Sélection de variables en apprentissage d'ordonnancement

Évaluation des SVM pondérés

Léa Laporte [1], **Sébastien Déjean** [2], **Josiane Mothe** [3]

1. *LIRIS, UMR 5205, INSA Lyon, Université de Lyon, France*
   lea.laporte@insa-lyon.fr
2. *IMT, UMR 5219, Université Paul Sabatier, Université de Toulouse, France*
   sebastien.dejean@math.univ-toulouse.fr
3. *IRIT, UMR 5505, ESPE Midi-Pyrénées, Université de Toulouse, France*
   josiane.mothe@irit.fr

RÉSUMÉ. *Sélectionner les caractéristiques les plus utiles et les moins redondantes au sein des fonctions d'ordonnancement et réduire les temps d'exécution sont des enjeux en apprentissage d'ordonnancement. Les algorithmes de sélection de variables basés sur les SVM régularisés sont des approches prometteuses dans ce cadre. Dans cet article, nous proposons de nouvelles méthodes de sélection de variables en apprentissage d'ordonnancement basées sur des approches de pondération des SVM en norme ℓ_2. Nous proposons une adaptation d'une méthode ℓ_2-AROM qui résout des SVM en norme ℓ_0 et un algorithme de pondération de la norme ℓ_2 qui résout les problèmes en norme ℓ_0 et ℓ_1. Nos évaluations sur des jeux de données industriels et de référence montrent que les méthodes proposées sont jusqu'à 7 fois plus rapides et 10 fois plus parcimonieuses que l'état de l'art, pour des qualités d'ordonnancement équivalentes.*

ABSTRACT. *To select the most useful and the least redundant features to be used in ranking function to reduce computational costs is an issue in learning to rank (LTR). Regularized SVM are promising approaches in this context. In this paper, we propose new feature selection algorithms for LTR based on weighted SVM. We investigate an ℓ_2-AROM algorithm to solve the ℓ_0 norm problem and a weighted ℓ_2 algorithm to solve ℓ_0 et ℓ_1 norm problems. Experiments on benchmarks and commercial datasets show that our algorithms are up to 10 times faster and use up to 7 times less features than state-of-the-art methods, with similar ranking performance.*

MOTS-CLÉS : *apprentissage d'ordonnancement, sélection de variables, SVM pondérés.*

KEYWORDS: *learning to rank, feature selection, weighted SVM algorithms.*

DOI:10.3166/DN.18.1.97-121

1. Introduction

L'apprentissage d'ordonnancement, ou *learning-to-rank*, a pour objectif l'optimisation des fonctions d'ordonnancement, ou modèles, utilisées par les systèmes de recherche d'information (RI) pour ordonner les documents retrouvés en réponse à une requête. Les algorithmes d'apprentissage d'ordonnancement utilisent des jeux de données composés de couples requête-document associés à des jugements de pertinence pour apprendre la fonction qui optimise l'ordonnancement. Cette fonction prédit ensuite l'ordre optimal des documents à restituer pour des requêtes ultérieures.

Les travaux existants se concentrent majoritairement sur la proposition d'algorithmes d'apprentissage d'ordonnancement (Burges *et al*., 2005 ; Cao *et al*., 2007 ; Chapelle, Keerthi, 2010 ; Freund *et al*., 2003 ; Joachims, 2002 ; Xu, Li, 2007). Alors que le nombre de caractéristiques combinées par les fonctions d'ordonnancement ne cesse d'augmenter (de quelques dizaines à plusieurs milliers), deux problématiques majeures apparaissent (Geng *et al*., 2007). Premièrement, l'augmentation du nombre de caractéristiques affecte les temps d'exécution des algorithmes, d'extraction des caractéristiques et de réponse des systèmes. Deuxièmement, des caractéristiques non pertinentes ou redondantes peuvent être présentes dans les données, pouvant rendre certaines approches d'apprentissage instables. La sélection de variables, qui contrôle les caractéristiques utilisées par les modèles, est un moyen prometteur de résoudre ces problèmes.

