous-ensembles, qui partagent cinq caractéristiques sur six.

Les algorithmes sélectionnent ainsi des sous-ensembles de variables qui sont très similaires, du moins pour le seuil observé. L'accord inter-algorithmes concernant les caractéristiques pertinentes pour un jeu de données est donc élevé.

	$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$
$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	-	1	1
$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	0.67	-	0.5
$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$	0.5	0.75	-

TABLE 6.3 – Ratio de variables communes pour les sous-ensembles pris deux à deux sur TD2004

	$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$
$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	-	1	0.75
$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	1	-	0.75
$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$	1	1	-

TABLE 6.4 – Ratio de variables communes pour les sous-ensembles pris deux à deux sur MQ2008

	$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$
$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	-	0.69	0.77
$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	0.75	-	0.67
$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$	1	0.8	-

TABLE 6.1 – Ratio de variables communes pour les sous-ensembles pris deux à deux sur HP2004

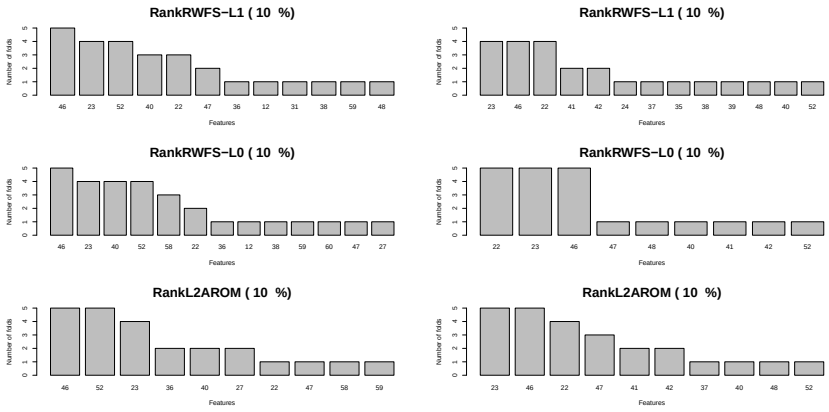
	$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$
$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	-	0.89	1
$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	0.62	-	0.69
$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$	0.9	0.9	-

TABLE 6.2 – Ratio de variables communes pour les sous-ensembles pris deux à deux sur NP2004

Nous souhaitons évaluer la capacité des algorithmes à sélectionner des caractéristiques informatives. La figure 6.13 présente les diagrammes en barres du nombre de sous-ensembles pour lesquels chaque caractéristique a été sélectionnée, ce pour chaque algorithme et chaque jeu de données. Nous nous intéressons principalement aux caractéristiques qui ont été sélectionnées le plus fréquemment sur l'ensemble des cinq sous-échantillons de chaque jeu de données.

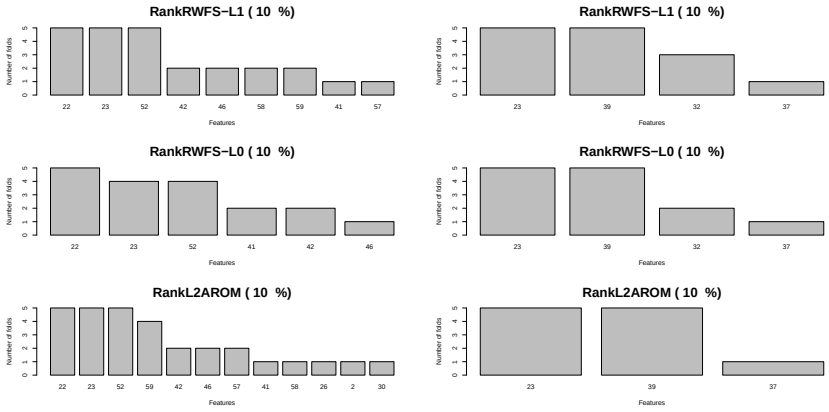
	$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$
$\mathcal{S}_{\text{RankRWFS-}\ell_0}$	-	1	1
$\mathcal{S}_{\text{RankRWFS-}\ell_1}$	0.83	-	0.83
$\mathcal{S}_{\text{Rank}\ell_2\text{-AROM}}$	0.83	0.83	-

TABLE 6.5 – Ratio de variables communes pour les sous-ensembles pris deux à deux sur Ohsumed



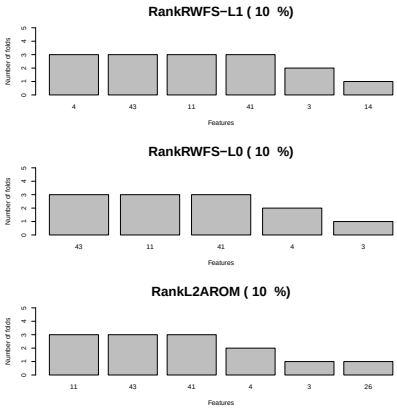
(a) HP2004

(b) NP2004



(c) TD2004

(d) MQ2008



(e) Ohsumed

FIGURE 6.13 – Sous-ensembles de caractéristiques sélectionnées au seuil de 10 % pour les trois algorithmes sur les jeux de données (a) HP2004, (b) NP2004, (c) TD2004, (d) MQ2008 et (e) Ohsumed.

Nous observons que sur les jeux de données HP2004, NP2004 et TD2004, les algorithmes ont sélectionné le plus fréquemment les variables d'indices 22, 23, 40, 46 et 52. Les caractéristiques 22 et 23 correspondent au score BM25 sur le titre et l'ancre du document. La variable 40 est basée sur un modèle de langue appris sur l'ensemble du document, tandis que la 46 est basée sur la notion de propagation des hyperliens. Enfin, la dernière variable correspond au HostRank. Il est intéressant de noter que, d'une part, ces caractéristiques sont considérées comme hautement informatives et que d'autre part, elles concordent avec les caractéristiques sélectionnées par d'autres algorithmes de l'état de l'art [Lai 2013a]. Sur Ohsumed, les algorithmes ont majoritairement sélectionné les caractéristiques 4, 11, 41 et 43, qui sont respectivement basées sur la fréquence des termes communs entre requête et document (tf), le score BM25 et les modèles de langues. Enfin, sur MQ2008, les variables 23 et 39 sont les plus fréquemment sélectionnées et correspondent à des modèles de langues. Ainsi, les méthodes proposées sélectionnent sur tous les jeux de données des variables hautement informatives.

Les algorithmes tendent à conserver majoritairement des caractéristiques de type BM25 ou modèles de langues, qui sont connues pour être hautement informatives pour l'apprentissage d'ordonnancement.

6.5 Conclusion et perspectives

Dans ce chapitre, nous avons présenté des travaux qui apportent différentes contributions dans le domaine de la sélection de variables en apprentissage d'ordonnancement.

Tout d'abord, nous proposons une nouvelle méthodologie, basée sur une approche IRLS, capable de résoudre des problèmes parcimonieux en norme ℓ_1 et en norme ℓ_0. A notre connaissance, ce sont les premiers travaux sur les SVM parcimonieux en sélection de variables pour l'apprentissage d'ordonnancement qui ne se limitent pas à l'utilisation d'une même norme parcimonieuse.

Ensuite, à notre connaissance, ce sont les premiers travaux à étudier l'apport des SVM en norme ℓ_0 pour la sélection de variables en apprentissage d'ordonnancement. Les études existantes [Sun 2009, Lai 2013a] se sont concentrées sur l'utilisation de la norme ℓ_1. Nous montrons que l'approche en norme ℓ_0 obtient des résultats comparables en matière de qualité d'ordonnancement, mais semble plus efficace du point de vue de la vitesse de calcul.

De plus, ces travaux proposent une analyse comparative poussée de différentes approches itératives repondérées, ce qui constitue une nouveauté dans le domaine de la sélection de variables pour l'apprentissage d'ordonnancement. Les approches IRLS n'ont, à notre connaissance, pas ou peu été étudiées dans ce contexte. Les approches proposées obtiennent des résultats comparables ou meilleurs que l'état de l'art lorsque l'on considère les mesures d'évaluation. Par ailleurs, les temps d'exécution sont raisonnables. Les temps de calcul obtenus avec la méthode RankRWFS-ℓ_0 sont meilleurs que les autres algorithmes, ce qui indique l'intérêt de la méthodologie

et de l'approche RWFS proposées.

Enfin, nous fournissons une analyse qualitative des modèles appris. Nous montrons que les approches proposées présentent l'intérêt de sélectionner des sous-ensembles de caractéristiques similaires et hautement informatives. Nous sommes ainsi en mesure d'isoler des sous-ensembles de variables hautement pertinentes pour l'apprentissage d'ordonnancement, pour chaque jeu de données de référence.

Les approches proposées et notamment la méthodologie RWFS apparaissent comme des méthodes prometteuses pour la sélection de variables en apprentissage d'ordonnancement. Dans nos travaux futurs, nous nous concentrerons sur la proposition d'approches IRLS en norme ℓ_0 qui ont été encore peu étudiées dans ce contexte. Notamment, nous envisageons d'incorporer d'autres approximations de la norme ℓ_0 à la méthodologie IRLS. Les approximations 6.2 et 6.3 seront ainsi étudiées.

Les approches proposées présentent un inconvénient : le norme ℓ_2 n'étant pas parcimonieuse, les coefficients tendent vers zéro, mais n'atteignent jamais totalement la valeur nulle. Il est donc nécessaire d'imposer un seuil en deçà duquel les coefficients sont fixés à zéro. Ce problème est résolu en considérant non pas une repondération de la norme ℓ_2 mais une repondération de la norme ℓ_1.

Dans de futurs travaux, nous nous intéresserons à une adaptation de l'approximation de la norme ℓ_0 par la norme ℓ_1 telle que proposée par Candès *et al.* [Candes 2008]. Nous proposons de modifier l'approche RWFS afin d'intégrer cette approximation comme présentée dans l'algorithme 6.5. La structure de l'algorithme reste similaire à celle des algorithmes RankRWFS-ℓ_1 et RankRWFS-ℓ_0, seuls la règle de mise à jour des poids et l'algorithme de résolution des SVM sont modifiés. La règle de mise à jour du vecteur $\mathbf{v}$ à chaque itération t devient ainsi $\mathbf{v}_i^{(t+1)} = \sqrt{|(\mathbf{w}_i^{(t)} + \varepsilon)\mathbf{v}_i^{(t)}|}$ où ε tend vers zéro. L'algorithme de résolution des SVM utilisés doit pouvoir résoudre le problème en norme ℓ_1. Nous pourrons par exemple utiliser l'algorithme FenchelRank, ou l'algorithme proximal proposé au chapitre 7.

Nous avons également mené une étude concernant, non pas l'utilisation et l'approximation de la norme ℓ_0, mais l'utilisation d'autres normes non convexes. L'intérêt est de considérer des normes plus parcimonieuses que la norme ℓ_1 et pouvant être approchée par repondération de cette dernière. Nous présentons dans le chapitre suivant une méthode de repondération de la norme ℓ_1 pour l'approximation de régularisations non convexes et montrons que nous obtenons d'excellents résultats dans ce contexte.

Algorithme 6.5 Algorithme Rank-RW-$\ell_1\ell_0$

Entrée : jeu de données d'apprentissage $(\mathbf{x}^j, \mathbf{y}^j)_{j=1}^m$ et nombre maximal de variables r

Sortie : vecteur de poids $\mathbf{w}$

%% Etape de sélection des variables
Initialisation $\mathbf{v} = [1...1] \in \mathbb{R}^d$, $t = 1$
Tant que $\|\mathbf{w}\|_0 > r$ **Faire**
Pour chaque observation $\mathbf{x}^j$ **Faire**
Pour $i = 1 \rightarrow d$ **Faire**
$\mathbf{x}_i^{'j} \leftarrow \mathbf{x}_i^j \mathbf{v}_i^{(t)}$
Fin Pour
Fin Pour
$\mathbf{w}^{(t)} \leftarrow$ solution de RankSVM-$\ell_1(\{(\mathbf{x}^{'j}, \mathbf{y}^j)\}_{j=1}^m)$
Pour $i = 1 \rightarrow d$ **Faire**
$\mathbf{v}_i^{(t+1)} \leftarrow \sqrt{|(\mathbf{w}_i^{(t)} + \varepsilon)\mathbf{v}_i^{(t)}|}$
Fin Pour
Pour $i = 1 \rightarrow d$ **Faire**
$\mathbf{w}_i \leftarrow \mathbf{w}_i^{(t)} \mathbf{v}_i^{(t)}$
Fin Pour
$t \leftarrow t + 1$
Fin Tant que
%% Récupération des variables pertinentes
$I \leftarrow \{i | \mathbf{w}_i \neq 0\}$
%% Apprentissage du modèle final
Pour $j = 1 \rightarrow m$ **Faire**
Pour $i \in I$ **Faire**
$\mathbf{z}_i^j \leftarrow \mathbf{x}_i^j$
Fin Pour
Fin Pour
$\mathbf{w} \leftarrow$ solution de RankSVM-Primal$(\{(\mathbf{z}^j, \mathbf{y}^j)\}_{j=1}^m)$
Retourner $\mathbf{w}$

Chapitre 7

Une approche proximale pour la sélection de variables via des SVM parcimonieux : proposition, principe et évaluation

Sommaire

Dans le chapitre précédent, nous avons proposé différents algorithmes de sélection de variables dédiés à l'apprentissage d'ordonnancement. Ces algorithmes approchent la norme ℓ_0 par repondération de la norme ℓ_2 et donnent de bons résultats, que nous considérions la qualité de prédiction ou la parcimonie. Néanmoins, ces approches présentent un inconvénient : la norme ℓ_2 n'est pas naturellement parcimonieuse. Ainsi, les coefficients n'atteignent jamais exactement zéro, même s'ils tendent vers la valeur nulle. Nous sommes contraints de fixer un seuil qui annulent les valeurs suffisamment petites.

Dans ce chapitre, nous proposons d'utiliser les régularisations non convexes présentées au chapitre 5 afin d'introduire de la parcimonie dans les modèles. Nous considérons une approche basée sur l'utilisation d'un schéma en norme ℓ_1 afin de résoudre le problème d'optimisation dans le cadre des régularisations non convexes.

Dans la section 7.1, nous proposons deux algorithmes *embedded* pour la sélection de variables en apprentissage d'ordonnancement.

Le premier, présenté à la section 7.1.1, est un algorithme de type FBS qui résout un problème d'ordonnancement par paire avec une régularisation ℓ_1. Le principe de ce type de méthodes, appelées aussi approches proximales, est tout d'abord présenté dans le cadre de la classification. Puis nous décrivons son adaptation dans le cadre d'apprentissage par paire. Enfin, nous présentons en détail l'algorithme proposé, que nous nommons RankSVM-ℓ_1.

Le second algorithme proposé, présenté à la section 7.1.2, repondère RankSVM-ℓ_1 afin de résoudre un problème d'ordonnancement par paire faisant intervenir des régularisations non convexes définies au chapitre 5 (MCP, *log* et ℓ_q, $q < 1$). Nous le nommons RankSVM-NC. L'algorithme est basé sur l'utilisation d'un schéma de repondération en norme ℓ_1 que nous détaillons. A notre connaissance, l'utilisation de régularisations non convexes n'a pas été étudiée dans le cadre de la sélection de variables en apprentissage d'ordonnancement en RI. Il s'agit donc du premier algorithme à traiter cette problématique.

Dans la section 7.2, nous évaluons l'algorithme RankSVM-ℓ_1 sur les neuf jeux de données issus de LETOR 3 et 4 en utilisant la MAP et le NDCG@10. Nous comparons les résultats obtenus à ceux de six algorithmes non parcimonieux de l'état de l'art. Nous analysons également les ratios de parcimonie. Une comparaison de la performance de RankSVM-ℓ_1 par rapport aux approches de sélection de variables de l'état de l'art et de l'algorithme utilisant les régularisations non convexes est effectuée à la section 7.3.

Dans la section 7.3, nous évaluons la performance de RankSVM-NC sur les neuf jeux de données issus de LETOR 3 et 4 en utilisant la MAP et le NDCG@10 ainsi que les ratios de parcimonie. Nous considérons les régularisations non-convexes MCP, *log* et ℓ_q, $q < 1$. Nous comparons les résultats obtenus pour chaque régularisation aux algorithmes RankSVM-ℓ_1, FenchelRank et FSMRank. Nous montrons que les régularisations non convexes sont performantes pour effectuer la sélection de variables, puisque nous obtenons des ratios de parcimonie jusqu'à six fois plus faibles qu'en utilisant des régularisations convexes, tout en préservant la qualité de l'ordonnancement.

Ces travaux ont été réalisés en collaboration avec Rémi Flamary du laboratoire Lagrange de l'Université de Nice-Sophia Antipolis et Stéphane Canu du Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes (LITIS) de l'INSA de Rouen. Ils ont été acceptés pour publication dans la revue internationale *IEEE Transactions on Neural Networks and Learning Systems* sous le titre "Non-regularizations for feature selection in ranking with sparse SVM" [Laporte 2014].

7.1 Algorithmes proximaux pour régularisations convexes et non-convexes

Dans cette section, nous présentons les deux algorithmes que nous proposons dans le cadre de la sélection de variables en apprentissage d'ordonnancement.

Dans un premier temps, nous introduisons le principe des approches FBS et montrons comment nous adaptons l'algorithme original dans le cadre de l'apprentissage d'ordonnancement. Nous nommons l'algorithme ainsi obtenu RankSVM-ℓ_1.

Dans un second temps, nous proposons un algorithme de repondération en norme ℓ_1 permettant l'utilisation de régularisations non-convexes telles que les pénalités $\ell_p, p < 1$, *log* et MCP. Notons que contrairement aux approches du chapitre 6, il n'est pas nécessaire de fixer de seuil de variables à atteindre. A notre connaissance, il s'agit du premier algorithme proposé en sélection de variables pour l'apprentissage d'ordonnancement qui considère l'utilisation de régularisations non convexes.

7.1.1 RankSVM-ℓ_1, un algorithme de résolution de régularisation convexe

RankSVM-ℓ_1 constitue la brique de base nécessaire pour la résolution du problème d'optimisation avec des régularisations non convexes. Nous proposons d'utiliser une approche de type FBS. Les approches ont été conçues pour résoudre des problèmes d'optimisation non différentiables de forme similaire à celui que nous considérons. Elles paraissent donc adaptées. Dans la suite de cette section, nous présentons le principe des algorithmes FBS dans le cadre de la discrimination, puis nous montrons comment il est adapté à l'apprentissage d'ordonnancement.

7.1.1.1 Principe des algorithmes FBS

Nous présentons ici les algorithmes FBS de façon succincte. Une présentation détaillée de ce type d'algorithmes est disponible dans l'ouvrage de Bach *et al.* [Bach 2011].

Les algorithmes FBS ont été initialement proposés pour résoudre des problèmes d'optimisation non différentiables de la forme :

$$\min_{\mathbf{w}\in\mathbb{R}^d} J_1(\mathbf{w}) + \lambda\Omega(\mathbf{w}) \tag{7.1}$$

où $J_1(.)$ est une fonction différentiable à gradient lipschitzien continu et $\Omega(.)$ est un terme de régularisation convexe possédant un opérateur proximal. L'opérateur proximal de la régularisation $\mu\Omega(.)$, noté $Prox_{\mu\Omega}(.)$, est défini de la façon suivante :

$$Prox_{\mu\Omega}(\mathbf{z}) = \arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{z}-\mathbf{w}\|_2^2 + \mu\Omega(\mathbf{w}) \tag{7.2}$$

Ces algorithmes fonctionnent de manière itérative. L'idée générale de l'approche est que le terme $J_1(.)$ peut être linéarisé à chaque itération de sorte que le problème

Algorithme 7.1 Algorithme FSB accéléré (FISTA)

Initialiser $\mathbf{w}^0$
Initialiser L comme constante de Lipschitz de $\nabla J_1(\cdot)$
$k = 1$, $\mathbf{z}^1 = \mathbf{w}^0$, $t^1 = 1$
repeat
  $\mathbf{w}^k \leftarrow \text{Prox}_{\frac{\lambda}{L}\Omega}(\mathbf{z}^k - \frac{1}{L}\nabla J_1(\mathbf{z}^k))$
  $t^{k+1} \leftarrow \frac{1+\sqrt{1+4(t^k)^2}}{2}$
  $\mathbf{z}^{k+1} \leftarrow \mathbf{w}^k + \left(\frac{t^k-1}{t^{k+1}}\right)(\mathbf{w}^k - \mathbf{w}^{k-1})$
  $k \leftarrow k + 1$
until Convergence

7.1 puisse être approché par un autre problème correspondant à l'expression de l'opérateur proximal. Résoudre le problème 7.1 équivaut à résoudre une succession de problèmes de la forme 7.2, tel que, à chaque itération k, le vecteur de poids $\mathbf{w}$ est mis à jour de la façon suivante :

$$\mathbf{w}^{k+1} = Prox_{\frac{\lambda}{L}\Omega}\left(\mathbf{w}^k - \frac{1}{L}\nabla J_1(\mathbf{w}^k)\right) \tag{7.3}$$

où L est une constante de Lipschitz qui assure la convergence de la méthode.

Dans ces travaux, nous nous intéressons plus particulièrement aux régularisations en norme ℓ_1 et ℓ_1 repondérée, qui possèdent chacune un opérateur proximal. L'opérateur proximal de la norme ℓ_1, noté $Prox_{\lambda\|.\|_1}$, est défini de la façon suivante :

$$\begin{aligned} Prox_{\lambda\|.\|_1}(\mathbf{w})_i &= \max\left(0, 1 - \frac{\lambda}{|\mathbf{w}_i|}\right)\mathbf{w}_i \\ &= sign(\mathbf{w}_i)(|\mathbf{w}_i| - \lambda)_+, \forall i \in 1, \ldots, d \end{aligned} \tag{7.4}$$

L'opérateur proximal de la norme ℓ_1 repondérée, noté $Prox_{\lambda\Omega_\beta}$, est défini de façon similaire, tel que :

$$\begin{aligned} Prox_{\lambda\Omega_\beta}(\mathbf{w})_i &= \max\left(0, 1 - \frac{\lambda\beta_i}{|\mathbf{w}_i|}\right)\mathbf{w}_i \\ &= sign(\mathbf{w}_i)(|\mathbf{w}_i| - \lambda\beta_i)_+, \forall i \in 1, \ldots, d \end{aligned} \tag{7.5}$$

L'un des inconvénients des algorithmes FBS est leur vitesse de convergence relativement lente. Beck et Teboulle [Beck 2009] ont ainsi proposé une adaptation de l'algorithme FBS initial qui assure une meilleure vitesse de convergence. Nommé Fast Iterative Shrinkage Thresholding Algorithm (FISTA) et présenté à l'algorithme 7.1, il s'agit d'une version multi-pas de l'algorithme initial. Au lieu de mettre à jour le vecteur $\mathbf{w}^{k+1}$ à partir de la valeur courante $\mathbf{w}^k$, la combinaison linéaire de la valeur courante et de la valeur antérieure $\mathbf{w}^k - \mathbf{w}^{k-1}$ est utilisée.

7.1.1.2 RankSVM-ℓ_1, un algorithme FBS pour l'apprentissage de préférences parcimonieux

Nous proposons d'adapter l'algorithme 7.1 à l'apprentissage d'ordonnancement de préférences dans le cadre des SVM parcimonieux. Cela peut être réalisé de façon quasi-directe.