Trois types d'approches ont été proposées en sélection de variables pour l'apprentissage d'ordonnancement. Les *méthodes par filtre* considèrent la sélection comme une étape de prétraitement. Le sous-ensemble de caractéristiques pertinentes est déterminé avant la phase d'apprentissage, indépendamment de cette dernière. Les *approches encapsulantes* sont aussi des étapes de prétraitement. Elles dépendent de l'algorithme d'apprentissage, qu'elles utilisent pour déterminer le meilleur sous-ensemble de caractéristiques. Les *méthodes embarquées* effectuent simultanément apprentissage et sélection, cette dernière étant spécifique de l'algorithme d'apprentissage considéré.

Geng *et al*. (2007) ont été les premiers à proposer une méthode de sélection de variables spécifique à l'apprentissage d'ordonnancement. Leur approche de type filtre, nommée *Greedy search Algorithm for feature Selection* (GAS), utilise un algorithme glouton pour sélectionner les caractéristiques les plus pertinentes, en minimisant leur similarité et en maximisant leur importance pour l'ordonnancement. D'autres travaux ont proposé des versions par filtre et encapsulantes d'un même algorithme. Hua *et al*. (2010) utilisent l'algorithme des k-moyennes pour créer des groupes de caractéristiques similaires. Les auteurs sélectionnent alors k variables représentatives, une par groupe, qu'ils utilisent pour apprendre la fonction d'ordonnancement. Yu *et al*. (2009) ont proposé deux méthodes basées sur une adaptation de l'algorithme Relief proposé en classification par Kira et Rendell (1992).

Certaines études se sont intéressées spécifiquement aux approches encapsulantes. Pan *et al*. (2009) utilisent les arbres de décision et un algorithme glouton pour effectuer la sélection. Dang et Croft (2010) adaptent l'algorithme *Best First Search* (Kohavi,

John, 1997) à l'apprentissage d'ordonnancement, tandis que Pahikkala *et al*. (2010) considèrent un algorithme glouton basé sur leur méthode RankRLS. Enfin, certains travaux comme ceux de Lai *et al*. (2011) ont considéré des approches de sélection de type Forward-Backward. Il est globalement difficile de comparer ces différentes approches, car elles ont été évaluées sur des jeux de données différents, sans que le code source soit disponible en ligne.

Relativement peu de travaux se sont intéressés aux approches embarquées. Ces études considèrent des problèmes d'optimisation régularisés, c'est-à-dire faisant intervenir un terme dit de régularisation, ou pénalité. Des pénalités usuelles sont par exemple la régularisation ℓ_0, telle que $\forall \mathbf{x} \in \mathbb{R}^m$, $\|\mathbf{x}\|_0 = \sum_{\mathbf{x}_i \neq 0} 1$, représentant le nombre d'éléments non nuls d'un vecteur et les régularisations ℓ_p , $p \geq 1$, telles que $\|\mathbf{x}\|_p = (\sum_{i=1}^{m} |\mathbf{x}_i|^p)^{\frac{1}{p}}$. L'utilisation de régularisations permet de contrôler le sur-apprentissage et dans le cas de régularisations parcimonieuses, de supprimer des caractéristiques tout en apprenant la combinaison optimale des critères restants. Dans ce cadre, une approche naturelle est de minimiser la régularisation ℓ_0. Néanmoins, cette régularisation étant non convexe, non différentiable et non continue, le problème d'optimisation ne peut être résolu par des algorithmes classiques. La régularisation ℓ_1, ou Lasso (Hastie *et al*., 2003), proposée pour la sélection de variables en discrimination, lui est généralement préférée. Sun *et al*. (2009) ont ainsi proposé RSRank, un algorithme parcimonieux qui optimise une mesure de RI via un problème d'optimisation régularisée en norme ℓ_1. Lai, Pan, Liu *et al*. (2013) ont considéré un algorithme primal-dual, nommé FenchelRank, pour la résolution de Séparateurs à Vaste Marge (SVM) régularisés en norme ℓ_1. L'utilisation de la régularisation mixte $\ell_1 - \ell_2$ a également été proposée dans l'algorithme FSMRank (Lai, Pan, Yong, Yong, 2013). Les auteurs de ces deux derniers algorithmes ont montré que ces approches amélioraient la qualité de l'ordonnancement comparativement aux méthodes de référence GAS et RSRank. D'autres travaux considèrent des régularisations non convexes (Laporte, Flamary *et al*., 2014).