En effet, nous observons que le problème d'optimisation des SVM pour l'apprentissage de préférences peut s'écrire sous la forme généralisée 7.1, telle que $J_1(\mathbf{w}) = \sum_{p=1}^{P} \max(0, 1 - \tilde{\mathbf{x}}_p^\top \mathbf{w})^2$, $\Omega(\mathbf{w}) = \|\mathbf{w}\|_1$ et $\lambda = \frac{1}{C}$. Pour pouvoir utiliser l'algorithme FISTA et en assurer la convergence, il est nécessaire de prouver la proposition 1, c'est-à-dire que le terme $J_1(\mathbf{w})$ possède un gradient lipschitzien continu. Cette démonstration est présentée dans notre article [Laporte 2014]. Elle est reproduite ci-après.

Proposition 1.

Soit $J_1(\mathbf{w})$ la fonction de perte Hinge au carré définie de la façon suivante :

$$J_1(\mathbf{w}) = \sum_{p=1}^{P} \max(0, 1 - \tilde{\mathbf{x}}_p^\top \mathbf{w})^2$$

Alors son gradient $\nabla J_1(\mathbf{w})$

$$\nabla J_1(\mathbf{w}) = -2 \sum_{p=1}^{P} \tilde{\mathbf{x}}_p \max(0, 1 - \tilde{\mathbf{x}}_p^\top \mathbf{w})$$

est lipschitzien continu.

Preuve Par définition, la fonction de perte Hinge au carré possède un gradient lipschitzien continu s'il existe une constante L telle que :

$$\|\nabla J_1(\mathbf{w}_1) - \nabla J_1(\mathbf{w}_2)\|_2 \leq L\|\mathbf{w}_1 - \mathbf{w}_2\|_2 \quad \forall \mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^d.$$

L'idée de la preuve est de montrer que le terme à l'intérieur de la somme, soit $\tilde{\mathbf{x}}_i \max(0, 1 - \tilde{\mathbf{x}}_i^\top \mathbf{w})$, est lui-même lipschitzien. Par extension, le gradient, qui est la somme de termes lipschitzien, est lipschitzien. En pratique, il suffit de montrer qu'il existe une constante $L' \in \mathbb{R}$ telle que :

$$\|\tilde{\mathbf{x}}_i \max(0, 1 - \tilde{\mathbf{x}}_i^\top \mathbf{w}_1) - \tilde{\mathbf{x}}_i \max(0, 1 - \tilde{\mathbf{x}}_i^\top \mathbf{w}_2)\| \leq L'\|\mathbf{w}_1 - \mathbf{w}_2\|$$

Considérons $\mathbf{w}_1$ and $\mathbf{w}_2$, tels que $1 - \tilde{\mathbf{x}}_i^T \mathbf{w}_1 \leq 0$ et $1 - \tilde{\mathbf{x}}_i^T \mathbf{w}_2 \leq 0$, alors le terme de gauche de l'inégalité précédente est égal à 0 et n'importe quel L' satisfait l'inégalité. Considérons à présent $\mathbf{w}_1$ and $\mathbf{w}_2$, tels que $1 - \tilde{\mathbf{x}}_i^T \mathbf{w}_1 \leq 0$ et $1 - \tilde{\mathbf{x}}_i^T \mathbf{w}_2 \geq 0$, alors le

terme de gauche de l'inégalité précédente s'écrit de la façon suivante :

$$\begin{aligned} lhs &= \|\tilde{\mathbf{x}}_i\|_2(1-\tilde{\mathbf{x}}_i^\top \mathbf{w}_2) \\ &\leq \|\tilde{\mathbf{x}}_i\|_2(\tilde{\mathbf{x}}_i^\top \mathbf{w}_1 - \tilde{\mathbf{x}}_i^\top \mathbf{w}_2) \\ &\leq \|\tilde{\mathbf{x}}_i\|_2^2\|\mathbf{w}_1-\mathbf{w}_2\|_2 \end{aligned}$$

Un raisonnement similaire quand $1-\tilde{\mathbf{x}}_i^T\mathbf{w}_1 \geq 0$ et $1-\tilde{\mathbf{x}}_i^T\mathbf{w}_2 \leq 0$ et quand $1-\tilde{\mathbf{x}}_i^T\mathbf{w}_1 \geq 0$ et $1-\tilde{\mathbf{x}}_i^T\mathbf{w}_2 \geq 0$ montre que $\tilde{\mathbf{x}}_i \max(0, 1-\tilde{\mathbf{x}}_i^\top \mathbf{w})$ est lipschitzien avec pour constante $\|\tilde{\mathbf{x}}_i\|^2$.

La preuve se conclue en disant que $\nabla_{\mathbf{w}} J$ est lipschitzien en tant que somme de fonctions lipschitziennes. La constante de Lipschitz correspondante est $\sum_{i=1}^n \|\tilde{\mathbf{x}}_i\|_2^2$. □

Nous avons donc prouvé que le terme $J_1(\mathbf{w})$ est lipschitzien : l'algorithme FISTA peut être utilisé pour résoudre le problème en norme ℓ_1. Notons que la preuve est similaire dans le cadre de la régularisation en norme ℓ_1 repondérée. Les gradients correspondants peuvent donc être utilisés directement par l'algorithme.

7.1.2 Algorithme repondéré pour les régularisations non-convexes

L'algorithme que nous proposons dans la section précédente constitue une nouvelle approche pour la sélection de variables en apprentissage d'ordonnancement. Néanmoins, l'utilisation de la norme ℓ_1 dans ce contexte n'est pas nouvelle. Elle a notamment été étudiée par Lai *et al.* [Lai 2013a] dans le cadre de l'algorithme FenchelRank, bien que l'algorithme en question résolve le problème d'une façon différente de celle présentée ici.

A notre connaissance, l'apport des régularisations non-convexes pour la sélection de variables en apprentissage d'ordonnancement de préférences en RI n'a jamais été étudié. Nous proposons ici un nouvel algorithme permettant de considérer des normes non-convexes. Nous le nommons RankSVM-NC

Néanmoins, les algorithmes FBS ne peuvent pas être utilisés avec des régularisations non convexes, celles-ci n'ayant pas de gradient lipschitzien continu. Nous adaptons une méthode proposée par Gasso *et al.* [Gasso 2009], que nous avons introduite au chapitre 5 et qui repose sur l'utilisation de la norme ℓ_1 repondérée. Considérons le problème 7.1 :

$$\min_{\mathbf{w}\in\mathbb{R}^d} J_1(\mathbf{w}) + \lambda\Omega(\mathbf{w})$$

où dans notre cas $J_1(\mathbf{w})$ est la régularisation Hinge au carré et $\Omega(\mathbf{w})$ est une régularisation non convexe introduite au chapitre 5, qui n'a donc pas de gradient lipschitzien. En se plaçant dans le cadre de la méthode proposée par Gasso *et al.* [Gasso 2009], $\Omega = \sum_{i=1}^d g(|\mathbf{w}_i|)$ où $g(.)$ est une fonction non convexe, l'idée est alors de se ramener à un problème de Majoration-Minimisation [Hunter 2004] en constatant que les régularisations non convexes sont concaves lorsque toutes leurs coordonnées sont positives.

Algorithme 7.2 Algorithme pour les régularisations non-convexes

1: Initialiser $\mathbf{w}^0$ et $k = 1$
2: Initialiser $\beta_j = 1, \forall_j$
3: **repeat**
4: $\quad \mathbf{w}^{k+1} \leftarrow \arg\min_{\mathbf{w}} J_1(\mathbf{w}) + \lambda \sum_{i=1}^{d} \beta_i |\mathbf{w}_i|$ à l'aide de l'algorithme 7.1.
5: $\quad \beta_j \leftarrow g'(|\mathbf{w}_j^k|), \forall j$
6: $\quad k \leftarrow k + 1$
7: **until** Convergence

Ainsi, pour tout point $x_0 > 0$ fixé et tout point $x > 0$ à déterminer, la fonction de perte $g(.)$ peut être majorée de la façon suivante :

$$g(x) \leq g(x_0) + g'(x_0)(x - x_0) \tag{7.6}$$

En considérant $\mathbf{x} = |\mathbf{w}_i|$, $x_0 = |\mathbf{w}_i^*|$ valeur connue de $\mathbf{w}_i$ à l'itération k et en négligeant la partie constante dans l'équation précédente, le terme de régularisation non convexe peut être majoré par une somme pondérée de termes convexes telle que :

$$\sum_{i=1}^{d} g(|\mathbf{w}_i|) \leq \sum_{i=1}^{d} g'(|\mathbf{w}_i|)|\mathbf{w}_i| + g(|\mathbf{w}_i^*|) - g'(|\mathbf{w}_i^*|)|\mathbf{w}_i^*| \tag{7.7}$$

où le terme $g(|\mathbf{w}_i^*|) - g'(|\mathbf{w}_i^*|)|\mathbf{w}_i^*|$ est constant. Résoudre le problème avec les régularisations non convexes est donc équivalent à résoudre de façon itérative une succession de problèmes en norme ℓ_1 repondérée qui minimise la majoration précédente, pour laquelle le terme constant peut alors être omis. A chaque itération $k + 1$, le vecteur de poids $\mathbf{w}$ est alors mis à jour de la façon suivante :

$$\mathbf{w}^{k+1} = \arg\min_{\mathbf{w}} J_1(\mathbf{w}) + \lambda \sum_{i=1}^{d} \beta_i |\mathbf{w}_i| \tag{7.8}$$

où $\beta_i = g'(|w_i^*|)$. Les poids sont calculés en utilisant la solution trouvée à l'itération précédente grâce à l'algorithme RankSVM-ℓ_1. La méthode finale utilisée pour la résolution dans le cas non-convexe est présentée dans l'algorithme 7.2. Cette approche est particulièrement intéressante puisqu'elle nous permet de résoudre le problème non-convexe en utilisant l'algorithme en norme ℓ_1 repondérée de façon quasiment directe. Par ailleurs, nous pouvons initialiser à chaud l'algorithme au début de chaque itération. De cette manière, l'algorithme non-convexe hérite des bonnes propriétés de convergence de la méthode FISTA et n'est pas trop pénalisé pas l'ajout des boucles sur l'algorithme RankSVM-ℓ_1.

Dans les chapitres suivants, nous présentons les résultats et propriétés expérimentales des algorithmes sur les jeux de données de référence. Nous montrons que nos approches donnent d'excellents résultats que nous considérions la qualité de prédiction ou la capacité à introduire de la parcimonie dans les modèles.

7.2 RankSVM-ℓ_1 : un algorithme de sélection performant

Dans cette section, nous présentons une évaluation de l'algorithme RankSVM-ℓ_1, que nous proposons et qui constitue la brique de base de notre approche. L'objectif premier de cette analyse est de montrer la compétitivité de RankSVM-ℓ_1, comparativement à des approches non parcimonieuses. Le protocole expérimental est d'abord détaillé, puis les résultats obtenus sont discutés.

7.2.1 Protocole expérimental

L'objectif de l'expérimentation est de montrer la compétitivité de RankSVM-ℓ_1 en tant qu'algorithme de résolution parcimonieux en norme ℓ_1, comparativement aux méthodes de référence non parcimonieuses. Nous nous intéressons non seulement à la qualité de prédiction, mesurée à l'aide de la MAP et du NDCG@10, mais aussi à la capacité de l'algorithme à introduire de la parcimonie dans les modèles. Cette dernière est mesurée à l'aide du ratio de parcimonie, défini au chapitre 6.

Nous évaluons RankSVM-ℓ_1 sur l'ensemble des neuf jeux de données des collections LETOR 3.0 et 4.0. Sur chacun d'eux, nous calculons les valeurs de MAP et de NDCG@10 et les comparons aux performances de six algorithmes d'apprentissage d'ordonnancement non parcimonieux. Plus précisément, nous considérons trois algorithmes de l'approche par paire, RankSVM-Dual [Joachims 2002], RankSVM-Primal [Chapelle 2010] et RankBoost [Freund 2003], ainsi que trois algorithmes de l'approche par liste, ListNet [Cao 2007], AdaRank-MAP et AdaRank-NDCG [Xu 2007], présentés au chapitre 2. Nous utilisons le test de Student unilatéral apparié pour évaluer la significativité des variations observées. Les ratios de parcimonie obtenus sont commentés.

7.2.2 Résultats

Les tableaux 7.1 et 7.2 comparent l'algorithme parcimonieux RankSVM-ℓ_1 aux algorithmes non parcimonieux de référence, selon la MAP et le NDCG@10 respectivement.

Nous observons que RankSVM-ℓ_1 présente majoritairement des résultats équivalents ou meilleurs que ceux obtenus avec les algorithmes de référence sur les jeux de données. Considérant la MAP, RankSVM-ℓ_1 fournit dans la majorité des cas des résultats comparables aux algorithmes non parcimonieux des approches par paire et par liste, sur l'ensemble des jeux de données. Nous constatons des résultats ponctuellement meilleurs pour sept couples jeu de données - algorithmes, avec des augmentations allant de 1% à 23%. Une diminution de la MAP, de l'ordre de 15%, est observée sur le jeu de données TD2004 comparativement à l'algorithme RankBoost. Cette dégradation reste néanmoins un phénomène marginal dû à la grande performance de l'algorithme RankBoost sur TD2004. En effet, comme indiqué au tableau 7.3, RankBoost surpasse l'ensemble des algorithmes de l'état de l'art sur ce

TABLE 7.1 – Comparaison des valeurs de MAP obtenues avec RankSVM-ℓ_1 avec les algorithmes non parcimonieux. Le symbole $\sim$ indique que les valeurs sont équivalentes.

Jeu de données	L1	RankSVM Dual	RankSVM Primal	RankBoost	AdaRank MAP	AdaRank NDCG	ListNet
Ohsumed	0.4491	$\sim$	+1% p=0.041	$\sim$	$\sim$	$\sim$	+1% p=0.016
MQ2008	0.4779	+2% p=0.03	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
MQ2007	0.4648	$\sim$	$\sim$	$\sim$	+2% p=0.015	+1% p=0.008	$\sim$
HP2004	0.6841	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
NP2004	0.6915	$\sim$	$\sim$	+23% p=0.002	$\sim$	$\sim$	$\sim$
TD2004	0.2219	$\sim$	+8% p=0.046	-15% p=0.027	$\sim$	$\sim$	$\sim$
HP2003	0.7638	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
NP2003	0.6775	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
TD2003	0.2679	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$

TABLE 7.2 – Comparaison des valeurs de NDCG@10 obtenues avec RankSVM-ℓ_1 avec les algorithmes non parcimonieux. Le symbole $\sim$ indique que les valeurs sont équivalentes.

Jeu de données	L1	RankSVM Dual	RankSVM Primal	RankBoost	AdaRank MAP	AdaRank NDCG	ListNet
Ohsumed	0.4545	$\sim$	$\sim$	+6% p=0.02	+3% p=0.03	$\sim$	$\sim$
MQ2008	0.2314	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
MQ2007	0.4412	-1% p=0.021	$\sim$	$\sim$	+2% p=0.04	+1% p=0.049	$\sim$
HP2004	0.7931	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
NP2004	0.8148	$\sim$	$\sim$	+18% p=0.0005	+9% p=0.04	+10% 0.05	$\sim$
TD2004	0.3059	$\sim$	$\sim$	-15% p=0.026	$\sim$	$\sim$	$\sim$
HP2003	0.8254	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	-1% p=0.016
NP2003	0.7839	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
TD2003	0.3538	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$

TABLE 7.3 – Pourcentage de diminution de la MAP comparativement à l'algorithme RankBoost sur TD2004. Les p-valeurs sont indiquées entre parenthèses.

RankSVM-Struct	RankSVM-Primal	AdaRank-MAP	AdaRank-NDCG	ListNet
-16% (0.017)	-21.2% (0.01)	-16.3% (0.007)	-25.9% (0.006)	-14.7% (0.01)

jeu de données bien précis.

Les résultats observés sur le NDCG@10 sont similaires et conduisent à la même interprétation. RankSVM-ℓ_1 fournit des valeurs de NDCG@10 comparables ou significativement meilleures à celles obtenues avec les algorithmes de référence dans la très grande majorité des cas.

Enfin, le tableau 7.4 montre que RankSVM-ℓ_1 obtient ces très bons résultats tout en utilisant en moyenne 60% de caractéristiques en moins comparativement aux approches non parcimonieuses. RankSVM-ℓ_1 constitue ainsi une bonne approche pour introduire de hauts niveaux de parcimonie dans les modèles tout en maintenant une très bonne qualité de prédiction.

7.3 Régularisations non-convexes : des méthodes efficaces pour la sélection de variables en apprentissage d'ordonnancement

Dans cette section, nous présentons les expérimentations mises en œuvre afin d'évaluer l'apport des régularisations parcimonieuses non-convexes et la performance de l'algorithme RankSVM-NC que nous proposons dans le cadre de la sélection de variables pour l'apprentissage d'ordonnancement. Dans un premier temps, nous détaillons notre protocole expérimental. Dans un second temps, nous discutons les résultats.

7.3.1 Protocole expérimental

Les expérimentations effectuées ont pour objectif de montrer l'apport de l'utilisation de regularisations non-convexes dans le cadre de l'apprentissage d'ordonnancement ainsi que la compétitivité de notre algorithme vis-à-vis des méthodes de sélection existantes.

Nous considérons trois régularisations non-convexes au sein de RankSVM-NC, à savoir les pénalités MCP, *log* et ℓ_q avec $q = 0.5$. Nous avons choisi de n'étudier qu'une seule valeur de q par souci de simplicité et de lisibilité. Explorer l'ensemble des valeurs possibles de q afin de sélectionner la configuration optimale est un sujet intéressant, mais n'est pas l'objet de cette étude. Nous nous intéressons ici au potentiel de l'utilisation de cette pénalité dans le contexte de la sélection de variables pour l'apprentissage d'ordonnancement, plutôt qu'à une analyse détaillée de son comportement suivant la valeur de q dans ce contexte.

Pour chacune de ces régularisations, nous apprenons le modèle qui maximise la MAP sur l'échantillon de validation et nous calculons la MAP et le NDCG@10 sur l'échantillon de test. Nous fixons $\varepsilon = 0.1$ pour la régularisation *log* et $\gamma = 2$ pour la pénalité MCP. Nous calculons également les taux de parcimonie. Nous comparons les valeurs de MAP, de NDCG@10 et les taux de parcimonie à ceux obtenus avec FenchelRank [Lai 2013a] et FSMRank [Lai 2013b], deux algorithmes de l'état de l'art, ainsi que RankSVM-ℓ_1, l'algorithme que nous avons précédemment proposé.

La significativité des variations des mesures de RI est évaluée avec le test de Student unilatéral apparié. Les résultats sont discutés dans la section suivante.

7.3.2 Résultats

Les résultats présentés ici sont également repris dans [Laporte 2014]. Dans un premier temps, nous présentons les ratios de parcimonies obtenus avec les différentes approches. Nous montrons la grande efficacité des approches non-convexes pour l'introduction de hauts niveaux de parcimonie dans les modèles. Dans un second temps, nous comparons les valeurs de MAP et de NDCG@10. Enfin, nous discutons les résultats et mettons en évidence la compétitivité des méthodes non-convexes, qui obtiennent des qualités de prédiction comparables aux algorithmes de référence, tout en sélectionnant en moyenne 50% de caractéristiques en moins.

7.3.2.1 Pénalités non-convexes : plus performantes pour induire de la parcimonie

Dans cette section, nous analysons les ratios de parcimonie optimaux indépendamment des valeurs des mesures de RI.

La figure 7.1 présente le *spider plot* de la proportion de caractéristiques supprimées, soit 1− le ratio de parcimonie, pour chacun des algorithmes sur chacun des jeux de données. Chaque axe de la toile représente un jeu de données, tandis que chaque ligne colorée représente un algorithme. Nous observons que les régularisations non convexes permettent de supprimer plus de caractéristiques que la pénalité ℓ_1 et les approches FenchelRank et FSMRank. Les régularisations *log* et $\ell_{0.5}$ introduisent le plus de parcimonie dans les modèles. Au contraire, l'approche FSMRank est celle qui supprime le moins de caractéristiques.

Ces observations sont également visibles au tableau 7.4, qui présente les ratios de parcimonie pour chaque algorithme et chaque jeu de données. Nous constatons que les régularisations *log* et $\ell_{0.5}$ conduisent aux plus faibles ratios de parcimonie, elles sont les plus efficaces pour supprimer un grand nombre de caractéristiques. En pratique, elles utilisent en moyenne deux fois moins de variables que les approches en norme ℓ_1 ou FSMRank pour l'apprentissage des modèles. La régularisation MCP apparaît comme la moins parcimonieuse des pénalités non convexes. Cette constatation n'est pas nécessairement surprenante, la pénalité MCP ayant été proposée pour corriger le biais induit par la norme ℓ_1 plus que pour introduire de plus hauts niveaux de parcimonie.

L'analyse des résultats de parcimonie pour chaque jeu de données pris indépendamment confirme nos observations. Les régularisations non convexes sont bien les plus parcimonieuses. Ainsi, la pénalité *log* présente les plus faibles taux de parcimonie pour quatre jeux de données sur neuf, à savoir Ohsumed, HP2004, NP2004 et NP2003. La régularisation $\ell_{0.5}$ est particulièrement efficace sur les quatre jeux de données MQ2008, MQ2007, HP2003 et TD2003, tandis que MCP est la plus efficace sur TD2004. Les écarts de parcimonie entre algorithmes sont variables suivant

TABLE 7.4 – Comparaison des ratios de parcimonie entre RankSVM-NC, RankSVM-ℓ_1 et l'état de l'art.

Jeu de données	**FenchelRank**	**FSMRank**	ℓ_1	**MCP**	***log***	$\ell_{0.5}$
Ohsumed	0.23	0.41	0.28	0.34	**0.12**	0.22
MQ2008	0.3	0.42	0.39	0.27	0.09	**0.07**
MQ2007	0.58	0.64	0.6	0.19	0.29	**0.05**
HP2004	0.19	0.26	0.17	0.24	**0.06**	0.12
NP2004	0.27	0.37	0.43	0.32	**0.13**	0.23
TD2004	0.46	0.67	0.42	**0.28**	0.36	0.3
HP2003	0.27	0.48	0.25	0.39	0.27	**0.16**
NP2003	**0.23**	0.44	0.48	0.36	**0.23**	0.27
TD2003	0.53	0.76	0.48	0.40	0.34	**0.20**
Moyenne	*0.34*	*0.49*	*0.39*	*0.31*	*0.21*	***0.18***

les jeux de données, ce qui peut s'expliquer par le fait que le nombre de variables pertinentes n'est pas nécessairement le même pour chacun d'eux.

Considérons la pénalité *log* pour les jeux de données sur lesquels elle détient le meilleur ratio de parcimonie. Nous observons que, comparativement à RankSVM-ℓ_1, FenchelRank et FSMRank, elle sélectionne :

- de 2 à 3.5 fois moins de caractéristiques sur Ohsumed,
- de 3 à 4 fois mois de caractéristiques sur HP2004 et
- de 2 à 3 fois moins de variables sur NP2004.

La pénalité $\ell_{0.5}$, comparativement à RankSVM-ℓ_1, FenchelRank et FSMRank, sélectionne :

- de 4 à 6 fois moins de variables sur MQ2008,
- de 11 à 13 fois moins de caractéristiques sur MQ2007,
- de 1.5 à 3 fois moins de variables sur HP2003 et
- de 2.5 à 3.5 environ de caractéristiques en moins sur TD2003.

L'algorithme de sélection de variables via les régularisations non convexes est donc particulièrement efficace pour supprimer un grand nombre de variables des modèles. Il surpasse très nettement les autres approches de l'état de l'art dans cette tâche.

7.3.2.2 Qualités de prédiction équivalentes à l'état de l'art

Dans cette section, nous confrontons les qualités de prédiction de notre algorithme utilisant les régularisations non convexes aux approches de l'état de l'art et à RankSVM-ℓ_1 que nous avons proposé. Les résultats obtenus sont présentés aux tableaux 7.5 et 7.6. Pour chaque jeu de données, la plus haute valeur de MAP et NDCG@10 obtenue est indiquée en gras. Tous les autres algorithmes sont alors comparés à cette valeur en utilisant le test de Student unilatéral apparié. Si aucune dégradation significative n'est observée, le symbole $\sim$ est indiqué ; sinon, le pourcentage de diminution ainsi que la p-valeur correspondante sont renseignés.

Nous observons que les résultats obtenus sont globalement équivalents pour l'en-

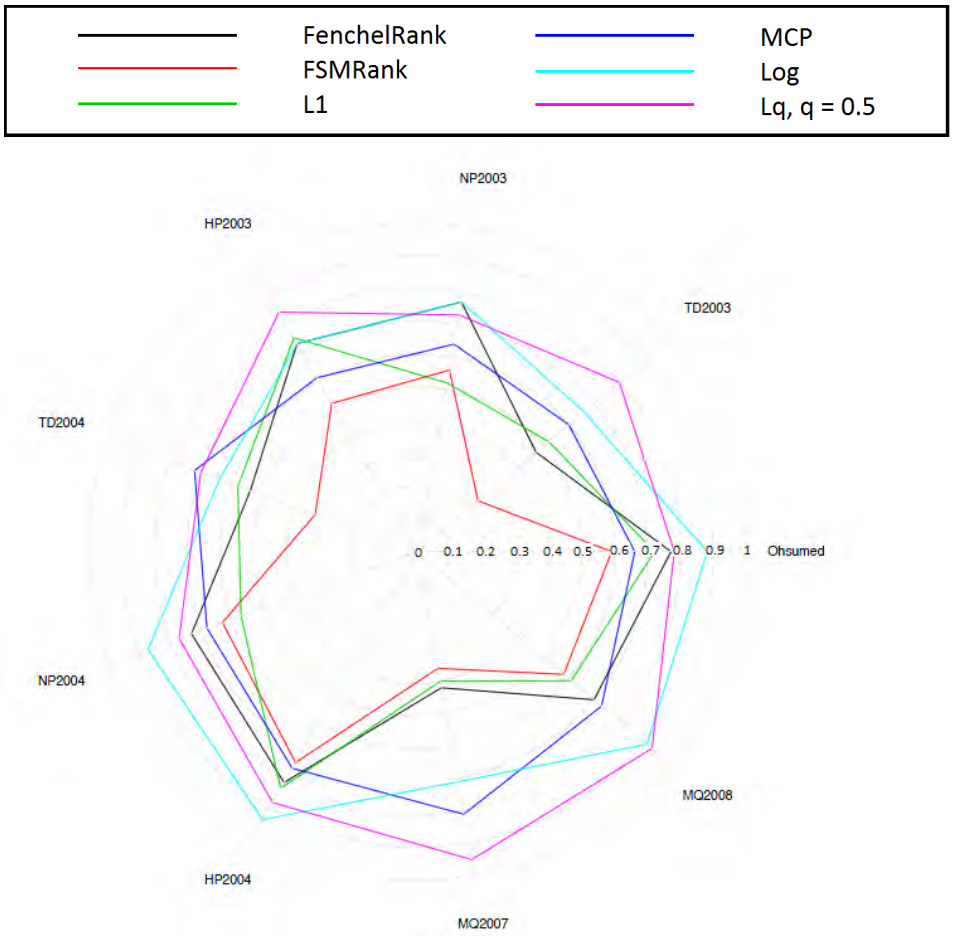


FIGURE 7.1 – Ratio de caractéristiques supprimées pour chaque algorithme sur chaque jeu de données.

semble des algorithmes et des jeux de données, quelle que soit la mesure considérée. Ponctuellement, des dégradations de la MAP ou du NDCG peuvent survenir. Nous les commentons de façon plus détaillée dans la suite de cette section.

Considérons tout d'abord les valeurs de MAP obtenues. Nous constatons que pour cinq jeux de données sur neuf, soient Ohsumed, MQ2008, NP2004, HP2003 et TD2003, les régularisations non convexes obtiennent d'aussi bonnes performances que l'algorithme RankSVM-ℓ_1 et les algorithmes de l'état de l'art tout en étant bien plus parcimonieuses. L'algorithme utilisant les pénalités non convexes fournit donc majoritairement des résultats équivalents à l'état de l'art. Par ailleurs, nous observons que sur trois des quatre jeux de données restants, la MAP n'est pas dégradée pour l'ensemble de régularisations non convexes, à l'exception de la pénalité MCP sur TD2004, la pénalité $\ell_{0.5}$ sur NP2003 et la pénalité *log* sur HP2004, les autres régularisations fournissant des résultats équivalents au meilleur algorithme. Certaines régularisations peuvent donc ne pas être adaptées pour certains jeux de données, mais de façon marginale. Enfin, sur MQ2007, le dernier jeu de données pour lequel une diminution de la MAP est observée, nous constatons que la dégradation affecte aussi bien l'algorithme utilisant les régularisations non convexes que FenchelRank. Par ailleurs, pour les régularisations MCP et *log*, la diminution de MAP effective est inférieure à celle constatée pour FenchelRank. L'utilisation des régularisations non convexes permet donc globalement d'obtenir des valeurs de MAP comparables à l'état de l'art tout en étant bien plus parcimonieuses, ce qui est un net avantage pour les algorithmes que nous proposons.

Considérons à présent les valeurs de NDCG@10. Nous constatons que les algorithmes présentent des résultats équivalents pour sept jeux de données sur neuf. RankSVM-ℓ_1 et la régularisation *log* fournissent les meilleurs résultats pour cinq jeux de données sur neuf, mais l'amélioration n'est pas statistiquement significative. Sur MQ2007 et HP2004, des dégradations sont observées, mais ne remettent pas en question la qualité de la méthode. En effet, sur HP2004, une diminution du NDCG@10 n'est constatée que pour la pénalité *log*, les régularisations MCP et $\ell_{0.5}$ fournissant des résultats similaires au meilleur algorithme, FenchelRank. Sur MQ2007, l'ensemble des algorithmes, convexes ou non convexes, présentent des résultats statistiquement moins bons que le meilleur algorithme FSMRank. Par ailleurs, la dégradation constatée pour les pénalité MCP et *log* est inférieure à celles de FenchelRank et RankSVM-ℓ_1.

Nous pouvons en conclure que l'algorithme utilisant les régularisations non convexes est aussi performant que les méthodes de l'état de l'art tout en étant bien plus parcimonieux.

7.3.2.3 Pénalités non-convexes : des méthodes compétitives pour la sélection de variables

Dans les deux sections précédentes, nous avons montré que les régularisations non convexes étaient plus efficaces pour introduire de la parcimonie dans les modèles et qu'elles obtenaient des résultats équivalents aux approches convexes, même si

TABLE 7.5 – Comparaison des valeurs de MAP entre la meilleure méthode et les autres algorithmes. Le symbole $\sim$ indique qu'il n'y a pas de différence significative par rapport à la meilleure méthode. En cas de variation significative, la p-valeur est donnée en italique.

Jeu de données	FenchelRank	FSMRank	ℓ_1	MCP	*log*	$\ell_{0.5}$
Ohsumed	$\sim$	$\sim$	$\sim$	**0,4513**	$\sim$	$\sim$
MQ2008	**0,4804**	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
MQ2007	-0.8% *0.02*	**0.4672**	$\sim$	-0.5% *0.03*	-0.7% *<0.001*	-3% *0.02*
HP2004	**0.7447**	$\sim$	$\sim$	$\sim$	-11% *0.01*	$\sim$
NP2004	**0.6946**	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
TD2004	$\sim$	**0.2327**	$\sim$	-4.7% *0.039*	$\sim$	$\sim$
HP2003	$\sim$	-4% *0.008*	**0.7638**	$\sim$	$\sim$	$\sim$
NP2003	$\sim$	**0.6823**	$\sim$	$\sim$	$\sim$	-4% *0.042*
TD2003	$\sim$	$\sim$	$\sim$	**0.2670**	$\sim$	$\sim$

TABLE 7.6 – Comparaison des valeurs de NDCG@10 entre la meilleure méthode et les autres algorithmes. Le symbole $\sim$ indique qu'il n'y a pas de différence significative par rapport à la meilleure méthode. En cas de variation significative, la p-valeur est donnée en italique.

Jeu de Données	FenchelRank	FSMRank	ℓ_1	MCP	*log*	$\ell_{0.5}$
Ohsumed	$\sim$	$\sim$	$\sim$	$\sim$	**0.4591**	$\sim$
MQ2008	$\sim$	**0.2323**	$\sim$	$\sim$	$\sim$	$\sim$
MQ2007	-1% *0.043*	**0.4445**	-0.7% *0.042*	-0.9% *0.013*	-0.5% *0.0045*	-5.6% *0.0016*
HP2004	**0.8274**	$\sim$	$\sim$	$\sim$	-4% *0.044*	$\sim$
NP2004	$\sim$	$\sim$	**0.8148**	$\sim$	$\sim$	$\sim$
TD2004	$\sim$	$\sim$	$\sim$	$\sim$	**0.3153**	$\sim$
HP2003	**0.8283**	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
NP2003	$\sim$	$\sim$	**0.7839**	$\sim$	$\sim$	$\sim$
TD2003	$\sim$	$\sim$	**0.3538**	$\sim$	$\sim$	$\sim$

des dégradations pouvaient être ponctuellement observées. Les analyses ont été effectuées de façon indépendante, nous les fusionnons à présent pour montrer la compétitivité de l'approche utilisant les régularisations non convexes.

Des dégradations de la MAP et du NDCG@10 ont été observées pour certains jeux de données. Sur MQ2007, les diminutions sont constatées pour la MAP et le NDCG@10 et l'ensemble des algorithmes comparativement à FSMRank. Notons que deux des régularisations non convexes présentent un pourcentage de diminution inférieur à celui de FenchelRank, algorithme de référence. Les dégradations constatées sur ce jeu de données sont donc plus dues à l'excellent comportement de FSMRank sur ce jeu de données très précis, plutôt qu'à un réel défaut de l'approche proposée.

Lorsque nous considérons les autres cas de dégradations, nous observons qu'elles apparaissent généralement pour le plus petit taux de parcimonie atteint sur l'échantillon. Ce point peut être interprété comme le signe que l'algorithme est allé "trop loin" dans la sélection des caractéristiques, ce qui conduit à une dégradation de la qualité de prédiction. Ce phénomène est à relativiser. En effet, le pourcentage de dégradation reste faible comparativement au gain observé sur le ratio de parcimonie. Ainsi, sur MQ2007, nous notons une dégradation de 3 % de la MAP et de 5.6 % du NDCG pour la régularisation $\ell_{0.5}$, tandis que le nombre de caractéristiques utilisées est de 11 à 13 fois inférieur à celui des méthodes de l'état de l'art. Il faut également noter que nous n'avons pas optimisé les paramètres ε et γ des régularisations $\ell_{0.5}$ et *log*. Les modèles ainsi appris ne sont pas nécessairement optimaux.

De façon générale, l'algorithme utilisant les régularisations non convexes est particulièrement compétitif comparé aux méthodes convexes et de l'état de l'art. Il permet en effet d'obtenir des qualités de prédiction comparables, tout en utilisant en moyenne deux fois moins de caractéristiques. Le graphe 7.2 illustre bien cet avantage. Les valeurs de MAP obtenues pour chaque algorithme sur trois jeux de données représentatifs sont représentées en fonction du ratio de parcimonie. Les lignes pointillées représentent les MAP moyennes de chaque échantillon. Nous constatons de façon nette que les régularisations non convexes, plus particulièrement *log* et $\ell_{0.5}$ présentent les plus petits ratios de parcimonie tout en conservant une qualité de prédiction similaire aux autres approches. Le gain en parcimonie est particulièrement marqué, tandis que les qualités de prédiction restent stables autour de la moyenne.

En conclusion, les méthodes utilisant des régularisations non convexes sont particulièrement efficaces pour la sélection de variables en apprentissage d'ordonnancement. Elles permettent de sélectionner en moyenne deux fois moins de caractéristiques que les méthodes convexes tout en conservant la même qualité de prédiction.

Conclusion et perspectives

Dans ce chapitre, nous avons proposé deux nouveaux algorithmes pour la sélection de variables en apprentissage d'ordonnancement. Le premier, nommé RankSVM-

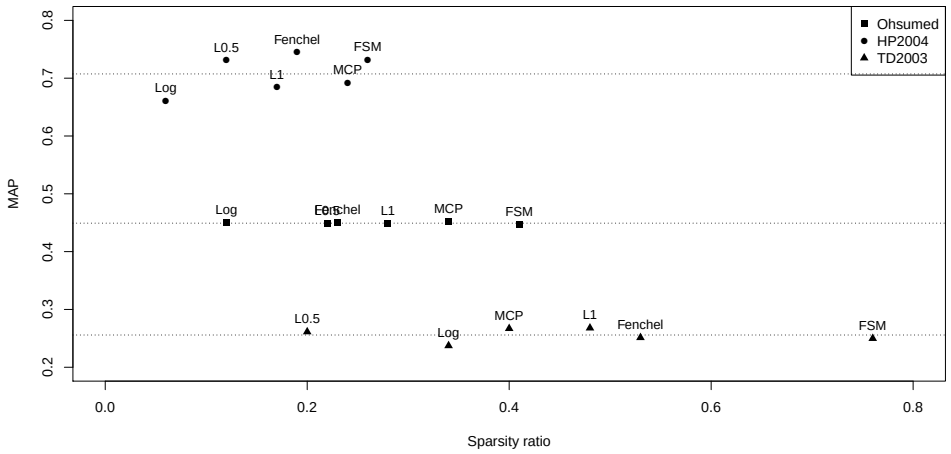


FIGURE 7.2 – Lien entre ratios de parcimonie et MAP pour les différents algorithmes sur trois jeux de données représentatifs. Les lignes pointillés représentent la valeur de MAP moyenne pour l'ensemble des algorithmes sur le jeu de données considéré. Les pénalités log et $\ell_{0.5}$ sont les meilleures pour les trois jeux de données.

ℓ_1 est basé sur l'utilisation de la norme ℓ_1 et d'une approche proximale pour effectuer la sélection de variables. Le second, nommé RankSVM-NC, utilise un schéma de repondération en norme ℓ_1 et le premier algorithme afin de considérer des régularisations non convexes.

Nous avons montré que l'utilisation des régularisations des normes non convexes étaient particulièrement efficaces pour la sélection de variables en apprentissage d'ordonnancement. Notre approche permet de sélectionner en moyenne deux fois moins de caractéristiques que les méthodes existantes, tout en conservant une qualité de prédiction semblable.

Dans le futur, nous prévoyons de concentrer nos travaux sur l'évaluation des temps de calcul des différentes méthodes, afin de déterminer quelle approche est la plus rapide et la plus efficace quand la parcimonie, la rapidité d'exécution et la qualité de prédiction sont considérées simultanément. Par ailleurs, nous prévoyons également d'analyser plus en détail l'impact des paramètres des régularisations sur la qualité des résultats.

Enfin, nous nous intéressons plus particulièrement à l'utilisation et l'adaptation de ces méthodes dans un contexte multitâche. Les approches multitâches permettent de prendre en compte les différentes tâches afin d'adapter l'apprentissage à chacune d'elles, directement dans l'algorithme. Cela nous parait une piste prometteuse pour l'adaptation de l'ordonnancement et pour la mise un place d'un moteur adaptatif.

Chapitre 8

Modèle d'évaluation implicite de la pertinence requête-document dans le cadre de moteurs à clics multiples

Sommaire

Dans les chapitres précédents, nous avons proposé un système d'ordonnancement basé sur la sélection de variables, permettant d'adapter l'ordonnancement au type de la requête. Nous avons proposé et évalué différents algorithmes de sélection de variables sur des jeux de données de référence. Néanmoins, nous n'avons pas étudié la performance de ces algorithmes ou l'apport du système adaptatif dans un cadre applicatif "réel", sur des données commerciales de grandes dimensions. Or, des travaux montrent que le comportement des algorithmes peut être différent lors du changement d'échelle [Liu 2011].

La création de jeux de données de référence de grande dimension, plus particulièrement l'évaluation de la pertinence des paires requête-document, est un processus long et coûteux. Il peut constituer un obstacle dans l'évaluation des travaux.

Dans ce chapitre, nous nous intéressons à la création automatique de jeux de données à partir des fichiers de connexion et des clics des utilisateurs. Ces jeux de données sont ensuite utilisés pour l'évaluation des algorithmes d'apprentissage d'ordonnancement que nous avons proposés et leur comparaison à l'état de l'art.

Dans un premier temps, nous justifions le besoin de méthodologie de création automatique de jeux de données d'évaluation. Nous présentons des méthodes et modèles proposés pour cette tâche dans la littérature.

Dans un deuxième temps, nous montrons que les approches existantes ne sont pas adaptées au cas particulier des moteurs de recherche à clics multiples, dont la majorité des moteurs de RI géoréférencées font partie. Nous présentons le modèle d'évaluation de la pertinence que nous avons proposé dans ce cadre.

Enfin, nous présentons le moteur Nomao et détaillons les caractéristiques du jeu de données créé.

8.1 Problématique et état de l'art

La création de jeu de données utilisables pour l'évaluation des approches est une composante centrale en RI. Dans cette section, nous rappelons brièvement le processus d'évaluation introduit au chapitre 1. Nous mettons en évidence certaines limites. Nous présentons de nouveaux besoins en matière de jeux de données et d'évaluation. Nous expliquons en quoi l'utilisation de fichiers de connexion et de modèles de clics est vue comme un moyen de répondre à ces problématiques et nous présentons un état de l'art des modèles existants.

8.1.1 De nouveaux besoins pour les jeux de données d'évaluation

L'évaluation constitue une composante centrale de la RI. Comme nous l'avons présenté au chapitre 1, la méthodologie Cranfield, associée à la technique du *pooling*, est la plus populaire dans ce contexte. L'annotation manuelle des jugements de pertinence par des experts humains garantit généralement la haute qualité des collections ainsi créées. Néanmoins, différents travaux ont mis en évidence certaines limites de cette approche [Borlund 2003][Harman 2010][Liu 2011].

Tout d'abord, construire des jeux de données annotés par des humains est un processus long et coûteux [Liu 2011]. Bien que la technique du *pooling* permette de réduire le nombre de documents considérés par les experts, le processus d'acquisition reste complexe à mettre en œuvre. D'une part, le nombre de requêtes et de documents doit être suffisamment grand pour garantir la représentativité des données et pour permettre l'utilisation de tests statistiques. D'autre part, la pertinence de chaque document vis-à-vis d'une requête doit être évaluée par différents annotateurs, afin de limiter le biais introduit par ceux-ci et de garantir des jugements de bonne qualité.

Ensuite, la technique du *pooling* ne garantit pas que tous les documents pertinents aient effectivement été restitués et annotés [Harman 2010]. Ces documents ne sont donc pas pris en compte dans l'évaluation de nouveaux modèles ou l'apprentissage de fonctions d'ordonnancement.

Enfin, les collections ainsi constituées peuvent manquer de réalisme [Borlund 2003]. En effet, le contexte de la recherche, et plus particulièrement l'utilisateur, sont généralement ignorés dans ce processus. Or, la pertinence d'un document vis-à-vis d'une requête est variable suivant l'utilisateur et son besoin d'information. Les annotateurs humains peuvent émettre des hypothèses sur le besoin utilisateur et essayer

d'appréhender le contexte de recherche, mais les jugements déduits ne sont pas nécessairement les mêmes que ceux d'utilisateurs réels.

L'utilisation de modèles de pertinence basés sur les informations contenues dans les fichiers de connexion de moteurs de recherche commerciaux, plus particulièrement des requêtes soumises par les utilisateurs et de leurs clics, permettent de contourner ces problèmes. Les moteurs de recherche commerciaux stockent chaque jour dans ces fichiers de connexion des millions d'actions effectuées par leurs utilisateurs : requêtes soumises, documents cliqués, temps passé sur les pages, etc. Les clics utilisateurs sont ainsi considérés comme des jugements implicites de la pertinence d'un document vis-à-vis d'une requête [Joachims 2002], bien que cette hypothèse soit parfois remise en cause, les clics étant considérés comme biaisés [Carterette 2007]. Différents travaux ont ainsi proposé des modèles basés sur les clics des utilisateurs permettant de prédire la pertinence d'un document et d'annoter automatiquement les couples requête-document. Nous les présentons dans la section suivante. Nous avons également publié un article dans la revue francophone *Document Numérique* relatif aux méthodes de l'état de l'art [Laporte 2013c].

8.1.2 Modèles de clics pour la création de jeux de données spécifiques

A notre connaissance, Joachims [Joachims 2002] a été le premier à proposer une stratégie d'extraction de préférences à partir des clics dans le cadre d'algorithmes. Joachims *et al.* [Joachims 2005] ont poursuivi ces travaux en testant différentes stratégies d'extraction de préférences dans le cadre d'algorithmes d'ordonnancement par paire. Parmi celles-ci, la plus connue, car la plus performante, est la stratégie *SkipAbove*. Elle stipule que, considérant un ensemble de résultats restitués par le système et l'ensemble des résultats cliqués correspondant, le document cliqué de rang le plus faible est préféré à l'ensemble des documents non cliqués de meilleur rang. Ainsi, si $(d_1, d_2, d_3, d_4, d_5)$ est la liste des documents restitués et si (d_1, d_4) est la liste des documents cliqués, nous obtenons les préférences suivantes : $d_4 \succ d_2$ et $d_4 \succ d_3$. Ces préférences sont directement utilisables par les algorithmes basés sur l'approche par paires. Radlinski *et al.* [Radlinski 2005] ont étendu ces stratégies dans le cadre de chaînes de requêtes, c'est-à-dire des successions de requêtes traduisant le même besoin mais tour à tour reformulées, généralisées ou spécialisées. La stratégie *SkipEarlierQuery* considère qu'un document cliqué est préféré à l'ensemble des documents non-cliqués à la requête précédente, dans la même chaîne de requêtes. Dans le cas où aucun document n'avait été cliqué pour la première requête, les auteurs considèrent qu'un document cliqué est préféré aux deux premiers documents de la requête précédente (stratégie *TopTwoEarlierQuery*).

Les approches basées sur les clics peuvent être biaisées par la présentation initiale des résultats. En effet, les utilisateurs sont généralement plus influencés par la position des résultats de recherche que par leur pertinence. Ainsi, ils ont tendance à cliquer sur les premiers résultats restitués par le système et à délaisser des do-

cuments situés plus bas dans la liste, même si ceux-ci sont les plus pertinents. Des approches ont été proposées afin de prendre en compte ce biais de position dans la modélisation des clics. La pertinence est alors inférée à partir de l'ensemble des sessions utilisateurs. Les premiers modèles proposés ont été le modèle de position et le modèle en cascade [Craswell 2008] .