Cet article est une version étendue des travaux présentés à la conférence CORIA 2014 (Laporte, Déjean, Mothe, 2014), augmentés d'une analyse de la redondance des caractéristiques, d'expérimentations sur deux jeux de données de référence supplémentaires et d'une évaluation sur un jeu de données de grande dimension issu du monde industriel. Dans ces travaux, nous nous concentrons sur les SVM parcimonieux régularisés, qui sont des techniques de sélection efficaces. Nous proposons d'utiliser des approches de pondération de type moindres carrés pondérés (IRLS). Celles-ci résolvent des problèmes d'optimisation difficiles à résoudre par itérations successives de problèmes plus faciles à résoudre, comme des SVM en norme ℓ_2. Nous proposons trois algorithmes pour effectuer la sélection de variables via des SVM régularisés en norme ℓ_0 ou ℓ_1. Nous souhaitons évaluer la capacité des approches de pondération à apprendre des fonctions d'ordonnancement parcimonieuses et de bonne qualité. Le premier, Rankℓ_2-AROM, considère un problème régularisé en norme ℓ_0 résolu par une approche de type ℓ_2-AROM (Weston *et al*., 2003). Les deux derniers algorithmes sont des approches de type IRLS, adaptées à l'apprentissage d'ordonnancement. L'un, RankRWFS-ℓ_1, résout le problème en norme ℓ_1 tandis que l'autre, RankRWFS-ℓ_0,

considère la régularisation ℓ_0. Tous deux utilisent la même structure, mais des règles de pondération de la norme ℓ_2 différentes permettent d'approcher chacune des deux régularisations. Nous débutons nos expérimentations par une analyse préliminaire mettant en évidence la redondance des caractéristiques présentes dans les jeux de données, ce qui motive et justifie l'utilisation d'algorithmes de sélection de variables. Nos expérimentations sur des jeux de données de référence en apprentissage d'ordonnancement montrent que nos approches utilisent de 2 à 7 fois moins de caractéristiques que les méthodes de l'état de l'art, tout en conservant la même qualité d'ordonnancement et en étant plus rapides. Une expérimentation sur un jeu de données de plus grande dimension, issu d'un moteur de recherche commercial, confirme que nos méthodes sont plus parcimonieuses que l'état de l'art et peuvent être utilisées dans un contexte industriel.

La section 2 décrit nos propositions. La section 3 détaille le protocole expérimental. La section 4 présente une analyse préliminaire de la redondance des caractéristiques dans les jeux de données. La section 5 présente les résultats. Nous discutons des perspectives en section 6.

2. Sélection de variables via des SVM itérativement pondérés

2.1. Cadre général

En apprentissage d'ordonnancement, des algorithmes d'apprentissage sont utilisés pour optimiser le classement de documents ou de pages web. Nous nous plaçons spécifiquement dans le cadre d'approches d'apprentissage d'ordonnancement par paire, dont l'objectif est de prédire des préférences entre documents. Un document d_i est préféré au document d_j pour une requête q s'il doit être classé plus haut dans la liste de résultats (par exemple si d_i est plus pertinent que d_j).

Considérons $\mathcal{Q} = \{q_k\}_{k=1,\ldots,Q}$ un ensemble de Q requêtes et $\mathcal{D} = \{d_i\}_{i=1,\ldots,N}$ l'ensemble des N documents associés à ces requêtes. Pour une requête q_k fixée, chaque document d_i est représenté par le vecteur de caractéristiques $\mathbf{x}_i^k \in \mathbb{R}^m$, où m est le nombre de caractéristiques. Les caractéristiques peuvent être propres à la requête (par exemple le nombre de termes de la requête), propres au document (par exemple le score PageRank du document), ou bien représenter une similarité entre la requête et le document (par exemple, le score obtenu avec le modèle BM25 entre la requête et le document).