Le modèle de position ("Position Model") considère que l'utilisateur clique sur un seul document qui est alors le document pertinent. La probabilité qu'un document soit cliqué dépend uniquement de sa position dans la liste de résultats et diminue avec celle-ci.

Le modèle en cascade ("Cascade Model") suppose que l'utilisateur regarde les résultats du haut vers le bas de la liste. Pour chaque résultat, il choisit de cliquer sur le document qui est alors le seul document pertinent ou de passer au résultat suivant. L'utilisateur clique ainsi sur un seul document qui lui paraît pertinent compte tenu des résultats précédents.

Il est important de noter que ces deux modèles postulent qu'il n'y a qu'un seul document pertinent, donc un seul clic, par session de recherche. Or, en pratique, les utilisateurs cliquent généralement sur plusieurs résultats au cours d'une même session. D'autres modèles ont ainsi été développés pour généraliser le modèle en cascade aux sessions pour lesquelles plusieurs résultats sont cliqués.

Le modèle de clics dépendants ("Dependent click model") considère que l'utilisateur consulte les résultats du haut vers le bas de la liste. Comme pour le modèle en cascade, l'utilisateur choisit à chaque étape de cliquer sur le document ou de passer au résultat suivant. Mais, contrairement au modèle en cascade où il s'arrête une fois le premier document cliqué, l'utilisateur peut revenir sur la liste de résultat et consulter d'autres documents. La probabilité qu'un clic ait lieu sur le document d_i à la position i est notée $c_{d_i,i}$ et est définie de la façon suivante [Guo 2009] :

$$c_{d_i,i} = r_{d_i} \prod_{j=1}^{i-1} (1 - r_{d_j} + \lambda_j r_{d_j})$$

où r_{d_i} est la pertinence du document d_i par rapport à la requête observée et λ est un paramètre qui traduit la tendance des utilisateurs à revenir sur la liste des résultats après avoir consulté un document. Ces deux quantités sont alors estimées de la façon suivante :

$$r_d = \frac{\#\text{clics sur le document sur cette position ou avant au cours des sessions}}{\#\text{nombre d'affichages du document avant la position considérée lors des sessions}}$$

$$\lambda_i = 1 - \frac{\#\text{sessions pour lesquelles le dernier document cliqué est à la position i}}{\#\text{sessions pour lesquelles la position i est cliquée}}$$

TABLE 8.1 – Modèle bayésien dynamique

$E_i = 1, A_i = 1 \leftrightarrow C_i = 1$	Un résultat est cliqué si et seulement si l'utilisateur a vu le résultat et a été attiré par celui-ci
$P(A_i = 1) = a_u$	La pertinence perçue est égale à la probabilité que l'utilisateur soit attiré par le document
$P(S_i = 1 \mid C_i = 1) = s_u$	La pertinence effective est la probabilité que l'utilisateur soit satisfait sachant qu'il a cliqué sur le document
$S_i = 1 \rightarrow E_{i+1} = 0$	Si un utilisateur est satisfait par un document alors il arrête sa recherche
$C_i = 0 \rightarrow S_i = 0$	Si le document n'a pas été cliqué alors l'utilisateur ne peut pas être satisfait
$P(E_{i+1} \mid E_i, S_i = 0) = \gamma$	L'utilisateur retourne sur la liste de résultats avec la probabilité γ s'il n'est pas satisfait par le document
$E_i = 0 \rightarrow E_{i+1} = 0$	L'utilisateur ne regarde le résultat suivant que s'il a traité le résultat en cours

Les coefficients r_d et λ_i sont relatifs à la requête q considérée, même si celle-ci n'est généralement pas indiquée pour ne pas alourdir la notation.

Le modèle bayésien dynamique ("Dynamic Bayesian Model") généralise également le modèle en cascade aux sessions à clics multiples et apporte une nouvelle contribution en définissant les notions de pertinence perçue et de pertinence effective [Chapelle 2009b]. La pertinence perçue a_u correspond à la probabilité que le document soit cliqué. Elle traduit le fait que l'utilisateur ait été attiré par le résultat avant consultation du contenu du document. La pertinence effective s_u traduit la satisfaction de l'utilisateur après consultation du contenu du document. Soient les évènements E_i : le résultat est vu dans la liste, C_i : le résultat est cliqué, A_i : l'utilisateur est attiré par le résultat et S_i : l'utilisateur est satisfait par le document, le modèle est alors défini [Chapelle 2009b] par les équations présentées à la table 8.1.

La pertinence globale du document est alors définie comme le produit de la pertinence perçue et de la pertinence effective $a_u s_u$. Elle est utilisée pour la construction des échantillons d'apprentissage. Liu *et al.* [Liu 2009a] ont également proposé un modèle bayésien, spécialement adapté à l'apprentissage de préférences dans les algorithmes d'ordonnancement de paires d'instances.

Ces modèles ont été proposés dans le cadre de moteurs de RI généralistes. En pratique, ils ne peuvent pas être utilisés sur l'ensemble des moteurs de recherche, comme nous l'expliquons dans la section suivante.

8.2 Proposition d'un modèle de clics pour la création automatique des jeux de données

Dans cette section, nous présentons les limites des modèles de clics existants, en particulier dans le cadre d'une utilisation sur des moteurs de recherche à clics multiples. Nous présentons une approche que nous avons proposée pour résoudre ces problèmes [Laporte 2012d] [Laporte 2013a].

8.2.1 Motivation

Les modèles présentés dans la section 8.1 ont été proposés dans le cadre de moteurs généralistes, pour lesquels les résultats de recherche sont présentés sous forme de liste d'URLs. Les approches considèrent que chaque résultat ne peut être cliqué qu'une seule fois au cours d'une même session de recherche. Cette hypothèse est très limitative. En effet, un même utilisateur peut cliquer sur un résultat, consulter la page associée, consulter d'autres pages pour finalement revenir sur l'URL précédemment consultée, qui est alors cliquée plusieurs fois. Par ailleurs, de plus en plus de moteurs proposent des visualisations variées des résultats, qui permettent des clics multiples sur un même résultat.

Considérons par exemple le cas des moteurs de RI géoréférencées. Les résultats sont généralement visualisés de façon usuelle, sous forme de liste, mais également localisés sur une carte géographique qui est elle-même cliquable. L'utilisateur peut accéder à la page web correspondante en cliquant soit sur la liste soit sur la carte. Plusieurs clics sont possibles. Par ailleurs, il est de plus en plus courant que les résultats soit présentés sous forme de liste de fiches, pour lesquelles différents éléments (titre du lieu, numéro de téléphone, ...) sont cliquables, comme par exemple sur le site fr.nomao.com (*cf.* figure 8.1). Cette caractéristique n'est pas spécifique aux moteurs de RI géoréférencées. Les moteurs de recherche bibliographique peuvent également proposer une visualisation par fiches. L'utilisateur peut alors cliquer sur l'icône du document ou sur son titre pour le consulter. Un lien de téléchargement peut également être proposé. Gallica, le moteur de recherche de la Bibliothèque Nationale de France (http ://gallica.bnf.fr) illustre bien ce type de fonctionnement (*cf.* figure 8.2).

Les méthodes existantes pour l'évaluation de la pertinence ne sont pas capables de prendre en compte les clics multiples sur un même document. Nous avons donc proposé un modèle spécialement conçu pour l'évaluation de la pertinence à partir des clics, dans le cadre de modèles à clics multiples. Il est présenté dans la section suivante. Ce modèle a par ailleurs été publié dans [Laporte 2013a].

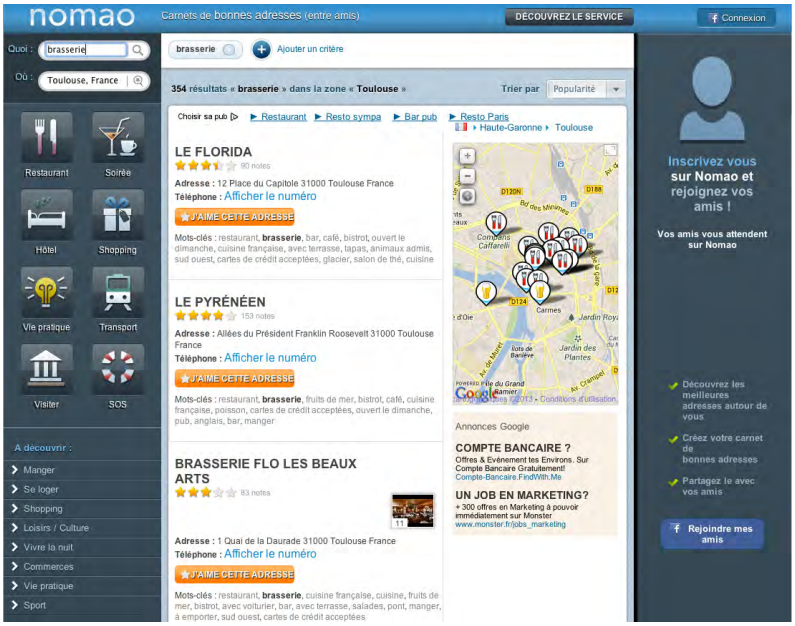


FIGURE 8.1 – Page de résultats du moteur de RI géoréférencé Nomao. Les résultats sont visualisés sous forme de fiches, dont le titre, le lien vers le numéro de téléphone, le bouton J'aime mais aussi le lien de réservation sont cliquables. Les résultats sont visualisés par des icônes cliquables sur une carte géographique (à droite).

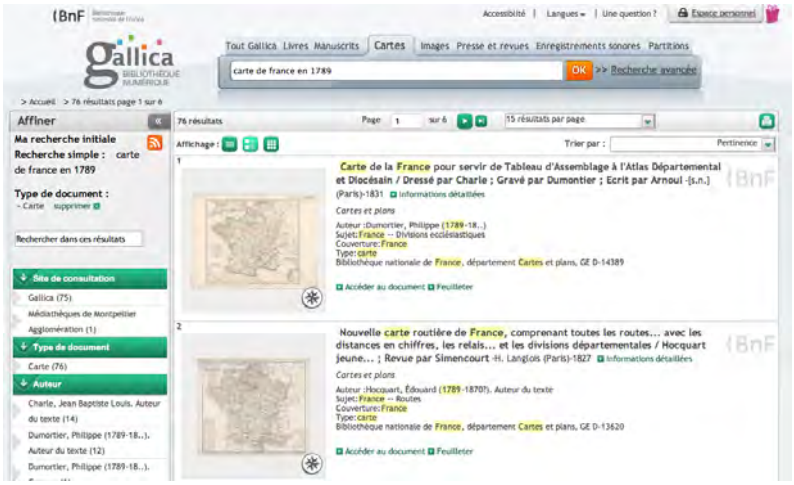


FIGURE 8.2 – Page de résultats du moteur de recherche bibliographique Gallica. Les résultats sont présentés sous forme de liste de fiches. Le titre du document, son icône et les liens permettant d'accéder et de feuilleter le document sont cliquables.

8.2.2 Présentation du modèle

Notre modèle est basé sur différentes hypothèses, décrites ci-après.

Hypothèse 1 *Un clic est un jugement implicite de pertinence.*
Cette hypothèse est partagée par l'ensemble des modèles de clics. Elle considère que si un utilisateur clique sur un document, c'est qu'il le perçoit comme pertinent a priori, même sans en avoir consulté le contenu. Elle nous paraît encore plus plausible dans le cadre de moteurs à clics multiples, car les utilisateurs ont accès sur la fiche à différents éléments détaillés du document et ont déjà une connaissance partielle relativement précise de son contenu.

Hypothèse 2 *Plus un document est cliqué par un utilisateur, plus il est pertinent pour celui-ci.*
Le nombre de clics traduit le fait que l'utilisateur consulte intensivement le document, ou qu'il revient sur celui-ci après avoir consulté d'autres résultats. Dans le premier cas, nous considérons que la recherche intensive traduit un intérêt accru de l'utilisateur, donc une pertinence a priori plus élevée. Dans le second cas, nous considérons qu'un utilisateur revenant sur un résultat après l'avoir délaissé le juge plus pertinent que les autres.

Hypothèse 3 *Différentes actions (ou clics) traduisent différents niveaux de pertinence.*
Chaque action, ou chaque clic, n'est pas équivalente en matière de satisfaction de l'utilisateur. Ainsi, sur un moteur de recherche bibliographique, un clic sur le titre indiquerait que le document semble intéressant pour l'utilisateur tandis qu'un clic sur le lien de téléchargement indiquerait que le document est suffisamment intéressant pour que l'utilisateur le télécharge. Le second type de clic traduit une pertinence plus importante que le premier. De façon similaire, sur un moteur de RI géoréférencées, un clic sur le numéro de téléphone d'un commerce indiquerait que celui-ci est suffisamment intéressant pour que l'utilisateur souhaite le contacter, comparativement à un commerce pour lequel il n'y a pas eu de clics sur le lien téléphonique.

D'après l'hypothèse 2, nous représentons la pertinence d'un couple document-requête pour un utilisateur comme la somme de tous les clics de cet utilisateur sur ce document. La troisième hypothèse nous conduit à pondérer chaque clic suivant le niveau de pertinence qu'il traduit.

Ces hypothèses nous conduisent donc à représenter la pertinence d'un document comme la somme pondérée des clics effectués sur ce document, telle que présentée à la proposition 2.

Proposition 2.

Considérons un utilisateur u, une requête q, un document d et un indice $i \in [1; N]$ représentant le type d'actions, parmi N actions possibles.
Soient α_i le poids de l'action i et $c_{i,d,q,u}$ le nombre de clics de type i sur le document d pour la requête q par l'utilisateur u.
Alors la pertinence $r_{d,q,u}$ du document d pour la requête q et l'utilisateur u est définie comme la somme pondérée des actions effectuées telle que

$$r_{d,q,u} = \sum_{i=1}^{N} \alpha_i c_{i,d,q,u}$$

Les poids α_i permettent de modéliser le fait que des actions peuvent avoir plus d'impact sur la pertinence que d'autres. Ainsi, des points différents indiqueront une importance variable des actions pour modéliser la pertinence, tandis que le cas équipondéré suggère que toutes les actions traduisent une satisfaction équivalente de la part de chaque utilisateur.

Plusieurs expérimentations ont été menées afin d'évaluer le modèle proposé. Nous les décrivons dans la section suivante et en présentons les résultats.

8.2.3 Evaluation du modèle proposé

Nous présentons dans cette section le jeu de données utilisé ainsi que les expérimentations menées pour évaluer le modèle proposé. Dans un premier temps, nous décrivons les caractéristiques du jeu de données. Dans un deuxième temps, nous montrons qu'utiliser l'ensemble des actions des utilisateurs constitue une bonne approche. Dans un troisième temps, nous montrons que différentes actions ont des impacts différents sur la modélisation de la pertinence.

8.2.3.1 Caractéristiques du jeu de données

Le modèle proposé a été évalué sur un jeu de données pilote issu du moteur de RI géoréférencées Nomao [1]. Celui-ci présente les résultats sous la forme d'une carte géographique cliquable et d'une liste de fiches comportant le nom du lieu, son numéro de téléphone, son adresse, des mots-clés, le lien vers son site internet, etc. Nous disposions d'un extrait des fichiers de connexion du moteur datant de 2011, correspondant à la version 2 de Nomao. Une troisième version est actuellement en ligne. Elle est présentée au sein du chapitre applicatif 9.

Différentes informations sont extraites des fichiers de connexion : l'identifiant et le contenu de la requête, les vingt premiers documents restitués ainsi que le type et le nombre de clics pour chaque association document-requête. Nous nous limitons

1. fr.nomao.com

aux requêtes pour lesquelles au moins deux documents ont été restitués et au moins un a été cliqué. Le jeu de données final contient 14 700 lignes, représentant 2014 requêtes distinctes pour lesquelles au moins un document a été cliqué, 1745 visiteurs uniques et 14 343 documents. Seuls les vingt premiers documents de chaque requête était disponibles. Cinq types de clics sont considérés pour modéliser la pertinence : sur le titre, sur la carte, sur le numéro de téléphone, sur le lien de réservation et sur le site web.

Le rang moyen des clics est égal à 3.54, ce qui semble indiquer que les utilisateurs cliquent en moyenne sur les troisième et quatrième documents. Néanmoins, la médiane des clics est égal à un, ce qui signifie qu'au moins la moitié des clics concerne le premier document. Ceci peut être expliqué soit par un biais de position (les utilisateurs cliquent sur le premier document car il est en haut de liste, même s'il n'est pas pertinent) soit par une performance inégale du moteur suivant les requêtes, celles-ci pouvant être plus ou moins faciles à interpréter par le système.

Nous observons également que certains clics sont plus fréquents que d'autres. Ainsi, les clics sur le titre, le numéro de téléphone et la carte représentent 96 % des actions (respectivement 44 %, 28 % et 24 %). Ceux sur le site web et le lien de réservation ne représentent que 3 % et 1% respectivement des actions effectuées par les utilisateurs. Certaines requêtes ne sont associées qu'à un seul type de clics.

Enfin, nous notons que le nombre de documents restitués varient grandement suivant les requêtes. Ainsi, le système restitue vingt documents pour 38 % des requêtes et moins de trois pour 26 % d'entre elles. Cette distribution des documents a un impact sur les valeurs de la mesure d'évaluation utilisée, en l'occurrence la MAP. En effet, les requêtes pour lesquelles seulement deux documents ont été restitués présentent des valeurs de précision moyenne égales soit à 1 soit à 0.5, ce qui conduit à des valeurs de MAP élevées comparativement aux jeux de données de l'état de l'art.

Le jeu de données a été découpé en 5 sous-échantillons de façon à effectuer une validation croisée de type 5-folds. Chaque échantillon a été utilisé comme ensemble de test, tandis que les quatre autres constituaient l'ensemble d'apprentissage. Nous avons conduit deux expérimentations. La première compare le score équipondéré à un score de pertinence aléatoire afin de montrer l'intérêt d'utiliser l'ensemble des clics. La deuxième expérimentation considère différents scores pour lesquels chaque type de clic reçoit le même poids, sauf un qui est fixé à zéro. L'objectif est de montrer l'impact de chaque clic sur la pertinence. Les scores sont utilisés pour annoter les jeux de données et des fonctions d'ordonnancement sont apprises en utilisant l'algorithme RankSVM [Joachims 2002]. La MAP obtenue est alors utilisée pour l'évaluation.

8.2.3.2 Evaluation du score équipondéré

Dans une première expérience, nous avons évalué la qualité du score équipondéré, *ie* $\alpha_i = 1 \ \forall i$, qui prend en compte l'ensemble des clics et leur affecte un poids équivalent. Nous l'avons comparé à un score de pertinence binaire généré

aléatoirement. Ces deux scores ont été utilisés pour annoter le jeu de données. Les fonctions d'ordonnancement ont alors été apprises. Nous avons calculé et comparé les valeurs de MAP. Nous nous sommes également intéressés au temps de calcul. Des plus grandes valeurs de MAP indiquent que le score correspondant permet de mieux classer les documents, tandis que de plus petits temps de calcul indique que le système apprend plus facilement le score de pertinence. Celui-ci est donc plus adapté pour la modélisation de la pertinence.

Quelle que soit la mesure utilisée, le modèle proposé fournit de meilleurs résultats que le score aléatoire. En effet, les temps d'exécution totaux de l'étape de validation croisée décroissent de 15000 secondes pour le score aléatoire à 10000 secondes pour le score équipondéré. De façon similaire, l'erreur de prédiction diminue de 82.7 % pour le score aléatoire à 38.6 % pour le score équipondéré. Enfin, nous observons que la MAP est plus élevée quand le score proposé est utilisé (0.78 contre 0.65 pour le score aléatoire). Utiliser les différents clics est donc efficace pour évaluer la pertinence.

8.2.3.3 Impact des différents types de clics sur la pertinence

Dans cette seconde expérimentation, nous nous sommes intéressés à l'importance de chaque type d'action sur la modélisation de la pertinence. Nous avons généré cinq scores. Pour chacun d'eux, quatre types de clics ont reçu la même pondération, tandis que le poids du cinquième type d'action a été fixé à zéro. Une action a ainsi été tour à tour annulée, afin d'observer son impact sur la qualité d'ordonnancement. Nous avons comparé les différents scores entre eux et au score équipondéré $\alpha_i = 1 \ \forall i$ en observant les valeurs de MAP obtenue. Une diminution de la MAP indique que le fait de ne pas prendre en compte l'action conduit à une diminution de la qualité de l'ordonnancement, donc que l'action en question est importante pour modéliser la pertinence. A l'inverse, une augmentation de la MAP indique que l'action introduit du bruit dans la modélisation et dégrade ainsi la qualité de l'ordonnancement.

Les résultats obtenus sont présentés à la table 8.2. Nous observons que quand $\alpha_{réservation} = 0$, c'est-à-dire quand les clics sur le lien de réservation ne sont pas pris en compte, la MAP diminue. Les clics sur le lien de réservation jouent donc un rôle important dans la modélisation de la pertinence. Ils semblent indiquer une plus grande satisfaction de l'utilisateur, ce qui correspond à l'intuition. L'utilisateur qui clique sur le lien de réservation veut a priori effectuer une réservation du lieu. Cela signifie que le lieu en question l'intéresse et qu'il est satisfait du résultat obtenu, puisqu'il souhaite effectuer une transaction.

A l'inverse, le résultat obtenu lorsque les clics sur le numéro de téléphone ne sont pas pris en compte est surprenant : la MAP n'est pas affectée par la suppression de cette action. Au contraire, elle est la plus élevée pour cette configuration. Intuitivement, nous supposions que les clics sur le numéro de téléphone auraient une importance similaire aux clics sur le lien de réservation. En effet, nous considérions

TABLE 8.2 – Valeurs de MAP sur les différents sous-échantillons de test

Test set	**Score**					
	$\alpha_i = 1 \forall i$	$\alpha_{réservation} = 0$	$\alpha_{téléphone} = 0$	$\alpha_{URL} = 0$	$\alpha_{carte} = 0$	$\alpha_{titre} = 0$
1	0.75	0.75	0.73	0.76	0.74	0.70
2	0.81	0.73	0.77	0.79	0.77	0.76
3	0.78	0.77	0.79	0.80	0.76	0.79
4	0.75	0.75	0.81	0.76	0.73	0.77
5	0.79	0.74	0.81	0.76	0 .75	0.79
Moyenne	0.78	**0.75**	**0.78**	0.77	**0.75**	0.76
Etendue	0.06	0.04	0.08	0.04	0.04	0.09

que si un utilisateur cherche à contacter un lieu, c'est qu'il est intéressé par celui-ci et que le système a répondu à son besoin. En pratique, les clics sur le téléphone semblent introduire du bruit. Par ailleurs, nous observons que l'étendue de la MAP obtenue est importante. Une hypothèse, que nous n'avons pas pu vérifier, est que la performance dépend du jeu de données.

Enfin, nous observons qu'en moyenne, la MAP diminue lorsque les clics sur la carte ne sont pas pris en compte. Ceux-ci sont donc importants pour la modélisation de la pertinence. Nous pouvons supposer que ces clics permettent de capter la satisfaction de l'utilisateur du point de vue de la localisation du lieu. Les clics sur la carte pourraient ainsi traduire la pertinence géographique du résultat, un élément important pour les moteurs d'informations géoréférencées.

Les expérimentations ont ainsi permis de montrer que l'utilisation de l'ensemble des clics est une approche efficace pour modéliser la pertinence dans le cadre de moteurs à clics multiples. Les scores obtenus peuvent ainsi être utilisés pour l'annotation des jeux de données.

Conclusion

Dans ce chapitre, nous nous sommes intéressés à l'évaluation de la pertinence à partir des actions des utilisateurs extraites de fichiers de connexion. Notre objectif est de proposer un modèle d'évaluation de la pertinence permettant d'annoter automatiquement un grand nombre de couples document-requête et de construire des jeux de données de référence de grande dimension.

Nous avons motivé l'utilisation de modèles d'évaluation de la pertinence basés sur l'utilisation des clics des utilisateurs extraits des fichiers de connexion. Nous avons présenté un état de l'art des méthodes existantes et montré que celles-ci ne sont pas applicables dans le cadre de moteurs à clics multiples, comme par exemple les moteurs de RI géoréférencées. Nous avons émis différentes hypothèses qui nous ont conduit à modéliser la pertinence comme la somme pondérée des actions des utilisateurs sur un document relatif à une requête donnée. Nous avons montré que cette approche avait du sens et que nous pouvions déduire un ordre d'importance entre les différentes actions disponibles sur le moteur de recherche considéré.

Ce modèle a été implémenté au sein du moteur commercial Nomao. Nous avons

ainsi développé au cours de la thèse, avec l'appui de l'équipe de l'entreprise, un outil d'extraction des actions utilisateur à partir des fichiers de connexion. Cet outil permet de générer différentes analyses statistiques à partir des clics des utilisateurs. Un module complémentaire a également été développé dans le cadre d'un stage que j'ai encadré pour le suivi des requêtes.

Cet outil est utilisé pour l'annotation de jeux de données, à des fins de recherche sur l'apprentissage d'ordonnancement au sein de l'entreprise. Des expériences sont en cours sur le super-calculateur Hypérion du groupement de calcul CALcul en MIdi-Pyrénées (CALMIP) [2]. Par ailleurs, un jeu de données a été généré afin d'évaluer les algorithmes de sélection proposés aux chapitres 6 et 7 ainsi que le système d'ordonnancement adaptatif présenté au chapitre 4. La préparation du jeu de données ainsi que les résultats des expériences préliminaires font l'objet de la troisième partie de ce manuscrit.

2. CALMIP est un groupement scientifique et le mésocentre de calcul de la région Midi-Pyrénées

Troisième partie

Un système d'ordonnancement basé sur la sélection de variables : application à un moteur commercial géoréférencé

Résumé

Dans cette partie, nous évaluons les algorithmes proposés dans un contexte d'exploitation industriel réel.

Dans le chapitre 9, nous utilisons le modèle d'évaluation proposé au chapitre 8 afin de construire un jeu de données d'évaluation. Celui-ci est issu des fichiers de connexion du moteur de RI géoréférencé Nomao, que nous présentons. Nous détaillons les types de clics utilisés par le modèle d'évaluation de la pertinence, la taille et la structure du jeu de données.

Dans le chapitre 10, nous évaluons les algorithmes que nous avons proposés aux chapitres 6 et 7 sur le jeu de données industriel ainsi obtenu. Nous montrons que les méthodes proposés sont plus performantes que l'état de l'art. En effet, elles permettent de supprimer jusqu'à 10 fois plus de caractéristiques, tout en conservant une bonne qualité d'ordonnancement. Les temps d'exécution restent généralement raisonnables (de quelques dizaines de secondes à quelques minutes) compte tenu du nombre de variables sélectionnées et de la taille du jeu de données.

Des travaux ultérieurs évalueront l'impact de la catégorisation des besoins sur la qualité de l'ordonnancement et l'apport du système adaptatif dépendant des requêtes.

Chapitre 9

Création d'un jeu de données spécifique au moteur Nomao

Sommaire

Dans ce chapitre, nous présentons le jeu de données utilisé pour évaluer les algorithmes dans un cadre d'application réel, le moteur de RI géoréférencées Nomao [1].

Dans une première section, nous décrivons le moteur de recherche à clics multiples Nomao, ainsi que les différentes actions qui ont été considérées pour l'évaluation de la pertinence. Dans une deuxième section, nous détaillons les propriétés du jeu de données créé à l'aide des informations relatives aux clics et du modèle de d'évaluation de la pertinence proposé au chapitre 8.

9.1 Description du moteur Nomao

Nomao est un moteur de RI géoréférencé français disponible sur mobile et sur le web. Nomao recommande des lieux aux utilisateurs, en prenant en compte les goûts de ceux-ci et de leur réseau social, la *e-reputation* du lieu et son adéquation avec la requête soumise.

Dans ces travaux, nous avons eu accès aux fichiers de connexion du moteur web, nous nous concentrons donc sur ce dernier. La structure des pages, la position et le nombre de liens ainsi que les graphismes ont subi plusieurs modifications au cours des trois dernières années. Le site actuellement disponible à l'url *fr.nomao.com* correspond à sa troisième version et est en ligne depuis mai 2012. Pour l'évaluation des algorithmes de sélection et du système d'ordonnancement adaptatif, nous nous sommes concentrés sur cette dernière version du moteur, dont une page de recherche est représentée à la figure 9.1.

1. http ://fr.nomao.com

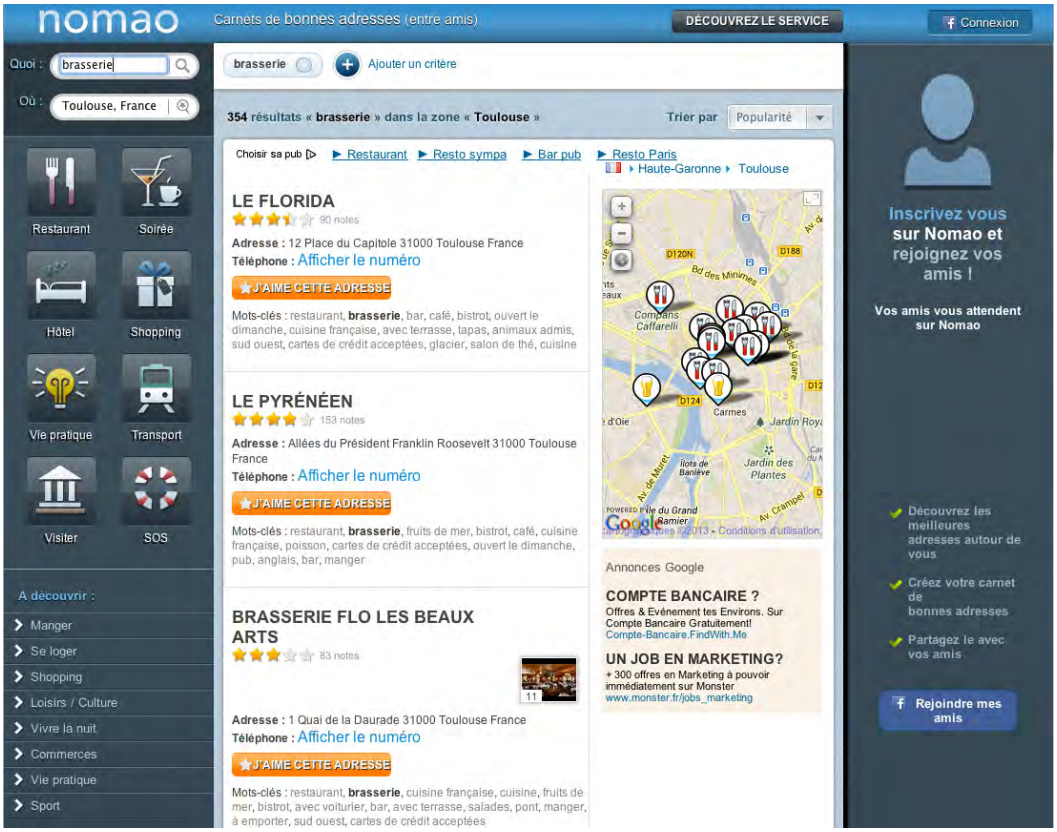


FIGURE 9.1 – Page de résultats du moteur Nomao. Les utilisateurs peuvent effectuer un clics sur le titre du lieu, sur le lien affichant le numéro de téléphone, sur le lien de réservation, sur le bouton "J'aime" et sur l'icône localisant le résultat sur la carte géographique.

Dans la suite de cette section, nous détaillons les informations que nous avons effectivement utilisées pour construire le jeu de données d'évaluation. Dans un premier temps, nous décrivons les différents types de recherche qui peuvent conduire à l'affichage d'une page de résultats sur Nomao et indiquons lesquels nous considérons. Dans un second temps, nous décrivons comment les résultats restitués sont présentés à l'utilisateur. Nous indiquons quelles actions ces derniers peuvent effectuer sur chaque résultat et nous précisons lesquelles actions sont retenues pour l'évaluation de la pertinence.

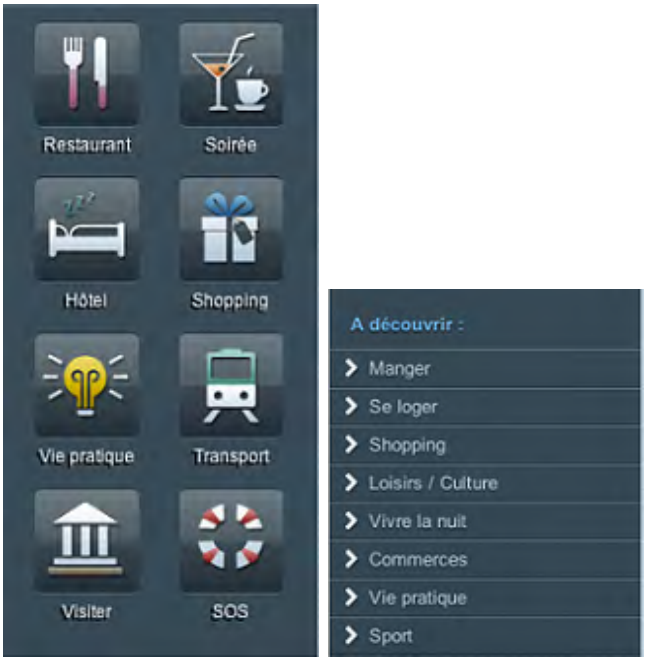
9.1.1 Différents types de recherche

Les fichiers de connexion stockent un grand nombre d'informations relatives aux recherches des utilisateurs, comme la requête soumise, les résultats consultés (cliqués) ou encore le type de clics. Le type de recherche (textuelle ou par icônes/liens) est également renseigné.

Tout d'abord, les utilisateurs peuvent accéder à une page de résultats ou à

la page d'un lieu en cliquant sur un lien depuis la page de résultats d'un autre moteur de recherche généraliste. Dans ce cas, la requête initiale a été soumise par les utilisateurs sur un autre moteur que Nomao. Pour notre étude, nous ne nous intéressons pas à ce type de recherches : nous nous concentrons sur les requêtes effectivement soumises par les utilisateurs à partir de l'interface de Nomao.

Sur l'interface Nomao, les utilisateurs peuvent rechercher l'information sans nécessairement taper leur requête. En pratique, deux types de recherches sont disponibles : textuelle en utilisant les barres de recherche et par navigation au travers des catégories prédéfinies par le moteur. Dans ce dernier cas, l'utilisateur va cliquer sur l'icône correspondant à la catégorie de son besoin (par exemple le couteau et la fourchette pour un restaurant, *cf.* figure 9.2 (a)) ou sur un des liens *à découvrir* situés sous les icônes (*cf.* figure 9.2 (b)). En pratique, nous nous sommes concentrés sur les recherches textuelles, c'est-à-dire pour lesquelles l'utilisateur écrit une requête.



(a) Recherche par icône (b) Recherche par liens

FIGURE 9.2 – Recherche par catégories sur le moteur Nomao. La recherche s'effectue en cliquant sur les icônes ou les liens correspondant aux catégories.

Pour effectuer une recherche textuelle sur Nomao, l'utilisateur utilise deux barres de recherche intitulée *Quoi* et *Où* (*cf.* figure 9.1, la barre de recherche est en haut à gauche). Le *Quoi* correspond au type de lieu recherché (bar, restaurant, commerces, administration) tandis que le *Où* spécifie la zone. Cette dernière est généralement pré-remplie, l'utilisateur étant géolocalisé dans une ville via son adresse IP.

9.1.2 Restitution des résultats et actions utilisateurs

Les résultats sont restitués sous deux formes : une carte géographique sur laquelle est localisé chaque résultat et une liste de fiches décrivant les lieux. Chaque lieu est caractérisé par différentes informations : son titre, son adresse, son numéro de téléphone, des mots-clés (*tag*), une description détaillée, sa géolocalisation, un score de *e-reputation*, des commentaires d'internautes et son nombre de favoris (nombre d'utilisateurs ayant déclaré aimer le lieu).

Au total, cinq actions sont réalisables pour chaque lieu (*cf.* figure 9.3), soit un clic sur :

- l'icône localisant le lieu sur la carte, qui redirige l'utilisateur vers la page de description détaillée du lieu ;
- le titre, qui redirige l'utilisateur vers la page de description détaillée du lieu ;
- le lien téléphonique, qui permet d'afficher un numéro surtaxé correspondant au lieu ;
- le lien de réservation, qui redirige l'utilisateur vers une centrale de réservation et
- le bouton "J'aime", qui permet à l'utilisateur de sauvegarder le lieu dans son carnet d'adresses et d'indiquer à son réseau social qu'il apprécie l'établissement en question.

FIGURE 9.3 – Détail des éléments cliquables sur la page de résultats, encadrés en rouge. Les utilisateurs peuvent cliquer sur le titre, le lien vers le numéro de téléphone, le bouton "J'aime", le lien de réservation mais aussi sur l'icône localisant le lieu sur la carte géographique.

Pour l'évaluation de la pertinence, nous nous sommes concentrés sur les quatre premiers types de clics présentés (sur la carte, le titre, le numéro de téléphone et le

lien de réservation) et avons négligé les clics sur le bouton "J'aime". Ce dernier point peut surprendre. En effet, le bouton "J'aime" indique que l'utilisateur apprécie le lieu et donc a priori que celui-ci est pertinent pour l'utilisateur. En pratique, les informations des fichiers de connexion stockent le fait que le bouton a été cliqué, mais ne font pas de distinction entre un utilisateur qui clique pour indiquer qu'il aime le lieu et un utilisateur qui clique pour indiquer qu'il n'aime plus le lieu (donc qui décoche le bouton). Nous avons donc considéré que l'information collectée n'est pas utilisable dans notre étude.

Les différents clics des utilisateurs ainsi que les requêtes soumises sont stockées chaque jour dans des fichiers de connexion. Nous avons développé avec Nomao un outil d'analyse des fichiers de logs, d'extraction et de stockage de statistiques relatives aux clics et aux requêtes. Il permet d'extraire les requêtes et de connaître les plus fréquentes, de connaître le nombre d'affichages d'une fiche de lieu et le nombre de clics correspondant pour la requête considérée et de calculer différentes statistiques comme la hauteur moyenne des clics. Un autre outil a été développé par Nomao pour calculer les différentes caractéristiques utilisées pour l'apprentissage. Ces outils et le modèle proposés ont été utilisés pour générer le jeu de données d'évaluation, dont nous décrivons les propriétés dans la section suivante.

9.2 Propriétés du jeu de données

Le jeu de données est composé de 9915 requêtes extraites aléatoirement des fichiers de connexion entre juillet 2012 et janvier 2013. La pertinence de chaque document vis-à-vis de la requête considérée a été déterminée à l'aide du modèle de clics présenté précédemment. Les clics sur la carte, le titre, le numéro de téléphone et le lien de réservation ont été utilisés. La pondération est la même pour chaque action. Les caractéristiques du jeu de données sont détaillées ci-après.

Le jeu de données final contient 124 195 paires requête-document dont nous extrayons 770 520 préférences. Il y a en moyenne 12.5 documents restitués par requête. Pour chaque requête, en moyenne 2.3 documents sont pertinents. Pour chaque paire, 101 caractéristiques sont calculées dont une majorité de similarités requête-document (certaines issues de la littérature et d'autres spécifiques au moteur Nomao). Les autres sont des caractéristiques du lieu, traduisant sa qualité (*e-reputation*, commentaires positifs/négatifs, favoris, ...), ou sont propres aux requêtes. Les variables sont normalisées au niveau de la requête, en utilisant la formule 2.2 présentée au chapitre 2.

Le fichier de données normalisé est partitionné à la manière des collections LETOR (*cf.* tableau 2.5, chapitre 2). Nous disposons ainsi de cinq répétitions ou *runs*, composé chacun d'un échantillon d'apprentissage, de validation et de test.

Le jeu de données ainsi structuré est utilisé pour l'évaluation des algorithmes de sélection de variables, dont les résultats sont présentés au chapitre 10.

Conclusion

La construction de ce jeu de données a nécessité l'utilisation de plusieurs outils développés à Nomao, incluant notamment l'outil d'analyse des fichiers de connexion et d'extractions de statistiques et un autre outil d'extraction des caractéristiques et de génération des jeux de données.

Chacun de ces deux modules a été développé en collaboration avec Nomao. Pour le premier outil, j'ai créé les spécifications, implémenté le *parser* (analyseur) des fichiers de connexion et le calcul des statistiques souhaitées. Les développeurs de Nomao se sont basés sur les spécifications fournies afin d'inclure de nouvelles informations à stocker dans les logs. Par ailleurs, le développement du *parser* a débuté avec leur aide, pour une intégration dans le système. Ce travail a été réalisé au cours de la première année de thèse.

Pour le second outil, j'ai écrit les spécifications, comprenant la liste des caractéristiques à coder, leur définition, la description du format des fichiers de sorties, de la normalisation et du partitionnement. L'outil a été implémenté par un développeur de Nomao avec lequel j'ai interagi pour apporter des compléments sur les spécifications ainsi que pour tester l'outil aux différentes étapes de son développement.

Dans le chapitre suivant, nous utilisons le jeu de données créé pour évaluer les algorithmes que nous avons proposés aux chapitres 6 et 7. En l'état actuel, le jeu de données ne peut pas encore être utilisé pour l'évaluation du système adaptatif, puisqu'il nécessite une catégorisation des requêtes. Des méthodes de catégorisation sont en cours d'analyse dans le cadre d'un stage de Master 1 que j'encadre. Nous prévoyons une première évaluation du système adaptatif dans le cadre industriel pour la fin d'année 2013.

Chapitre 10

Evaluation des algorithmes de sélection de variables sur le moteur commercial

Sommaire

Dans ce chapitre, nous appliquons les algorithmes de sélection de variables pour l'apprentissage d'ordonnancement que nous proposons sur le jeu de données issu du moteur de recherche Nomao. Nous les évaluons et comparons leur performance à l'algorithme de référence le plus récent au jour de la rédaction, FSMRank [Lai 2013b].

Dans un premier temps, nous présentons le protocole expérimental : algorithmes utilisés, valeurs des seuils et des paramètres et mesures d'évaluation considérés.

Dans un second temps, nous analysons les résultats obtenus. Nous comparons les algorithmes suivant les valeurs obtenues pour chaque mesure. Nous effectuons également une analyse plus globale où nous croisons les mesures deux à deux afin de faire apparaître les algorithmes qui présentent les meilleurs rapports MAP *vs.* ratio de parcimonie et temps d'exécution *vs.* ratio de parcimonie. Nous discutons brièvement des caractéristiques retenues par les modèles.

10.1 Protocole expérimental

L'objectif de cette étude est d'évaluer la performance des méthodes que nous proposons dans un cadre d'exploitation industriel. Le protocole expérimental est similaire à ceux des chapitres 6 et 7, mais l'évaluation est réalisée sur le jeu de données commercial issu du moteur Nomao présenté au chapitre précédent. Nous considérons au total 8 algorithmes, à savoir :

- les trois algorithmes de repondération en norme ℓ_2, Rankℓ_2-AROM, RankRWFS-ℓ_1 et RankRWFS-ℓ_0 ;
- l'algorithme proximal en norme ℓ_1 RankSVM-ℓ_1 ;

– l'algorithme proximal de repondération en norme ℓ_1 pour les régularisations non convexes, avec les trois régularisations MCP, *log* et ℓ_q, $q < 1$ et
– l'algorithme de référence FSMRank.

De façon similaire aux expérimentations du chapitre 6, pour les algorithmes de repondération en norme ℓ_2, nous évaluons cinq seuils correspondant à 90 %, 70 %, 50 %, 30 % et 10 % de caractéristiques restantes. Une caractéristique est retirée du modèle lorsque le poids associé est inférieur à 10^{-5}. Pour les régularisations non convexes, de façon similaire aux expérimentations du chapitre 7, nous nous limitons à l'analyse d'une seule valeur du paramètre q pour la norme ℓ_q, soit $q = 0.5$. Nous considérons les mêmes valeurs de paramètres pour les normes MCP et *log*, à savoir $\gamma = 2$ pour MCP et $\varepsilon = 0.1$ pour *log*.

Dans ces expérimentations, nous comparons les algorithmes selon leur capacité à :

1. Fournir des listes de résultats de haute qualité ;
2. Introduire de hauts niveaux de parcimonie dans les fonctions d'ordonnancement et
3. Effectuer la sélection de variables et l'apprentissage en un temps raisonnable.

Contrairement aux expérimentations des chapitres 6 et 7, nous comparons les qualités d'ordonnancement des algorithmes en utilisant les valeurs de MAP (*cf.* définition 8 chapitre 1). Nous ne considérons pas le NDCG, applicable dans le cas de jugements gradués de pertinence, non disponibles ici. Pour évaluer la capacité des algorithmes à supprimer un grand nombre de variables, nous considérons les ratios de parcimonie (*cf.* définition 17 chapitre 6). Enfin, nous nous intéressons au temps d'exécution nécessaire pour effectuer la sélection de variables et l'apprentissage du modèle. Les résultats obtenus sont présentés à la section suivante.

10.2 Résultats

Dans cette section nous présentons les résultats obtenus lors de l'évaluation des algorithmes sur le jeu de données Nomao.

Pour évaluer les algorithmes, nous nous concentrons sur leur capacité à apprendre des modèles ayant une bonne qualité d'ordonnancement, dans des temps raisonnables, tout en utilisant le plus faible nombre de caractéristiques possibles. Nous nous concentrons dans un premier temps sur le croisement des valeurs de MAP et des ratios de parcimonie, puis dans un second temps, sur le croisement des temps d'exécution et des ratios de parcimonie.