En apprentissage d'ordonnancement par paire, nous souhaitons disposer des paires de documents (d_i, d_j) telles qu'une relation de préférence entre ces deux documents existe et déterminer un jugement de préférence $y^{(i,j)} \in \{-1, +1\}$ associé à chaque paire [1]. Pour une requête q_k fixée, nous notons $d_i \succ_{q_k} d_j$ si le document d_i est préféré au document d_j et $y^{(i,j)} = 1$. À l'inverse nous notons $d_j \succ_{q_k} d_i$ si le document d_j est préféré au document d_i et $y^{(i,j)} = -1$. Notons qu'il n'existe pas néces-

1. La relation de préférence peut par exemple être déduite des degrés de pertinence de chaque document pour la requête q_k.

sairement une relation de préférence entre toutes les paires de documents distincts, notamment si des documents ne sont pas pertinents pour la requête. Pour chaque requête q_k, nous pouvons construire un ensemble $\mathcal{P}_k$ constitué des paires de documents (d_i, d_j) pour lesquelles une préférence existe pour la requête q_k et un ensemble $\mathcal{Y}_k$ constitué des jugements de préférences $y^{(i,j)}$ associés à chaque paire. Chaque paire $p \in \mathcal{P}_k, p = 1, \ldots, Card(\mathcal{P}_k)$ est alors représentée dans l'espace des caractéristiques par le vecteur $\tilde{\mathbf{x}}^p$ tel que $\tilde{\mathbf{x}}^p = (\mathbf{x}_i^k - \mathbf{x}_j^k)^\top$, vecteur ligne de $\mathbb{R}^m$, auquel est associé le jugement de préférence y^p. Ainsi, si nous considérons par exemple la requête q, trois documents d_1, d_2 et d_3 et trois caractéristiques tels que d_1 est le seul document pertinent pour q et $\mathbf{x}_1 = (0.8, 0.9, 0.6)$, $\mathbf{x}_2 = (0.1, 0.1, 0.2)$ et $\mathbf{x}_3 = (0.3, 0.1, 0.1)$. Alors $d_1 \succ_q d_2$ et $y^{1,2} = 1$, $d_1 \succ_q d_3$ et $y^{1,3} = 1$ et nous pouvons extraire deux paires de documents p_1 et p_2 pour lesquelles il existe une relation de préférence pour la requête q. Chaque paire est représentée respectivement dans l'espace des caractéristiques par les vecteurs $\tilde{\mathbf{x}}_1$ et $\tilde{\mathbf{x}}_2$ tels que $\tilde{\mathbf{x}}_1 = (\mathbf{x}_1 - \mathbf{x}_2)^\top = (0.7, 0.8, 0.4)^\top$ et $\tilde{\mathbf{x}}_2 = (\mathbf{x}_1 - \mathbf{x}_3)^\top = (0.5, 0.8, 0.5)^\top$. Nous pouvons alors définir l'ensemble $\mathcal{P} = \cup_k \mathcal{P}_k$ de toutes les paires $\tilde{\mathbf{x}}_p$ pour toutes les requêtes de $\mathcal{Q}$ et $\mathcal{Y} = \cup_k \mathcal{Y}_k$ l'ensemble des jugements de préférences associés, pour prendre en compte l'ensemble des requêtes.

Le problème d'optimisation des SVM linéaires non parcimonieux est alors défini de la façon suivante (Chapelle, Keerthi, 2010) :

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{p=1}^{P} L(\tilde{\mathbf{x}}^p\mathbf{w}) \tag{1}$$

où $\mathbf{w} \in \mathbb{R}$ est le vecteur colonne des poids affectés aux caractéristiques, $\|\mathbf{w}\|_2^2 = \sum_{i=1}^m \mathbf{w}_i^2$ est le terme de régularisation en norme ℓ_2, $\sum_{p=1}^P L(\tilde{\mathbf{x}}^p\mathbf{w})$ est le terme d'ajustement aux données et C est un paramètre strictement positif permettant de contrôler le compromis entre qualité d'ajustement et capacité de généralisation du modèle appris. Ce paramètre est choisi par validation croisé.

Dans cet article, nous nous concentrons sur des formulations parcimonieuses des SVM. Nous nous intéressons aux SVM régularisés en norme ℓ_0 définis comme suit :

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|_0 + C\sum_{p=1}^{P} L(\tilde{\mathbf{x}}^p\mathbf{w}) \tag{2}$$

où $\|\mathbf{w}\|_0 = \sum_{\mathbf{w}_i \neq 0} 1$.

Minimiser la norme ℓ_0 revient à minimiser le nombre de caractéristiques intervenant dans le modèle linéaire. Bien qu'il s'agisse d'une façon naturelle de procéder à la sélection de variables, ce problème est en pratique NP-complet. Du fait du caractère non continu, non différentiable et non convexe de la norme ℓ_0, il est particulièrement difficile à résoudre par les algorithmes usuels d'optimisation. Le problème régularisé en norme ℓ_1

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|_1 + C\sum_{p=1}^{P} L(\tilde{\mathbf{x}}^p\mathbf{w}) \tag{3}$$

tel que $\|\mathbf{w}\|_1 = \sum_{i=1}^{m} |\mathbf{w}_i|$, constitue une relaxation convexe du précédent et lui est généralement préféré, car il est plus facile à résoudre. Cependant, la norme ℓ_1, bien que convexe, est non différentiable en zéro et des algorithmes spécifiques doivent être utilisés.

Les problèmes précédents en apprentissage d'ordonnancement par paire sont similaires à ceux rencontrés en discrimination. Chapelle et Keerthi justifient l'utilisation de méthodes spécifiques à l'apprentissage d'ordonnancement pour des raisons de temps de calcul (Chapelle, Keerthi, 2010). Le choix du modèle optimal s'effectue non pas en minimisant le risque quadratique, mais en maximisant une mesure de recherche d'évaluation sur le jeu de données de validation, ce qui justifie la proposition de méthodes spécifiques à l'apprentissage d'ordonnancement (Geng *et al.*, 2007).

Dans cet article, nous nous intéressons aux approches de pondération en norme ℓ_2 pour l'approximation des problèmes parcimonieux. Nous souhaitons évaluer l'apport de ces algorithmes et des différentes régularisations pour la sélection de variables en apprentissage d'ordonnancement. Nous souhaitons déterminer s'il existe des différences de performances suivant la régularisation considérée, notamment sur la qualité de l'ordonnancement ou les temps d'exécution. Dans la suite, nous proposons trois algorithmes de sélection de variables spécifiques à l'apprentissage d'ordonnancement : Rank-ℓ_2-AROM, basé sur une approche de type ℓ_2-AROM (Weston *et al.*, 2003) ainsi que RankRWFS-ℓ_1 et RankRWFS-ℓ_0 qui utilisent des règles de pondération spécifiques aux normes ℓ_1 et ℓ_0 respectivement.

2.2. Rank-ℓ_2-AROM, approximation de la norme ℓ_0 pour l'apprentissage d'ordonnancement

L'algorithme ℓ_2-AROM (ℓ_2 *Approximation of the zeRO-norm Minimization*) a été initialement proposé dans le cadre de la discrimination (Weston *et al.*, 2003). Il permet d'approcher la solution des SVM régularisés en norme ℓ_0 par itérations successives des SVM pondérés en norme ℓ_2.

À chaque itération t, un algorithme standard de résolution des SVM en norme ℓ_2 est utilisé. Le vecteur de poids obtenus, noté $\mathbf{w}^{(t)}$ est utilisé pour pondérer les valeurs des caractéristiques courantes. Ainsi, pour chaque observation $\mathbf{x}^{j(t)}$, la valeur de chaque caractéristique i, notée $\mathbf{x}_i^{j(t)}$, est multipliée par le poids courant, tel que $\mathbf{x}_i^{j(t)} \leftarrow \mathbf{w}_i^{(t)} \mathbf{x}_i^{j(t-1)}$. Le processus est répété jusqu'à convergence, ou, dans le cas de la sélection de variables, jusqu'à ce que le nombre de variables souhaitées soit atteint. Dans ce dernier cas, le sous-ensemble de variables sélectionnées est utilisé pour apprendre les poids finaux via l'algorithme standard.

Nous adaptons cette méthode à l'apprentissage d'ordonnancement.Nous utilisons un solveur spécifique à l'apprentissage d'ordonnancement. Par ailleurs, il est important de noter que la norme ℓ_2 n'étant pas parcimonieuse, il est nécessaire de fixer un seuil de nullité des poids du vecteur $\mathbf{w}$, afin d'éliminer les caractéristiques non pertinentes. L'algorithme que nous obtenons est nommé Rank-ℓ_2-AROM.