10.2.1 Excellent rapport MAP - ratio de parcimonie des méthodes proposées

La figure 10.1 présente le lien entre MAP et ratio de parcimonie sur le jeu de données considéré pour l'ensemble des huit algorithmes. La ligne pointillée repré-

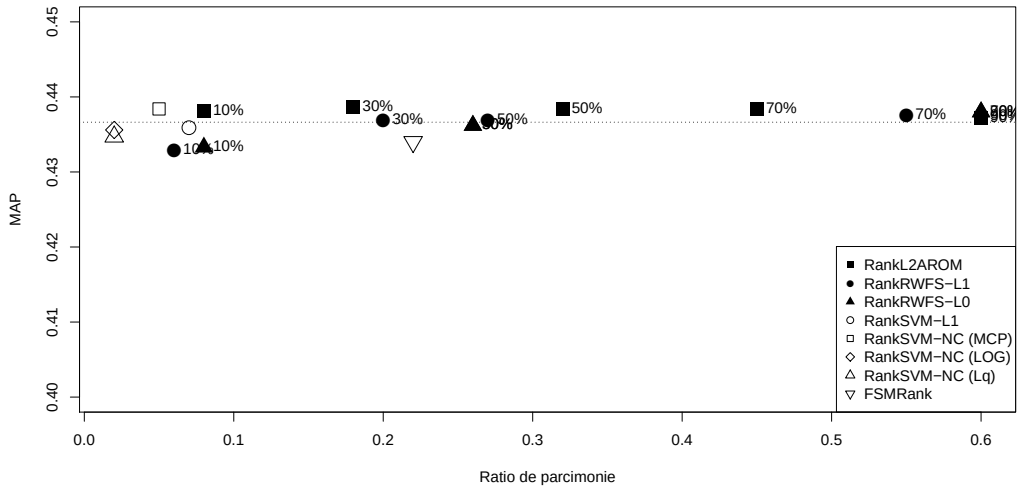


FIGURE 10.1 – Lien entre MAP et ratio de parcimonie pour l'ensemble des algorithmes. La ligne pointillée représente la moyenne des MAP obtenues pour tous les algorithmes. Les meilleures méthodes sont en haut à gauche du graphe.

sente la valeur moyenne de MAP obtenue par les différentes méthodes. Les méthodes les plus performantes sont celles qui combinent des MAP élevées (comparativement aux autres méthodes) et des ratios de parcimonie faibles. Elles se trouvent dans le coin supérieur gauche du graphique. L'analyse de ce graphe conduit à plusieurs observations.

Tout d'abord, nous observons que les valeurs de MAP restent stables pour l'ensemble des algorithmes. Les écarts restent faibles, de l'ordre du millième. Nous pouvons conclure que les différentes méthodes sont globalement équivalentes sur ce jeu de données lorsque la qualité d'ordonnancement est considérée.

Ensuite, nous constatons au contraire une grande variabilité des ratios de parcimonie effectivement atteints, allant de 0.6 (60% de caractéristiques conservées) à 0.02 (2% de caractéristiques conservées). En pratique, nous distinguons trois groupes d'algorithmes. Le premier, le plus à droite du graphique, est composé des algorithmes les moins parcimonieux. De façon logique, il s'agit des algorithmes de repondération pour lesquels nous avions fixé des seuils de sélection élevés, à 70% et 90% de caractéristiques restantes. Il est intéressant de noter que les ratios de parcimonies atteints sont grandement inférieurs à ceux demandés (entre 40% et 60% de caractéristiques restantes). Une proportion importante de variables n'est donc pas nécessaire pour les modèles. Le deuxième groupe est constitué des algorithmes de repondération en norme ℓ_2 avec des seuils théoriques de 30% et 50% et de l'algorithme de référence FSMRank. Les ratios de parcimonie effectifs varient entre 0.2 et 0.3. Nous observons par ailleurs que les algorithmes que nous proposons sont

meilleurs que l'état de l'art, dès que le seuil théorique demandé est inférieur ou égal à 30%. Le troisième groupe est constitué des méthodes les plus parcimonieuses, à savoir les algorithmes de repondérations en norme ℓ_2 avec un seuil théorique à 10%, l'algorithme proximal en norme ℓ_1 et les approches non convexes. Ce sont toutes des méthodes que nous avons proposées. Nos approches sont ainsi particulièrement efficaces pour introduire de la parcimonie dans les modèles. Dans ce domaine, elles surpassent l'état de l'art.

Enfin, lorsque nous procédons à une analyse croisée des valeurs de MAP et des ratios de parcimonie (*cf.* figure 10.1), nous constatons que nos approches sont les plus compétitives. Les algorithmes de repondération en norme ℓ_2 avec un seuil à 10%, l'algorithme proximal RankSVM-ℓ_1 et les approches non convexes obtiennent une qualité d'ordonnancement au moins équivalente à l'état de l'art, tout en supprimant de 3 à 10 fois plus de caractéristiques. Les régularisations non convexes *log* et ℓ_q obtiennent les plus petits ratios de parcimonie (de l'ordre de 2% de caractéristiques restantes) tout en conservant une MAP équivalente à l'état de l'art. La régularisation MCP est légèrement moins parcimonieuse (5% de caractéristiques restantes) mais obtient la valeur de MAP la plus élevée de tous les algorithmes.

Les approches que nous proposons sont donc particulièrement efficaces pour apprendre des fonctions d'ordonnancement de bonne qualité, tout en supprimant un grand nombre de caractéristiques des modèles.

10.2.2 Des rapports temps d'exécution - ratio de parcimonie raisonnables

La figure 10.2 présente le croisement entre temps d'exécution et ratio de parcimonie de chaque algorithme. La ligne pointillée présente le temps moyen nécessaire pour sélectionner les caractéristiques et apprendre le modèle. Idéalement, un algorithme doit pouvoir sélectionner un nombre important de caractéristiques dans un temps raisonnable. Les expérimentations ont été réalisées sur un serveur composé de 24 processeurs Intel Core cadencés à 3 GHz chacun et 96 Go de RAM partagés par l'ensemble des processeurs. Les algorithmes ont été exécutés en parallèle. Nous considérons que les temps d'exécution sont raisonnables s'ils vont de quelques secondes à quelques minutes, vue la taille du jeu de données.

Nous constatons une grande variabilité des temps d'exécution des algorithmes. De façon globale, le temps d'exécution augmente logiquement avec le nombre de caractéristiques supprimées.

Les algorithmes utilisant des régularisations non convexes sont les plus parcimonieux, mais également les moins rapides. Nous notons également que l'algorithme proximal RankSVM-ℓ_1 est également moins rapide que les autres approches. Cela peut expliquer la lenteur relative des approches non convexes. En effet, leur vitesse est liée à celle de RankSVM-ℓ_1 à travers le schéma de repondération. Il serait intéressant de regarder si des optimisations ne sont pas possibles au sein du code de l'algorithme proximal, afin d'améliorer les temps de calcul.

Nous constatons que l'algorithme utilisant la régularisation non convexe *log*

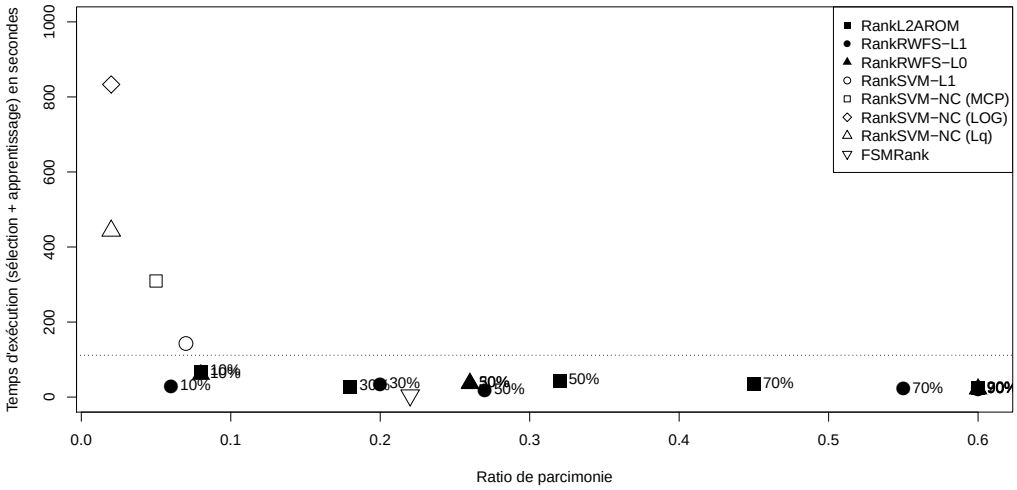


FIGURE 10.2 – Lien entre temps d'exécution et ratio de parcimonie pour l'ensemble des algorithmes. La ligne pointillée représente la moyenne des temps d'exécution obtenus pour tous les algorithmes.

est environ deux fois plus lent que celui utilisant la régularisation ℓ_q, pour des ratios de parcimonie similaires. La régularisation MCP est plus rapide, mais moins parcimonieuse. En pratique, les régularisations *log* et ℓ_q suppriment 2.5 fois plus de variables que MCP, prenant respectivement 2.7 et 1.4 fois plus de temps que cette dernière.

Afin de mieux observer le comportement des autres algorithmes, plus rapides, nous continuons l'analyse en supprimant les algorithmes non convexes et proximaux du graphique (*cf.* figure 10.3). Nous observons que, pour les algorithmes de repondération de la norme ℓ_2, les temps d'exécution ont tendance à augmenter avec le nombre de variables à supprimer. Les algorithmes les plus parcimonieux sont également les moins rapides, avec des temps d'exécution allant de 30 secondes à 1 minute en moyenne. L'algorithme le plus rapide est FSMRank, qui apprend les fonctions en moins de 5 secondes en moyenne. A ratio de parcimonie équivalent, les algorithmes de repondération sont environ 5 à 6 fois plus lents en moyenne. Quand nous considérons les plus petits seuils de parcimonie, nous observons que les algorithmes suppriment de 3 à 4 fois plus de caractéristiques que FSMRank, mais en prenant de 6 à 12 fois plus de temps. FSMRank, l'algorithme de référence, semble donc être plus performant pour apprendre des modèles parcimonieux de façon rapide.

Néanmoins, ce dernier point est à nuancer. Les temps d'exécution des algorithmes de repondération en norme ℓ_2 restent raisonnables vis-à-vis des ratio de parcimonie atteints. Par ailleurs, nous n'avons pas pris en compte le calcul de la

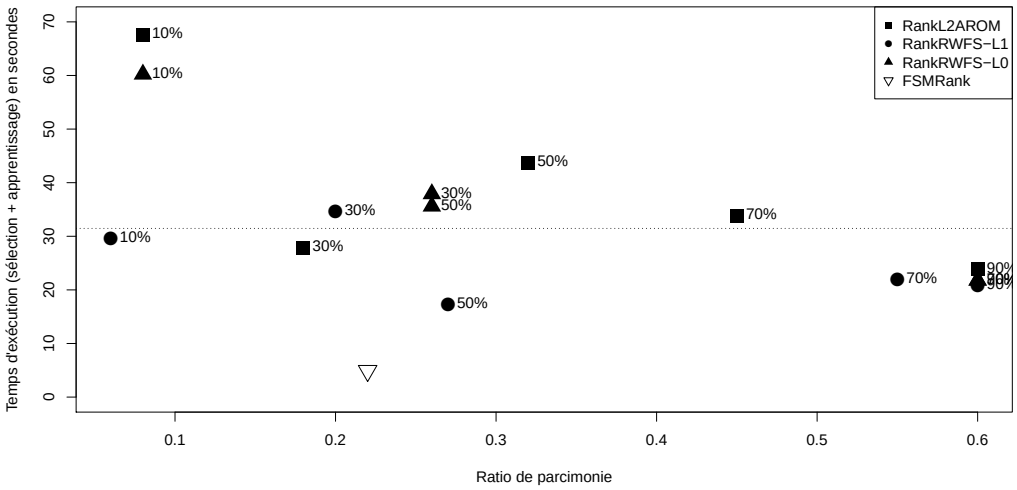


FIGURE 10.3 – Lien entre temps d'exécution et ratio de parcimonie pour les algorithmes les plus rapides. La ligne pointillée représente la moyenne des temps d'exécution obtenus pour tous les algorithmes.

matrice des corrélations donnée en entrée de l'algorithme FSMRank. Dans le cas de ce jeu de données, le calcul de cette matrice prend en moyenne une quinzaine de secondes. La prise en compte de ce temps permet de réduire l'écart entre FSMRank et les autres algorithmes. Pour les plus petits seuils de parcimonie, les algorithmes de repondération en norme ℓ_2 supprime 3 à 4 fois plus de caractéristiques que FSMRank, en prenant de 1.5 à 3.5 fois plus de temps. Le rapport temps d'exécution - ratio de parcimonie est ainsi plus favorable pour les approches que nous proposons.

Conclusion

Cette évaluation a permis de vérifier l'efficacité des méthodes proposées dans un cadre d'exploitation industrielle. Les algorithmes que nous proposons sont plus performant que l'état de l'art pour introduire de la parcimonie dans les modèles tout en conservant de bonnes qualités d'ordonnancement. Les modèles sont appris dans des temps raisonnables compte tenu des ratios de parcimonie atteint. Les approches de repondération en norme ℓ_2 sont les plus rapides parmi les méthodes proposées. Elles présentent un rapport temps d'exécution - ratio de parcimonie satisfaisant vis-à-vis de l'algorithme de l'état de l'art. Les approches non convexes sont moins efficaces lorsque le temps d'exécution est considéré. Leur amélioration constitue une piste de travail à court terme.

Les résultats de cette étude ont été utilisés au sein de l'entreprise Nomao afin de sélectionner un algorithme de sélection de variables pour l'apprentissage d'ordon-

nancement à intégrer au système existant. L'algorithme RankRWFS a été choisi. Il est en cours d'intégration dans le système et devrait être opérationnel d'ici décembre 2013. Une analyse des variables sélectionnées a également été effectuée. Elle a permis de déterminer de nouvelles caractéristiques à implémenter qui seront utilisées dans les fonctions d'ordonnancement du moteur de recherche.

Les algorithmes que nous avons proposés au cours de cette thèse se révèlent performants à la fois sur des données universitaires et issues du monde industriel. Les résultats obtenus entre les deux expérimentations concordent : les méthodes que nous proposons sont capables d'apprendre des modèles très parcimonieux sans dégrader la qualité d'ordonnancement, sur des données académiques et commerciales. Par ailleurs, nous vérifions sur le jeu de données issu du moteur Nomao que les algorithmes en norme ℓ_2 présentent des temps de calcul raisonnables, de quelques dizaines de secondes à quelques minutes. Les travaux réalisés sont ainsi intégrés dans un cadre d'exploitation industriel, au sein d'un moteur de recherche commercial.

Chapitre 11

Conclusion et perspectives

Sommaire

Nous avons présenté dans ce manuscrit les travaux que nous avons menés au cours de ces trois années de thèse. Nos contributions se situent dans le cadre de l'apprentissage d'ordonnancement adapté au contexte. Dans ce dernier chapitre, nous commençons par dresser une synthèse de nos contributions, incluant un rappel des méthodes proposées et des résultats obtenus, avant de discuter de nos perspectives de recherche.

11.1 Conclusion

Nos travaux se situent dans le cadre de la RI et plus particulièrement de l'apprentissage d'ordonnancement adapté au contexte. Nous avons ainsi proposé un nouveau système d'ordonnancement adaptatif, basé sur la sélection de variables. Cela nous a conduit à proposer d'une part, de nouveaux algorithmes de sélection de variables pour l'apprentissage d'ordonnancement et d'autre part, un nouveau modèle d'évaluation de la pertinence dédié aux moteurs de recherche à clics multiples pour la création de jeux de données d'apprentissage. Nous avons également débuté une évaluation de nos contributions dans un cadre d'exploitation industrielle réel.

Dans les sections suivantes, nous résumons les différentes contributions ainsi que les résultats obtenus.

11.1.1 Système adaptatif basé sur la sélection de variables

Dans une première contribution, nous avons proposé un système d'ordonnancement adaptatif basé sur la sélection de variables. Les méthodes actuelles d'apprentissage d'ordonnancement ne font pas de distinction entre les différentes requêtes ou les différents besoins des utilisateurs. En pratique, les mêmes fonctions sont apprises puis utilisées pour l'ensemble des requêtes.

De nombreux travaux [Geng 2008, Banerjee 2009, Liu 2011, Laporte 2013c] ont montré qu'adapter l'ordonnancement aux types de requêtes améliorait la qualité de la liste de résultats restituée par les systèmes. Ces travaux proposent d'apprendre des fonctions d'ordonnancement spécifiques à différents groupes de requêtes.

Dans le chapitre 4, nous avons ainsi formalisé le système adaptatif mis en place dans les approches précédentes. Nous avons proposé un nouveau système qui est basé sur la catégorisation de requêtes et la sélection de variables, dont l'utilisation présente à notre sens plusieurs avantages.

Tout d'abord, elle permet de supprimer des caractéristiques bruitées ou non pertinentes pour le besoin d'information. Or, la présence de celles-ci peut potentiellement conduire à une dégradation de la qualité d'apprentissage et de prédiction des fonctions. La sélection de variables est ainsi vue comme un moyen de contrôler la qualité des fonctions d'ordonnancement.

Ensuite, en réduisant le nombre de caractéristiques utilisées par les algorithmes, elle permet de réduire les temps de calcul, ce qui est un enjeu important en apprentissage d'ordonnancement. En effet, le nombre de caractéristiques utilisables par les algorithmes est très élevé (potentiellement plusieurs dizaines de milliers). Les algorithmes ne sont pas nécessairement conçus pour supporter cette dimensionnalité, qui peut conduire à des temps de calcul prohibitifs. Bien que d'autres approches soient envisageables pour réduire ces temps de calcul, par exemple la parallélisation des algorithmes, l'utilisation de la sélection de variables apparaît comme une piste intéressante à explorer pour beaucoup de travaux [Geng 2007, Lai 2013a, Lai 2013b].

Enfin, combiner la sélection de variables à l'apprentissage de fonctions pour chaque groupe de requêtes permet d'apprendre des fonctions réellement spécifiques au besoin de l'utilisateur. En effet, certaines caractéristiques peuvent être pertinentes pour certains types de besoins mais pas pour d'autres. En pratique, il est naturel de supposer que certaines caractéristiques sont pertinentes pour l'ensemble des besoins tandis que d'autres n'ont de sens que dans certaines situations. Effectuer une sélection des caractéristiques sur chaque groupe de requêtes permet non seulement d'apprendre des poids différents selon le groupe pour une caractéristique commune à l'ensemble des groupes, mais aussi de sélectionner des caractéristiques différentes et adaptées à chaque catégorie de besoins.

Le système proposé est actuellement en cours d'évaluation dans un contexte d'exploitation réel, sur des données issues du moteur de recherche d'informations géoréférencées Nomao. Sa mise en place a nécessité la proposition et le développement de méthodes de sélection de variables ainsi que d'évaluation de la pertinence pour la création de jeux de données d'apprentissage. Les contributions réalisées dans ce cadre sont rappelées dans les section suivantes.

11.1.2 SVM parcimonieux pour la sélection de variables en apprentissage d'ordonnancement

La proposition d'un système d'ordonnancement adaptatif basé sur la sélection de variables nous a conduit à développer des approches de sélection de variables pour l'apprentissage d'ordonnancement. Dans ce contexte, nous nous sommes particulièrement intéressés à l'utilisation de SVM parcimonieux utilisant des normes convexes et non convexes. Nos travaux ont porté sur la proposition de méthodes d'approximation des problèmes parcimonieux en norme ℓ_1 et ℓ_0 par repondération de problèmes en norme ℓ_2 d'une part, d'un algorithme FBS en norme ℓ_1 et d'un algorithme de résolution de problèmes utilisant des régularisations parcimonieuses non convexes d'autre part.

11.1.2.1 Approches de repondération en norme ℓ_2

Dans le chapitre 6, nous nous sommes intéressés à l'utilisation de méthodes de repondération en norme ℓ_2 avec seuillage des poids des variables pour approcher la solution des SVM parcimonieux en norme ℓ_1 et ℓ_0. Nous avons proposé et évalué trois algorithmes dans ce cadre.

L'algorithme Rankℓ_2-AROM constitue une adaptation de la méthode ℓ_2-AROM proposée par Weston [Weston 2003]. Il est utilisé pour approcher la solution des SVM parcimonieux en norme ℓ_0. L'algorithme fonctionne par itérations successives. A chaque itération t, chaque caractéristique est repondérée par un vecteur $\mathbf{v}_i^{(t)}$ qui correspond au produit des poids $\mathbf{w}_i$ appris au cours des $t-1$ itérations précédentes. Un modèle est alors appris par l'algorithme en norme ℓ_2. Les caractéristiques dont le poids est inférieur à un certain seuil sont retirées du modèle. Le processus est itéré jusqu'à ce que le nombre de caractéristiques restantes soit inférieur ou égal au seuil maximal fixé par l'utilisateur.

Les algorithmes RankRWFS-ℓ_1 et RankRWFS-ℓ_2 constituent deux variantes d'un même schéma, nommé RankRWFS, que nous proposons. Ils approchent respectivement les problèmes des SVM régularisés en norme ℓ_1 et en norme ℓ_0. Si le fonctionnement des deux algorithmes est identique dans sa structure, la règle de repondération est spécifique à la régularisation considérée. A chaque itération t, chaque observation est pondérée par un vecteur $\mathbf{v}_i^{(t)}$, calculé selon la règle de mise à jour appropriée à la régularisation considérée, $\mathbf{v}_i(t) = \sqrt{|\mathbf{w}_i^{(t-1)}\mathbf{v}_i^{(t-1)}|}$ pour la

norme ℓ_1 et $\mathbf{v}_i(t) = |\mathbf{w}_i^{(t-1)}\mathbf{v}_i^{(t-1)}|$ pour la norme ℓ_0. Un modèle est alors appris par l'algorithme en norme ℓ_2 et les caractéristiques dont le poids est inférieur à un certain seuil sont retirées du modèle. Le processus est itéré jusqu'à ce que le nombre de caractéristiques restantes soit inférieur ou égal au nombre maximal requis.

Ces trois algorithmes ont été évalué sur cinq jeux de données issus de LETOR (Ohsumed, HP2004, NP2004 et TD2004). Les expérimentations ont montré que nos algorithmes étaient beaucoup plus performants pour introduire de la parcimonie que les approches existantes, tout en conservant une qualité de prédiction équivalente aux approches de l'état de l'art. Par ailleurs, nous avons observé que les algorithmes effectuaient la sélection dans des temps raisonnables. De façon générale, il est apparu que l'approche RankRWFS est globalement plus performante, notamment l'algorithme RankRWFS-ℓ_0 qui présente le meilleur comportement (parcimonie, qualité de prédiction).

11.1.2.2 Algorithme proximal en norme ℓ_1 et schéma de repondération pour l'utilisation de régularisations non convexes

Les algorithmes précédents présentent un inconvénient. La norme ℓ_2 n'est pas parcimonieuse et le schéma de repondération n'introduit donc pas naturellement la parcimonie. Il est nécessaire de seuiller les variables pour les annuler. Dans le chapitre 7, nous nous sommes focalisés sur une méthode de résolution du problème en norme ℓ_1 ne nécessitant pas de seuillage et d'un schéma de repondération de cette dernière pour la prise en compte de régularisations non convexes.

Nous avons d'abord proposé un algorithme FBS pour résoudre le problème de l'apprentissage parcimonieux de préférences avec des SVM en norme ℓ_2. Cet algorithme, nommé RankSVM-ℓ_1, surpasse les algorithmes non parcimonieux de l'état de l'art et présente une performance similaire aux algorithmes de sélection de variables en apprentissage d'ordonnancement les plus récents.

Nous avons également proposé un schéma de repondération de RankSVM-ℓ_1, nommé RankSVM-NC, qui permet de prendre en compte des régularisations non convexes. Ces dernières présentent les avantages d'être soit plus parcimonieuses (*log*, ℓ_q, $q < 1$) soit moins biaisées (MCP) que la norme ℓ_1. De plus, leur utilisation dans le cadre de la sélection de variables en apprentissage d'ordonnancement n'a, à notre connaissance, jamais été étudiée. Nous avons montré que ce schéma de repondération permettait de supprimer en moyenne quatre fois plus de caractéristiques que les approches existantes, tout en conservant la même qualité de prédiction. En ce sens, l'approche proposée surpasse nettement les méthodes existantes.