2.3. RankRWFS : approche de repondération des normes ℓ_1 et ℓ_0 pour la sélection de variables en apprentissage d'ordonnancement

L'algorithme Rankℓ_2-AROM est conçu pour approcher la solution du problème en norme ℓ_0 uniquement. Nous proposons une approche basée sur les SVM itérativement pondérés qui considère les normes ℓ_0 et ℓ_1, via des règles de pondération spécifiques. Ces travaux sont inspirés de l'algorithme RW proposé dans le cadre de la discrimination pour l'approximation de la régularisation ℓ_1 (Kujala *et al.*, 2009).

2.3.1. Approximation des normes ℓ_1 et ℓ_0

Le principe des méthodes d'approximation par pondération du problème en norme ℓ_2, comme par exemple l'algorithme RW, est de ré-écrire les normes parcimonieuses pour faire apparaître la régularisation ℓ_2. Considérons le problème des SVM par paire en norme ℓ_1 défini à la formule (3). Soit $\mathbb{I} = \{i|\mathbf{w}_i \neq 0\}$, la régularisation ℓ_1 peut s'écrire :

$$\|\mathbf{w}\|_1 = \sum_{i=1}^{m} |\mathbf{w}_i| = \sum_{i\in\mathbb{I}} |\mathbf{w}_i| = \sum_{i\in\mathbb{I}} \frac{\mathbf{w}_i^2}{|\mathbf{w}_i|} \tag{4}$$

Soit un instant t donné, $p_i^{(t)} = \frac{1}{\sqrt{|\mathbf{w}_i^{(t)}|}}$ et $\mathbf{r}_i^{(t)} = p_i^{(t)}\mathbf{w}_i^{(t)}$, alors la norme ℓ_1 devient :

$$\|\mathbf{w}^{(t)}\|_1 = \sum_{i\in\mathbb{I}} (p_i^{(t)}\mathbf{w}_i^{(t)})^2 = \sum_{i\in\mathbb{I}} (\mathbf{r}_i^{(t)})^2 = \|\mathbf{r}^{(t)}\|_2^2$$

Un changement de variables similaire permet de ré-écrire la fonction de perte. Soit $P(t)$ la matrice diagonale de $R^{m\times m}$ telle que $\mathbf{r}^{(t)} = P^{(t)}\mathbf{w}^{(t)}$. Elle s'écrit : $P(t) = diag(p_i^{(t)})_{i\in\{1,\ldots,m\}}$. En constatant que $\tilde{\mathbf{x}}^p\mathbf{w} = \tilde{\mathbf{x}}^p P^{-1} P\mathbf{w} = \tilde{\mathbf{x}}^p P^{-1}\mathbf{r}$ et en posant $\tilde{\mathbf{z}}^p = \tilde{\mathbf{x}}^p P^{-1}$, le problème en norme ℓ_1 équivaut à un instant t au problème pondéré suivant :

$$\min_{\mathbf{r}} \frac{1}{2}\|\mathbf{r}\|_2^2 + C\sum_{p=1}^{P} L(\tilde{\mathbf{z}}^p\mathbf{r}) \tag{5}$$

Un raisonnement similaire peut être mené dans le cadre du problème d'optimisation en norme ℓ_0 (*cf.* équation (2)) en utilisant la ré-écriture suivante :

$$\|\mathbf{w}\|_0 = \sum_{i=1}^{m} \mathbb{1}_{\{\mathbf{w}_i\neq 0\}} = \sum_{\mathbf{w}_i\neq 0} 1 = \sum_{\mathbf{w}_i\neq 0} \frac{\mathbf{w}_i^2}{\mathbf{w}_i^2} \tag{6}$$

et en posant $p_i^{(t)} = \frac{1}{|\mathbf{w}_i^{(t)}|}$.

L'algorithme RW (Kujala *et al.*, 2009) est une implémentation de l'approximation du problème régularisé en norme ℓ_1 par pondération de la norme ℓ_2 dans le cas de

la discrimination. À chaque itération t, considérant un vecteur $\mathbf{v}^{(t)} \in \mathbb{R}^m$ tel que $\mathbf{v}^{(1)} = [1 \dots 1]$ et $\forall t > 1$, $\mathbf{v}^{(t)} = [\prod_{s=1}^{(t-1)} \mathbf{w}_1^{(s)} \dots \prod_{s=1}^{(t-1)} \mathbf{w}_m^{(s)}]$, une observation $\mathbf{x}$ et une caractéristique i, la mise à jour $\mathbf{z}_i^{(t)}$ de $\mathbf{x}_i^{(t)}$ est calculée comme suit :

$$\mathbf{z}_i^{(t)} \leftarrow \sqrt{|\mathbf{w}_i^{(t-1)} \mathbf{v}_i^{(t-1)}|} \mathbf{x}_i \tag{7}$$

Le processus est itéré T fois, où T est le nombre d'itérations fixé par l'utilisateur. À la dernière itération, les vecteurs finaux $\mathbf{w}$ et $\mathbf{v}$ sont multipliés pour obtenir les poids. Les valeurs inférieures à un seuil donné sont alors annulées. L'algorithme RW n'est pas spécifiquement conçu pour la sélection de variables en apprentissage d'ordonnancement, ni pour la prise en compte de la régularisation ℓ_0. Nous proposons une approche générique que nous nommons RWFS, qui permet de considérer les problèmes avec pénalité ℓ_0 et ℓ_1 en sélectionnant la règle de mise à jour spécifique à chaque régularisation. Nous nommons ces algorithmes RankRWFS-ℓ_0 et RankRWFS-ℓ_1.

2.3.2. RankRWFS : repondération en norme ℓ_2 pour la sélection de variables en apprentissage d'ordonnancement

Notre approche fournit un algorithme générique pour la sélection de variables et l'apprentissage du modèle final, pouvant utiliser aussi bien la pénalité ℓ_0 que la pénalité ℓ_1. Cette méthode est basée sur trois points clés :

1. Le choix de la règle de mise à jour adéquate pour chaque régularisation parcimonieuse (ℓ_0 ou ℓ_1) pour la résolution itérative par pondération de la norme ℓ_2 ;
2. La définition d'une structure de contrôle du nombre de variables à atteindre et
3. L'adaptation à l'apprentissage d'ordonnancement.

Règles de mises à jour : nous utilisons les ré-écritures présentées respectivement aux équations (4) et (6). Dans le cas de la norme ℓ_1, nous considérons la règle de mise à jour définie à l'équation (7), tandis que la résolution de la norme ℓ_0 fait appel à la règle de pondération suivante :

$$\mathbf{z}_i^{(t)} \leftarrow |\mathbf{w}_i^{(t-1)} \mathbf{v}_i^{(t-1)}| \mathbf{x}_i \tag{8}$$

Un seuil est utilisé pour fixer les poids à zéro quand leur valeur est faible.

Contrôle du nombre de caractéristiques : comme indiqué dans (Weston *et al.*, 2003), les approches de type ℓ_2-AROM ou RW qui suppriment des caractéristiques pour approcher une régularisation parcimonieuse peuvent aller trop loin dans le processus de sélection en supprimant trop de variables. La capacité de généralisation du modèle peut alors être dégradée. Une solution pour remédier à ce problème est de fixer un seuil maximal de variables à conserver r, l'algorithme stoppant la sélection dès que la contrainte est atteinte. Nous avons retenu cette approche dans nos algorithmes.

Adaptation à l'apprentissage d'ordonnancement : à chaque itération t, nous résolvons le problème pondéré en norme ℓ_2 à l'aide d'un algorithme spécifique à l'apprentissage d'ordonnancement. À la fin de l'étape de sélection, un dernier apprentissage est réalisé avec le sous-ensemble de variables sélectionnées et l'algorithme non

parcimonieux en norme ℓ_2, afin de stabiliser les poids finaux de la fonction d'ordonnancement.