Nous avons ainsi proposé plusieurs algorithmes de sélection de variables pour l'apprentissage d'ordonnancement qui s'avèrent compétitifs vis-à-vis des approches existantes. Nous les avons évalués sur des jeux de données de références issus des collection LETOR. Nous avons également débuté une évaluation sur un jeu de données commercial. Ces expérimentations donnent d'excellents résultats et montrent la performance des méthodes proposées dans un cadre d'exploitation réel. Les algorithmes

constituent également un élément central de l'évaluation du système adaptatif basé sur la sélection de variables, qui est en cours de réalisation.

11.1.3 Modèle d'évaluation de la pertinence à partir des clics des utilisateurs pour les modèles à clics multiples

L'évaluation du système adaptatif et des algorithmes que nous proposons dans un contexte d'exploitation réel nécessite la création de jeu de données d'évaluation. Or, évaluer manuellement la pertinence de paires requête-document, en prenant en compte le contexte, est une étape longue et coûteuse. Dans le chapitre 8, nous avons proposé un modèle d'évaluation de la pertinence basé sur les clics des utilisateurs pour les moteurs à clics multiples, qui constituent notre cadre d'application industrielle.

De nombreux modèles d'évaluation de la pertinence basés sur les clics et actions des utilisateurs existent dans la littérature, comme nous l'avons présenté au chapitre 8. Les clics sont vus comme des jugements implicites de pertinence, avec l'hypothèse sous-jacente qu'un utilisateur qui clique sur un résultat le perçoit a priori comme pertinent pour sa requête. Nous avons montré en quoi les modèles existants ne sont pas adaptés pour l'utilisation dans le cas de moteurs à clics multiples et nous avons proposé un modèle prenant en compte cette spécificité.

Pour proposer un nouveau modèle, nous avons émis trois hypothèses distinctes. Nous avons d'abord supposé que tout clic sur un résultat, que ce soit sur son titre ou un autre de ses élements, constitue un jugement implicite de pertinence. Nous avons ensuite supposé que plus un utilisateur clique sur un résultat, plus ce dernier est pertinent. Enfin, nous avons considéré que toutes les actions ne traduisaient pas le même degré de pertinence. Par exemple, un clic sur le lien de téléchargement d'un document traduit un plus grand intérêt de l'utilisateur qu'un clic sur le titre. Ces trois hypothèses nous ont conduits à représenter la pertinence comme la somme pondérée du nombre de clics sur un documents.

Les expérimentations, réalisées à partir de données extraites des fichiers de connexion de Nomao, ont montré que ce modèle constitue une bonne approche pour l'évaluation de la pertinence sur les moteurs à clics multiples et la construction de jeux de données de référence. Un outil d'analyse des fichiers de connexion et de génération a été développé dans ce cadre. Le modèle proposé est par ailleurs utilisé actuellement pour générer des jeux de données d'apprentissage pour l'évaluation des approches d'apprentissage d'ordonnancement.

11.1.4 Évaluation des contributions dans un cadre d'exploitation réel

Les algorithmes proposés ont été évalués dans un premier temps sur des jeux de référence issus des collections LETOR. Nous souhaitions également évaluer leur comportement dans un cadre d'exploitation réel, en l'occurrence, sur le moteur d'information géoréférencés Nomao. De façon similaire, nous souhaitons également

évaluer le système d'ordonnancement adaptatif, pour lequel nous n'avons réalisé que des expériences préliminaires [Laporte 2013c].

Dans le chapitre 9, nous avons présenté le jeu de données utilisé pour l'évaluation des algorithmes de sélection de variables que nous avons proposés. Les paires requête-document extraites ont été annotées grâce au modèle d'évaluation de la pertinence que nous avions proposé. Nous avons également spécifié les caractéristiques que nous souhaitions utiliser à l'équipe technique de Nomao, qui s'est chargée de leur calcul.

Dans le chapitre 10, nous avons évalué les algorithmes que nous avons proposés sur le jeu de données ainsi construit. Nous avons comparé les résultats (MAP, temps d'exécution, taux de parcimonie) à ceux obtenus avec l'algorithme FSMRank. Nous avons montré que nos algorithmes étaient plus performants que l'état de l'art dans un cadre réel d'exploitation, soit du point de vue de la parcimonie (RankSVM-ℓ_1, RankSVM-NC) soit du point de vue de la parcimonie et des temps de calcul (Rankℓ_2-AROM, RankRWFS).

L'analyse du système adaptatif intégrant la sélection de variables devrait être réalisée d'ici à la fin de l'année 2013..

11.2 Perspectives

Les travaux réalisés au cours de cette thèse ont donné de bons résultats. Des analyses sont actuellement en cours et constituent des pistes de recherche à cours termes. Les résultats obtenus ont ouvert la voie à d'autres projets de recherche, à court et moyen termes. Nous détaillons nos perspectives de recherche dans les sections suivantes.

11.2.1 Évaluation complète du système adaptatif proposé dans un cadre d'exploitation réel

La première perspective de travail à court terme est l'évaluation complète du système adaptatif dans le cadre de Nomao. L'objectif est de déterminer si l'adaptation de l'ordonnancement aux groupes de requêtes par l'utilisation de la classification et de la sélection de variables permet d'améliorer de façon significative la qualité des liste de résultats. L'amélioration obtenue devra également être quantifiée.

Le jeu de données utilisé sera celui présenté au chapitre 9. Plusieurs méthodes pourront être utilisées pour créer des groupes de requêtes ou de besoins : classification supervisée, non supervisée, ou encore utilisation combinée de typologies et de ressources sémantiques spécifiques (hiérarchies de mots clés, thésauri). La proposition de nouvelles approches de catégorisation de requêtes constitue également une piste de recherche à moyen terme (*cf.* section 11.2.4).

Une fois les groupes de requêtes définis, nous procèderons à l'évaluation du système adaptatif. Des fonctions d'ordonnancement spécifiques seront apprises pour chaque groupe et évaluées sur le jeu de données. Les résultats de l'approche adaptative en matière de MAP seront comparés à ceux obtenus par la méthode non

adaptative.

Le système adaptatif pourra par ailleurs être évalué sur d'autres jeux de données pour lesquels une classification des requêtes est envisageable, afin de vérifier que la méthode donne des résultats similaires pour différents cadres d'application. Par ailleurs, nous envisageons également d'adapter notre approche afin d'apprendre des fonctions d'ordonnancement propres, non pas à des catégories de besoins, mais à des catégories d'usages (par exemple usage web *vs* mobile).

11.2.2 Apprentissage multitâche parcimonieux en apprentissage d'ordonnancement

Dans ces travaux, nous avons proposé d'apprendre des fonctions d'ordonnancement spécifiques à des groupes de besoins ou de requêtes à l'aide d'algorithmes de sélection de variables pour l'apprentissage d'ordonnancement. En pratique, cela revient à créer x groupes de requêtes, à scinder le jeu de données en x sous-ensembles correspondants et à apprendre une fonction pour chaque groupe. Les x fonctions d'ordonnancement sont ainsi apprises indépendamment les unes des autres. Comme nous l'avons constaté dans certains travaux préliminaires [Laporte 2013c], certaines caractéristiques sont pertinentes pour différentes tâches et devraient être sélectionnées de façon commune pour tous les groupes.

L'apprentissage multitâche parcimonieux apprend des fonctions spécifiques à chaque tâche tout en sélectionnant les caractéristiques pertinentes de façon jointe à l'ensemble des tâches. Autrement dit, la sélection est effectuée en essayant au maximum de conserver les mêmes caractéristiques pertinentes sur l'ensemble des tâches. Cette approche nous paraît prometteuse dans le contexte de l'adaptation des fonctions d'ordonnancement. Peu de travaux se sont intéressés à l'utilisation de méthodes d'apprentissage d'ordonnancement multitâche non parcimonieuses et à notre connaissance, l'utilisation d'approches d'apprentissage multitâches parcimonieuses n'a pas encore été étudiée.

Nous notons par ailleurs que l'algorithme RankSVM-ℓ_1, que nous avons proposé, peut être adapté pour effectuer de la sélection multitâche. Nous envisageons d'évaluer l'apport de l'apprentissage multitâche parcimonieux pour l'aprentissage d'ordonnancement en RI.

11.2.3 Méthodologies d'évaluation de la pertinence à partir de jugements implicites

Dans cette thèse, nous avons proposé une première approche pour l'évaluation de la pertinence à partir des clics des utilisateurs sur les moteurs de recherche à clics multiples. Nous avons pu mettre en évidence que certaines actions traduisent un niveau de pertinence perçue plus élevé que d'autres. Le modèle proposé permet de prendre en compte ce fait en pondérant différemment les différents types d'actions. Néanmoins il ne permet pas d'apprendre automatiquement les poids à affecter à chacune de ces différentes actions. L'ordre d'importance relative entre actions ne

peut pas être déterminé automatiquement.

Nous prévoyons de développer et proposer de nouveaux modèles d'évaluation de la pertinence basés sur les clics des utilisateurs. Ils devront prendre en compte l'existence des différents niveaux d'importance des types de clics pour l'évaluation de la pertinence. Ils devront également être suffisamment génériques pour être rapidement adaptables aux différents types de moteurs rencontrés.

Par ailleurs, un autre problème requiert une attention particulière. Un document pertinent qui n'est jamais restitué par le système, ou restitué sur une page de résultats trop éloignée, ne sera probablement jamais cliqué. Il ne sera donc jamais considéré comme pertinent alors qu'il l'est effectivement. La qualité d'ordonnancement du système initial peut donc potentiellement introduire un biais dans les modèles d'évaluation de la pertinence. Il est donc nécessaire de proposer des méthodes qui puissent évaluer la pertinence de l'ensemble des documents, en limitant le biais dû au système d'ordonnancement d'origine. Nous réfléchirons donc à la prise en compte de ce biais dans les modèles que nous proposerons.

11.2.4 Classification automatique de requêtes suivant le besoin de l'utilisateur et le contexte

Comme nous l'avons indiqué dans notre manuscrit au chapitre 4, de nombreux travaux se sont intéressés à la mise en place de typologies de besoins ou à la détection de requêtes similaires pour l'adaptation. Dans nos travaux, nous proposons d'intégrer au système d'ordonnancement un module de catégorisation des requêtes pour l'adaptation. Nous nous intéressons donc naturellement au développement de méthodologie de classification des requêtes.

Nous pouvons envisager d'utiliser et/ou de définir des typologies de besoins ou de requêtes [Broder 2002, Jansen 2008], donc d'utiliser des a priori pour effectuer la classification. Une thèse est en cours sur ce sujet dans notre équipe, en partenariat avec un laboratoire de linguistique. La définition de nouvelles typologies implique d'effectuer certaines analyses statistiques sur des ensembles de requêtes issues de moteurs de recherche. Différentes méthodes de statistiques descriptives peuvent être envisagées dans ce but. L'affectation d'une requête à la catégorie pré-établie, suivant la typologie, peut alors s'effectuer à l'aide de méthodes de classification supervisée, comme par exemple les SVM multiclasses [Mendoza 2009] ou les arbres de décision [Kang 2003]. Elle peut également être réalisée à l'aide de ressources sémantiques, comme une hiérarchie de mots-clés ou des thésauri.

Nous pouvons également envisager d'utiliser des méthodes de classification non supervisée, comme la classification hiérarchique ascendante ou l'algorithme des k moyennes, ou en proposer de nouvelles. Dans ce cas, il est nécessaire de réfléchir au développement de mesures de similarités permettant d'affecter à une classe ainsi déterminée chaque nouvelle requête soumise au système.

Différentes solutions sont ainsi envisageables et pourront être explorées. Leur intégration au sein du système d'ordonnancement adaptatif permettra d'évaluer la performance de celui-ci. Par ailleurs, l'utilisation de différentes approches de

catégorisations des requêtes au sein du système permettra de quantifier l'impact de ce module sur la qualité de l'ordonnancement obtenu.

Bibliographie

[Adomavicius 2011] Gediminas Adomavicius et Alexander Tuzhilin. *Context-Aware Recommender Systems.* In Recommender Systems Handbook, pages 217–253. 2011. (Cité en page 6.)

[Amaldi 1993] Edoardo Amaldi et Viggo Kann. *The Complexity and Approximability of Finding Maximum Feasible Subsystems of Linear Relations.* Theoretical Computer Science, vol. 147, pages 181–210, 1993. (Cité en page 91.)

[Amini 2008] Massih Reza Amini, Tuong Vinh Truong et Cyril Goutte. *A boosting algorithm for learning bipartite ranking functions with partially labeled data.* In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '08, pages 99–106, New York, NY, USA, 2008. ACM. (Cité en page 35.)

[Amini 2013] Massih-Reza Amini et Eric Gaussier. Recherche d'information. applications, modèles et algorithmes. Eyrolles, 2013. (Cité en pages 5 et 16.)

[Bach 2011] Francis Bach, Rodolphe Jenatton, Julien Mairal et Guillaume Obozinski. *Convex optimization with sparsity-inducing norms.* Optimization Mach. Learning, 2011. (Cité en page 133.)

[Baeza-Yates 1999] Ricardo A. Baeza-Yates et Berthier Ribeiro-Neto. Modern information retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. (Cité en page 16.)

[Bai 2008] Jing Bai et Jian-Yun Nie. *Adapting information retrieval to query contexts.* Information Processing and Management, vol. 44, no. 6, pages 1901–1922, Novembre 2008. (Cité en pages 6 et 54.)

[Banerjee 2009] Somnath Banerjee, Avinava Dubey, Jinesh Machchhar et Soumen Chakrabarti. *Efficient and accurate local learning for ranking.* In SIGIR 2009 Workshop on learning to rank for information retrieval (LR4RI 2009), 2009. (Cité en pages 67 et 182.)

[Beck 2009] Amir Beck et Marc Teboulle. *A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems.* SIAM Journal Imaging Sciences, vol. 2, no. 1, pages 183–202, Mars 2009. (Cité en page 134.)

[Bellot 2011] Patrice Bellot. Recherche d'information contextuelle assistée et personnalisée. Hermès-Lavoisier, 2011. (Cité en page 6.)

[Bian 2010] Jiang Bian, Tie-Yan Liu, Tao Qin et Hongyuan Zha. *Ranking with query-dependent loss for web search.* In Proceedings of the third ACM international conference on Web search and data mining, WSDM '10, pages 141–150, New York, NY, USA, 2010. ACM. (Cité en page 66.)

[Bigot 2011] Anthony Bigot, Claude Chrisment, Taoufiq Dkaki, Gilles Hubert et Josiane Mothe. *Fusing Different Information Retrieval Systems According*

to Query-Topics. Information Retrieval Journal, vol. 14, no. 6, pages 617–648, avril 2011. (Cité en page 54.)

[Blanco 2011] R. Blanco et H. Zaragoza. *Beware of Relatively Large but Meaningless Improvements*. Rapport technique, Yahoo ! Research 2011-001, 2011. (Cité en page 51.)

[Borlund 2003] Pia Borlund. *The IIR evaluation model : a framework for evaluation of interactive information retrieval systems*. Information research, vol. 8, no. 3, pages 8–3, 2003. (Cité en page 150.)

[Broder 2002] Andrei Broder. *A taxonomy of web search*. SIGIR Forum, vol. 36, no. 2, pages 3–10, Septembre 2002. (Cité en pages 54, 66 et 188.)

[Burges 1997] C. J. Burges, L. Kauffman, A. Smola et V. Vapnik. *Support Vector Regression Machines*. In Proceedings of the 9th conference on Neural Information Processing Systems, pages 155–161, 1997. (Cité en page 83.)

[Burges 2005] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton et Greg Hullender. *Learning to rank using gradient descent*. In Proceedings of the 22nd international conference on Machine learning, ICML '05, pages 89–96, New York, NY, USA, 2005. ACM. (Cité en pages 5 et 35.)

[Burges 2007] Christopher J.C. Burges, Robert Ragno et Quoc Viet Le. *Learning to Rank with Nonsmooth Cost Functions*. In B. Schölkopf, J. Platt et T. Hoffman, editeurs, Advances in Neural Information Processing Systems 19, pages 193–200. MIT Press, Cambridge, MA, 2007. (Cité en page 35.)

[Busa-Fekete 2012] R. Busa-Fekete, G. Szarvas, T. Élteto et B. Kégl. *An apple-to-apple comparison of Learning-to-rank algorithms in terms of Normalized Discounted Cumulative Gain*. In 20th European Conference on Artificial Intelligence (ECAI 2012) : Preference Learning : Problems and Applications in AI Workshop, Montpellier, France, 2012. LAL 12-311. (Cité en page 51.)

[Candes 2008] E.J. Candes, M.B. Wakin et S.P. Boyd. *Enhancing sparsity by reweighted l1 minimization*. Journal of Fourier Analysis and Applications, vol. 14, no. 5, pages 877–905, 2008. (Cité en pages 93, 99 et 129.)

[Cao 2006] Yunbo Cao, Jun Xu, Tie-Yan Liu, Hang Li, Yalou Huang et Hsiao-Wuen Hon. *Adapting ranking SVM to document retrieval*. In Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '06, pages 186–193, New York, NY, USA, 2006. ACM. (Cité en page 36.)

[Cao 2007] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai et Hang Li. *Learning to rank : from pairwise approach to listwise approach*. In Proceedings of the 24th international conference on Machine learning, ICML '07, pages 129–136, 2007. (Cité en pages 36, 38 et 138.)

[Carterette 2007] Ben Carterette et Rosie Jones. *Evaluating search engines by modeling the relationship between relevance and clicks*. In In Proceedings of the

Advances in Neural Information Processing Systems (NIPS, 2007. (Cité en page 151.)

[Chapelle 2009a] Olivier Chapelle. *Direct optimization for web search ranking.* In SIGIR 2009 Workshop on learning to rank for information retrieval (LR4RI 2009), 2009. (Cité en page 51.)

[Chapelle 2009b] Olivier Chapelle et Ya Zhang. *A dynamic bayesian network click model for web search ranking.* In Proceedings of the 18th international conference on World wide web, WWW '09, pages 1–10, New York, NY, USA, 2009. ACM. (Cité en page 153.)

[Chapelle 2010] Olivier Chapelle et Sathiya S. Keerthi. *Efficient algorithms for ranking with SVMs.* Information Retrieval, vol. 13, no. 3, pages 201–215, Juin 2010. (Cité en pages 5, 36 et 138.)

[Chapelle 2011a] Olivier Chapelle et Yi Chang. *Yahoo ! Learning to Rank Challenge Overview.* Journal of Machine Learning Research - Proceedings Track, vol. 14, pages 1–24, 2011. (Cité en pages xi, 47, 48, 49 et 51.)

[Chapelle 2011b] Olivier Chapelle, Yi Chang et Tie-Yan Liu. *Future directions in learning to rank.* Journal of Machine Learning Research - Proceedings Track, vol. 14, pages 91–100, 2011. (Cité en page 70.)

[Chartrand 2008] Rick Chartrand et Wotao Yin. *Iteratively reweighted algorithms for compressive sensing.* In ICASSP, pages 3869–3872, 2008. (Cité en page 98.)

[Cleverdon 1966] Cyril W. Cleverdon, Jack Mills et Michael. Keen. *Factors determining the performance of indexing systems.* 1966. (Cité en pages 22 et 23.)

[Cohen 1999] William W. Cohen, Robert E. Schapire et Yoram Singer. *Learning to order things.* Journal of Artificial Intelligence Research, vol. 10, no. 1, pages 243–270, Mai 1999. (Cité en page 35.)

[Cossock 2006] David Cossock et Tong Zhang. *Subset ranking using regression.* In Proceedings of the 19th annual conference on Learning Theory, COLT'06, pages 605–619. Springer-Verlag, 2006. (Cité en page 32.)

[Crammer 2001] Koby Crammer et Yoram Singer. *Pranking with Ranking.* In Advances in Neural Information Processing Systems 14, pages 641–647. MIT Press, 2001. (Cité en page 34.)

[Crammer 2002] Koby Crammer et Yoram Singer. *On the algorithmic implementation of multiclass kernel-based vector machines.* Journal of Machine learning Research, vol. 2, pages 265–292, 2002. (Cité en page 83.)

[Craswell 2003] Nick Craswell, David Hawking, Ross Wilkinson et Mingfang Wu. *Overview of the TREC 2003 Web Track.* In TREC, pages 78–92, 2003. (Cité en page 39.)

[Craswell 2008] Nick Craswell, Onno Zoeter, Michael Taylor et Bill Ramsey. *An experimental comparison of click position-bias models.* In Proceedings of the

international conference on Web search and web data mining, WSDM '08, pages 87–94, New York, NY, USA, 2008. ACM. (Cité en page 152.)

[Croft 2009] Bruce Croft, Donald Metzler et Trevor Strohman. Search engines : Information retrieval in practice. Addison-Wesley Publishing Company, USA, 1st édition, 2009. (Cité en page 16.)

[Dang 2010] Van Dang et Bruce Croft. *Feature Selection for Document Ranking using Best First Search and Coordinate Ascent.* In SIGIR Workshop on Feature Generation and Selection for Information Retrieval, 2010. (Cité en pages 54 et 78.)

[Deerwester 1990] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer et Richard Harshman. *Indexing by latent semantic analysis.* Journal of the American Society for Information Science, vol. 41, no. 6, pages 391–407, 1990. (Cité en page 16.)

[Delpech 2013] Estelle Delpech, Laurent Candillier, Léa Laporte et Samuel Phan. *Identification de compatibilité entre descripteurs de lieux et apprentissage automatique.* In Conférence Internationale Francophone sur l'Extraction et la Gestion de Connaissance (EGC), Toulouse, 29/01/2013-01/02/2013, pages 311–316, http ://www.antsearch.univ-tours.fr/rnti/, 2013. Revue des Nouvelles Technologies de l'Information (RNTI). (Cité en page 58.)

[Doan 2009] Biech Lien Doan, Joemon Jose, Massimo Melucci et Lynda Tamine, editeurs. Contextual and retrieval evaluation seeking evaluation workshop (cirse) in conjunction with the 31st european conference on information retrieval (ecir), toulouse, 28/03/2009 - 28/03/2009, http ://www.irit.fr/, mars 2009. IRIT. (Cité en page 6.)

[Fan 2001] Jianqing Fan et Runze Li. *Variable selection via nonconcave penalized likelihood and its oracle properties.* Journal of the American Statistical Association, no. 456, pages 1348–1360, 2001. (Cité en page 93.)

[Figueiredo 2003] Mario A. T. Figueiredo. *Adaptive sparseness fo rsupervised learning.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 9, pages 1150–1159, 2003. (Cité en page 99.)

[Freund 1995] Yoav Freund et Robert E. Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting.* In Proceedings of the Second European Conference on Computational Learning Theory, EuroCOLT'95, pages 23–37, London, UK, UK, 1995. Springer-Verlag. (Cité en page 33.)

[Freund 2003] Yoav Freund, Raj Iyer, Robert E. Schapire et Yoram Singer. *An efficient boosting algorithm for combining preferences.* Journal of Machine Learning Research, vol. 4, pages 933–969, 2003. (Cité en pages 5, 35 et 138.)

[Fu 1998] Wenjiang J Fu. *Penalized regressions : the bridge versus the lasso.* Journal of computational and graphical statistics, vol. 7, no. 3, pages 397–416, 1998. (Cité en page 92.)

[Gan 2008] Qingqing Gan, Josh Attenberg, Alexander Markowetz et Torsten Suel. *Analysis of geographic queries in a search engine log.* In Proceedings of the first international workshop on Location and the web, LOCWEB '08, pages 49–56, New York, NY, USA, 2008. ACM. (Cité en page 54.)

[Gasso 2009] Gilles Gasso, Alain Rakotomamonjy et Stéphane Canu. *Recovering sparse signals with a certain family of nonconvex penalties and DC programming.* IEEE Transaction on Signal Processing, vol. 57, no. 12, pages 4686–4698, 2009. (Cité en pages 92 et 136.)

[Geng 2007] Xiubo Geng, Tie-Yan Liu, Tao Qin et Hang Li. *Feature selection for ranking.* In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and development in information retrieval, pages 407–414, 2007. (Cité en pages 6, 54, 72, 76 et 182.)

[Geng 2008] Xiubo Geng, Tie-Yan Liu, Tao Qin, Andrew Arnold, Hang Li et Heung-Yeung Shum. *Query dependent ranking using K-nearest neighbor.* In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '08, pages 115–122, New York, NY, USA, 2008. ACM. (Cité en pages 6, 67 et 182.)

[Gorodnitsky 1997] Irina F. Gorodnitsky et Bhaskar D. Rao. *Sparse signal reconstruction from limited data using FOCUSS : a re-weighted minimum norm algorithm.* IEEE Transactions on Signal Processing, vol. 45, no. 3, pages 600–616, 1997. (Cité en page 98.)

[Guo 2009] Fan Guo, Chao Liu et Yi Min Wang. *Efficient multiple-click models in web search.* In Proceedings of the Second ACM International Conference on Web Search and Data Mining, WSDM '09, pages 124–131, New York, NY, USA, 2009. ACM. (Cité en page 152.)

[Guyon 2003] Isabelle Guyon et André Elisseeff. *An introduction to variable and feature selection.* Journal of Machine Learning Research, vol. 3, pages 1157–1182, 2003. (Cité en pages 69 et 95.)

[Harman 2010] Donna Harman. *Is the cranfield paradigm outdated ?* In SIGIR, page 1, 2010. (Cité en page 150.)

[Hastie 2003] Trevor Hastie, Robert Tibshirani et Jerome H. Friedman. The Elements of Statistical Learning. Springer, corrected édition, Juillet 2003. (Cité en pages 33 et 90.)

[Hersh 1994] William Hersh, Chris Buckley, T. J. Leone et David Hickam. *OHSUMED : an interactive retrieval evaluation and new large test collection for research.* In Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '94, pages 192–201, New York, NY, USA, 1994. Springer-Verlag New York, Inc. (Cité en page 39.)

[Hua 2010] Guichun Hua, Min Zhang, Yiqun Liu, Shaoping Ma et Liyun Ru. *Hierarchical feature selection for ranking.* In Proceedings of the 19th Interna-

tional Conference on World Wide Web, pages 1113–1114, 2010. (Cité en pages 77 et 78.)

[Huang 2008] Jim C. Huang et Brendan J. Frey. *Structured ranking learning using cumulative distribution networks*. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, Léon Bottou, Daphne Koller, Dale Schuurmans, Yoshua Bengio et Léon Bottou, editeurs, NIPS, pages 697–704. MIT Press, 2008. (Cité en page 38.)

[Hubert 2010] Gilles Hubert. *Recherche d'information et contexte*. Habilitation à diriger les recherches, Université Toulouse 3, Décembre 2010. (Cité en page 6.)

[Hunter 2004] David R. Hunter et Kenneth Lange. *A tutorial on MM algorithms*. American Statistician, pages 30–37, 2004. (Cité en page 136.)

[Jansen 2008] Bernard J. Jansen, Danielle L. Booth et Amanda Spink. *Determining the informational, navigational, and transactional intent of Web queries*. Information Processing and Management, vol. 44, no. 3, pages 1251 – 1266, 2008. (Cité en pages 54 et 188.)

[Joachims 2002] Thorsten Joachims. *Optimizing search engines using clickthrough data*. In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data mining, pages 133–142, 2002. (Cité en pages 5, 23, 88, 138, 151 et 158.)

[Joachims 2005] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke et Geri Gay. *Accurately interpreting clickthrough data as implicit feedback*. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '05, pages 154–161, New York, NY, USA, 2005. ACM. (Cité en page 151.)

[John 1994] George H. John, Ron Kohavi et Karl Pfleger. *Irrelevant Features and the Subset Selection Problem*. In Proceedings of the eleventh international conference on machine learning, pages 121–129. Morgan Kaufmann, 1994. (Cité en page 69.)

[Kang 2003] In-Ho Kang et GilChang Kim. *Query type classification for web document retrieval*. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, SIGIR '03, pages 64–71, New York, NY, USA, 2003. ACM. (Cité en pages 67 et 188.)

[Kira 1992] Kenji Kira et Larry A. Rendell. *A practical approach to feature selection*. In Proceedings of the ninth international workshop on Machine learning, ML92, pages 249–256, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc. (Cité en page 77.)

[Kleinberg 1999] Jon M. Kleinberg. *Authoritative sources in a hyperlinked environment*. Journal of ACM, vol. 46, no. 5, pages 604–632, Septembre 1999. (Cité en page 20.)

[Kohavi 1997] Ron Kohavi et George H. John. *Wrappers for Feature Subset Selection.* Artificial Intelligence, vol. 97, no. 1, pages 273–324, 1997. (Cité en page 78.)

[Kujala 2009] Jussi Kujala, Timo Aho et Tapio Elomaa. *A Walk from 2-Norm SVM to 1-Norm SVM.* In Proceedings of the 2009 Ninth IEEE International Conference on Data Mining, ICDM '09, pages 836–841, Washington, DC, USA, 2009. IEEE Computer Society. (Cité en pages 99 et 100.)

[Lai 2011] Hanjiang Lai, Yong Tang, Hai-Xia Luo et Yan Pan. *Greedy feature selection for ranking.* In CSCWD, pages 42–46, 2011. (Cité en page 80.)

[Lai 2013a] Hanjiang Lai, Yan Pan, Cong Liu, Liang Lin et Jie Wu. *Sparse Learning-to-Rank via an Efficient Primal-Dual Algorithm.* IEEE Transactions on Computers, vol. 62, no. 6, pages 1221–1233, 2013. (Cité en pages 6, 54, 80, 105, 128, 136, 140 et 182.)

[Lai 2013b] Hanjiang Lai, Yan Pan, Tang Yong et Ru Yong. *FSMRank : A Feature Selection Algorithme for Learning-to-Rank.* IEEE Transactions on Neural Networks and Learning Systems, 2013. (Cité en pages 6, 54, 80, 105, 120, 140, 173 et 182.)

[Lancaster 1974] Frederick W. Lancaster et Emily G. Fayen. Information retrieval on-line. Wiley-Becker and Hayes, 1974. (Cité en page 15.)

[Laporte 2010a] Léa Laporte. Learning to Rank : géolocalisation et personnalisation pour les systèmes de recherche d'information (Séminaire FREMIT, Toulouse, 20/09/2010-21/09/2010). 2010. (Cité en page 59.)

[Laporte 2010b] Léa Laporte. Sélection de variables et apprentissage de fonctions d'ordonnancement en recherche d'information (Journées de jeunes statisticiens de la Société Française de Statistique, Aussois, 05/09/2011-09/09/2011). 2010. (Cité en page 59.)

[Laporte 2011a] Léa Laporte. Data Mining pour la recherche d'information contextuelle - Learning to rank dans les moteurs de recherche d'information géoréférencés (Journée de fouille de données du GDR-I3, BU Lyon 1 Campus de la Doua, Lyon, 28 septembre 2011). 2011. (Cité en page 59.)

[Laporte 2011b] Léa Laporte. *RI et Contextualisation (student paper).* In Forum de Jeunes Chercheurs INFORSID 2011, Lille, 25/05/2011, pages 443–444. INFORSID, mai 2011. (Cité en pages 55 et 59.)

[Laporte 2012a] Léa Laporte. Approche proximale pour la sélection de variables en apprentissage d'ordonnancement via des SVM parcimonieux (Séminaire DocToMe, Toulouse, 20/12/2012). 2012. (Cité en page 59.)

[Laporte 2012b] Léa Laporte. Evaluation de la pertinence dans les moteur de recherche géoréférencés - Application des SVM (Séminaire FREMIT, Toulouse, 21/03/2012). 2012. (Cité en page 59.)

[Laporte 2012c] Léa Laporte. *Ordonnancement des résultats sur les moteurs de recherche : principes, limites et applications au géoréférencement (short pa-*

per). In Séminaire Veille Stratégique Scientifique et Technologique (Séminaire VSST), Ajaccio, 23/05/2012-25/05/2012. Veille Stratégique Scientifique et Technologique(VSST), mai 2012. (Cité en pages 57 et 59.)

[Laporte 2012d] Léa Laporte, Laurent Candillier, Sébastien Déjean et Josiane Mothe. *Évaluation de la pertinence dans les moteurs de recherche géoréférencés*. In INFORSID, pages 281–298, 2012. (Cité en pages 57, 59 et 154.)

[Laporte 2013a] Léa Laporte, Sébastien Déjean et Josiane Mothe. *Multiple Clicks Model for Web Search of Multi-clickable Documents (short paper)*. In International Conference on Enterprise Information Systems (ICEIS), Angers, 04/07/2013-07/07/2013, http ://www.scitepress.org/, 2013. SciTePress. (Cité en pages 57, 58 et 154.)

[Laporte 2013b] Léa Laporte, Sébastien Déjean et Josiane Mothe. *Reweighted algorithms for feature selection in ranking*. In Article en préparation, 2013. (Cité en pages 56 et 97.)

[Laporte 2013c] Léa Laporte. *De l'apprentissage d'ordonnancement à l'adaptation au contexte*. Document numérique, vol. 16, no. 1, pages 97–121, 2013. (Cité en pages 55, 58, 71, 151, 182, 186 et 187.)

[Laporte 2014] Léa Laporte, Rémi Flamary, Stéphane Canu, Sébastien Déjean et Josiane Mothe. *Non-convex regularizations for feature selection in Ranking with sparse SVM*. IEEE Transaction on Neural Networks and Leaning Systems, 2014. (Cité en pages 56, 58, 132, 135 et 141.)

[Li 2007] P. Li, C. Burges, Q. Wu, J. C. Platt, D. Koller, Y. Singer et S. Roweis. *McRank : Learning to Rank Using Multiple Classification and Gradient Boosting*. Advances in Neural Information Processing Systems, pages 845–582, 2007. (Cité en page 33.)

[Li 2011] Hang Li. *A Short Introduction to Learning to Rank*. IEICE Transactions, vol. 94-D, no. 10, pages 1854–1862, 2011. (Cité en page 49.)

[Liu 2009a] Chao Liu, Fan Guo et Christos Faloutsos. *BBM : bayesian browsing model from petabyte-scale data*. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '09, pages 537–546, New York, NY, USA, 2009. ACM. (Cité en page 153.)

[Liu 2009b] Tie-Yan Liu. *Learning to Rank for Information Retrieval*. Foundations and Trends in Information Retrieval, vol. 3, no. 3, pages 225–331, Mars 2009. (Cité en page 49.)

[Liu 2009c] Yi Liu, Liangjie Zhang, Ruihua Song, Jian-Yun Nie et Ji-Rong Wen. *Clustering queries for better document ranking*. In Proceedings of the 18th ACM conference on Information and knowledge management, CIKM '09, pages 1569–1572, New York, NY, USA, 2009. ACM. (Cité en page 68.)

[Liu 2011] Tie-Yan Liu. Learning to rank for information retrieval. Springer-Verlag Berlin Heidelberg, 2011. (Cité en pages xi, 5, 9, 15, 29, 34, 37, 38, 41, 43, 44, 46, 49, 50, 70, 94, 149, 150 et 182.)

[Long 2010] Bo Long, Olivier Chapelle, Ya Zhang, Yi Chang, Zhaohui Zheng et Belle Tseng. *Active learning for ranking through expected loss optimization.* In Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '10, pages 267–274, New York, NY, USA, 2010. ACM. (Cité en page 70.)

[Mangasarian 2006] Olvi L. Mangasarian. *Exact 1-Norm Support Vector Machines Via Unconstrained Convex Differentiable Minimization.* Journal of Machine Learning Research, vol. 7, pages 1517–1530, 2006. (Cité en page 92.)

[Mendoza 2009] Marcelo Mendoza et Juan Zamora. *Identifying the Intent of a User Query Using Support Vector Machines.* In Jussi Karlgren, Jorma Tarhio et Heikki Hyyrö, editeurs, String Processing and Information Retrieval, volume 5721 of *Lecture Notes in Computer Science*, pages 131–142. Springer Berlin Heidelberg, 2009. (Cité en page 188.)

[Mishne 2006] Gilad Mishne et Maarten De Rijke. *A study of blog search.* Advances in information retrieval, pages 289–301, 2006. (Cité en page 54.)

[Mothe 2005] Josiane Mothe et Ludovic Tanguy. *Linguistic features to predict query difficulty.* In In ACM SIGIR 2005 Workshop on Predicting Query Difficulty - Methods and Applications, 2005. (Cité en page 54.)

[Nallapati 2004] Ramesh Nallapati. *Discriminative models for information retrieval.* In Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '04, pages 64–71, 2004. (Cité en page 33.)

[Nie 2006] Lan Nie, Brian D. Davison et Xiaoguang Qi. *Topical link analysis for web search.* In Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '06, pages 91–98, New York, NY, USA, 2006. ACM. (Cité en page 21.)

[Nocedal 2006] Jorge Nocedal et Stephen J. Wright. Numerical optimization. Springer, New York, 2nd édition, 2006. (Cité en page 85.)

[Page 2001] Lawrence Page. *Method for node ranking in a linked database*, 09 2001. (Cité en page 21.)

[Pahikkala 2009] Tapio Pahikkala, Evgeni Tsivtsivadze, Antti Airola, Jouni Järvinen et Jorma Boberg. *An efficient algorithm for learning to rank from preference graphs.* Machine Learning, vol. 75, no. 1, pages 129–165, Avril 2009. (Cité en page 79.)

[Pahikkala 2010] Tapio Pahikkala, Antti Airola, Pekka Naula et Tapio Salakoski. *Greedy RankRLS : a Linear Time Algorithm for Learning Sparse Ranking Models.* In SIGIR 2010 Workshop on Feature Generation and Selection for Information Retrieval, pages 11–18, 2010. (Cité en page 79.)

[Pan 2009] Feng Pan, Tim Converse, David Ahn, Franco Salvetti et Gianluca Donato. *Feature selection for ranking using boosted trees.* In Proc. 18th ACM Conf. Information and Knowledge Management, pages 2025–2028, 2009. (Cité en page 79.)

[Pan 2011] Feng Pan, Tim Converse, David Ahn, Franco Salvetti et Gianluca Donato. *Greedy and Randomized Feature Selection for Web Search Ranking.* In Proceedings of the 2011 IEEE 11th International Conference on Computer and Information Technology, CIT '11, pages 436–442, Washington, DC, USA, 2011. IEEE Computer Society. (Cité en page 79.)

[Ponte 1998] Jay M. Ponte et W. Bruce Croft. *A language modeling approach to information retrieval.* In Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '98, pages 275–281, New York, NY, USA, 1998. ACM. (Cité en pages 16 et 18.)

[Qin 2005] Tao Qin, Tie-Yan Liu, Xu-Dong Zhang, Zheng Chen et Wei-Ying Ma. *A study of relevance propagation for web search.* In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '05, pages 408–415, New York, NY, USA, 2005. ACM. (Cité en page 21.)

[Qin 2008] Tao Qin, Xu-Dong Zhang, Ming-Feng Tsai, De-Sheng Wang, Tie-Yan Liu et Hang Li. *Query-level loss functions for information retrieval.* Information Processing and Management, vol. 44, no. 2, 2008. (Cité en page 39.)

[Qin 2010] Tao Qin, Tie-Yan Liu et Hang Li. *A general approximation framework for direct optimization of information retrieval measures.* Information Retrieval, 2010. (Cité en page 37.)

[Radlinski 2005] Filip Radlinski et Thorsten Joachims. *Query chains : learning to rank from implicit feedback.* In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pages 239–248. ACM, 2005. (Cité en page 151.)

[Robertson 1976] Stephen E. Robertson et Sparck K. Jones. *Relevance weighting of search terms.* Journal of the American Society for Information Science, vol. 27, no. 3, pages 129–146, 1976. (Cité en page 17.)

[Robertson 1994] Stephen E. Robertson et Stephen Walker. *Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval.* In Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '94, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc. (Cité en pages 16 et 17.)

[Rose 2004] Daniel E. Rose et Danny Levinson. *Understanding user goals in web search.* In Proceedings of the 13th international conference on World Wide Web, pages 13–19. ACM, 2004. (Cité en page 54.)

[Salton 1975] Gerard Salton, Anita Wong et Chung Shu Yang. *A vector space model for automatic indexing.* Communications of ACM, vol. 18, no. 11, pages 613–620, Novembre 1975. (Cité en page 16.)

[Scholkopf 2001] Bernhard Scholkopf et Alexander J. Smola. Learning with kernels : Support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge, MA, USA, 2001. (Cité en page 83.)

[Sculley 2009] D. Sculley et Google Inc. *Large scale learning to rank.* In In NIPS 2009 Workshop on Advances in Ranking, 2009. (Cité en pages 36 et 70.)

[Shakery 2003] Azadeh Shakery et ChengXiang Zhai. *Relevance Propagation for Topic Distillation UIUC TREC 2003 Web Track Experiments.* In TREC, pages 673–677, 2003. (Cité en page 21.)

[Sun 2009] Zhengya Sun, Tao Qin, Qing Tao et Jue Wang. *Robust sparse rank learning for non-smooth ranking measures.* In Proc. 32nd Int. ACM SIGIR Conf. Research and development in information retrieval, pages 259–266, 2009. (Cité en pages 6, 54, 80 et 128.)

[Taylor 2008] Michael Taylor, John Guiver, Stephen Robertson et Tom Minka. *SoftRank : optimizing non-smooth rank metrics.* In Proceedings of the 2008 International Conference on Web Search and Data Mining, WSDM '08, pages 77–86, New York, NY, USA, 2008. ACM. (Cité en page 37.)

[Tibshirani 1996] Robert Tibshirani. *Regression Shrinkage and Selection Via the Lasso.* Journal of the Royal Statistical Society, Series B, vol. 58, pages 267–288, 1996. (Cité en pages 90 et 91.)

[Tsai 2007] Ming-Feng Tsai, Tie-Yan Liu, Tao Qin, Hsin-Hsi Chen et Wei-Ying Ma. *FRank : a ranking method with fidelity loss.* In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '07, pages 383–390, New York, NY, USA, 2007. ACM. (Cité en page 35.)

[Vapnik 1995] Vladimir N. Vapnik. The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, USA, 1995. (Cité en pages 83 et 90.)

[Verbene 2009] Suzan Verbene, Hans Van Halteren, Daphne Theijssen, Stephan Raaijmakers et Lou Boves. *Learning to rank qa data.* In SIGIR 2009 Workshop on learning to rank for information retrieval (LR4IR2009), 2009. (Cité en page 49.)

[Volkovs 2009] Maksims N. Volkovs et Richard S. Zemel. *BoltzRank : learning to maximize expected ranking gain.* In Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, pages 1089–1096, New York, NY, USA, 2009. ACM. (Cité en page 38.)

[Weston 1999] Jason Weston et Chris Watkins. *Support Vector Machines for Multi-Class Pattern Recognition.* In Proceedings of the European Symposium on Artificial Networks ESANN'1999, pages 219–224, 1999. (Cité en page 83.)

[Weston 2003] Jason Weston, André Elisseeff, Bernhard Schölkopf et Mike Tipping. *Use of the zero norm with linear models and kernel methods.* Journal of Machine Learning Research, vol. 3, pages 1439–1461, 2003. (Cité en pages 91, 93, 98, 101, 102, 104 et 183.)

[Xia 2008] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang et Hang Li. *Listwise approach to learning to rank : theory and algorithm.* In Proceedings of the 25th international conference on Machine learning, ICML '08, pages 1192–1199, New York, NY, USA, 2008. ACM. (Cité en pages 36, 38, 39 et 49.)

[Xu 2007] Jun Xu et Hang Li. *AdaRank : a boosting algorithm for information retrieval.* In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '07, pages 391–398, 2007. (Cité en pages 38 et 138.)

[Xue 2005] Gui-Rong Xue, Qiang Yang, Hua-Jun Zeng, Yong Yu et Zheng Chen. *Exploiting the hierarchical structure for link analysis.* In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '05, pages 186–193, New York, NY, USA, 2005. ACM. (Cité en page 21.)

[Yeh 2007] Jen Y. Yeh, Jung Y. Lin, Hao R. Ke et Wei P. Yang. *Learning to Rank for Information Retrieval Using Genetic Programming.* In Thorsten Joachims, Hang Li, Tie Y. Liu et Chengxiang Zhai, editeurs, SIGIR 2007 workshop : Learning to Rank for Information Retrieval, Juillet 2007. (Cité en page 38.)

[Yilmaz 2010] Emine Yilmaz et Stephen Robertson. *On the choice of effectiveness measures for learning to rank.* Information Retrieval, vol. 13, no. 3, pages 271–290, Juin 2010. (Cité en page 37.)

[Yu 2009] Hwanjo Yu, Jinoh Oh et Wook-Shin Han. *Efficient feature weighting methods for ranking.* In Proceedings of the 18th ACM Conference on Information and knowledge management, pages 1157–1166, 2009. (Cité en pages 54, 77 et 78.)

[Yue 2007] Yisong Yue, Thomas Finley, Filip Radlinski et Thorsten Joachims. *A support vector method for optimizing average precision.* In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '07, pages 271–278, 2007. (Cité en page 37.)

[Zha 2006] Hongyuan Zha, Zhaohui Zheng, Haoying Fu et Gordon Sun. *Incorporating query difference for learning retrieval functions in world wide web search.* In CIKM, pages 307–316, 2006. (Cité en page 66.)

[Zhai 2001] Chengxiang Zhai et John Lafferty. *A study of smoothing methods for language models applied to Ad Hoc information retrieval.* In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '01, pages 334–342, New York, NY, USA, 2001. ACM. (Cité en pages 18 et 20.)

[Zhang 2009] Min Zhang, Da Kuang, Guichun Hua, Yiqun Liu et Shaoping Ma. *Is learning to rak effective for web search ?* In SGIR 2009 Workshop on learing to rank for information retrieval (LR4IR 2009), 2009. (Cité en pages 49 et 50.)

[Zhang 2010] Yingsong Zhang et N. Kingsbury. *FAST L0-based sparse signal recovery*. In Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on, pages 403–408, 2010. (Cité en pages 93 et 98.)

[Zhu 2003] Ji Zhu, Saharon Rosset, Trevor Hastie et Rob Tibshirani. *1-norm Support Vector Machines*. In Neural Information Processing Systems, page 16. MIT Press, 2003. (Cité en page 92.)

[Zoeter 2008] Onno Zoeter, Michael Taylor, Ed Snelson, John Guiver, Nick Craswell et Martin Szummer. *A Decision Theoretic Framework for Ranking using Implicit Feedback*. In SIGIR 2008 Workshop : Learning to Rank for Information Retrieval, pages 31–38, Juillet 2008. (Cité en page 37.)

[Zou 2006] Hui Zou. *The adaptive lasso and its oracle properties*. Journal of the Amererican Statistical Association, vol. 101, no. 476, pages 1418–1429, 2006. (Cité en page 92.)