Nous appelons cette méthode générique RankRWFS, dont le détail est donné à l'algorithme 1. Dans une première étape, cette approche sélectionne les variables adéquates en approchant un problème de SVM parcimonieux par pondération d'un problème en norme ℓ_2. À chaque itération t, un vecteur de poids $\mathbf{w}$ est appris en utilisant la règle de pondération spécifique à la régularisation parcimonieuse considérée et l'algorithme d'apprentissage d'ordonnancement en norme ℓ_2. Les caractéristiques dont le poids est inférieur à un seuil fixé sont retirées du modèle. Le processus est répété jusqu'à ce que le nombre de variables restantes soit inférieur ou égal à la valeur souhaitée. Dans une seconde étape, le sous-ensemble de caractéristiques sélectionnées est utilisé par l'algorithme d'apprentissage en norme ℓ_2, afin d'apprendre les poids finaux. Nous étudions deux versions de cette approche, RankRWFS-ℓ_0 et RankRWFS-ℓ_1, qui considèrent respectivement les régularisations ℓ_0 et ℓ_1.

3. Cadre expérimental

3.1. Données, mesures d'évaluation et références

Dans un premier temps, nous évaluons nos algorithmes sur cinq jeux de données issus des collections internationales de référence LETOR : Ohsumed, MQ2008, HP2004, NP2004 et TD2004. Ces jeux de données ainsi que leur description (format, liste des caractéristiques) sont disponibles directement sur le site de Microsoft Research [2]. Ohsumed est issu du corpus du même nom, pour lequel un sous-ensemble de 106 requêtes a été extrait. MQ2008 a été construit à partir d'un sous-ensemble de 784 requêtes et documents associés issus de la tâche *Million Query Track* de TREC 2008. Enfin, HP2004, NP2004 et TD2004 ont été construits à partir des requêtes et documents issus des sous-tâches *Homepage Finding*, *Name Page Finding* et *Topic Distillation* de la tâche Web de TREC 2004. Des caractéristiques, correspondant soit à des caractéristiques de la requête, soit à des scores de similarité entre requête et document, ont été extraites pour chacun de ces cinq jeux de données.

Dans un second temps, nous considérons également un jeu de données à plus large échelle (nombre de requêtes, de paires requête-document et de caractéristiques) issus du moteur de recherche Nomao [3]. Nomao est un moteur de recherche et de recommandations de lieux d'intérêts (restaurant, hôtels, etc). Un document correspond donc à un lieu. Ce jeu de données a été construit à partir d'un échantillon aléatoire de 9 915 requêtes issues des traces de connexion du moteur entre juillet 2012 et janvier 2013. La pertinence d'un document vis-à-vis d'une requête a été calculée à partir d'un modèle basé sur les clics utilisateurs proposé dans ce cadre (Laporte *et al.*, 2013). Les caracté-

2. http://research.microsoft.com/en-us/um/beijing/projects/letor/
3. www.nomao.fr

Algorithme 1 Approche RankRWFS

Entrée : Jeu de données d'apprentissage $(\tilde{\mathbf{x}}^j, \mathbf{y}^j)_{j=1}^P$, sueil s et nombre maximal de variables r à conserver
Sortie : Vecteur de poids $\mathbf{w}$

```
Initialisation v = [1...1] ∈ R^m, t = 1
Tant que ||w||_0 > r Faire
  Pour chaque observation x̃^j Faire
    Pour i = 1 → m Faire
      z_i^j ← x̃_i^j v_i^(t)
    Fin Pour
  Fin Pour
  w^(t) ← solution de RankSVM-Primal({(z^j, y^j)}_{j=1}^m)
  Pour i = 1 → m Faire
    Si Norme == ℓ1 alors
      v_i^(t+1) ← sqrt(|w_i^(t) v_i^(t)|)  %%%% Mise à jour pour ℓ1 %%%%
    Sinon Si Norme == ℓ0 alors
      v_i^(t+1) ← |w_i^(t) v_i^(t)|  %%%% Mise à jour pour ℓ0 %%%%
    Fin Si
  Fin Pour
  Pour i = 1 → m Faire
    w_i ← w_i^(t) v_i^(t)
  Fin Pour
  t = t + 1
Fin Tant que
I ← {i, |w_i| ≥ s}
Pour j = 1 → P Faire
  Pour i ∈ I Faire
    u_i^j ← x_i^j
  Fin Pour
Fin Pour
w ← solution de RankSVM-Primal({(u^j, y^j)}_{j=1}^m)  %% Modèle final
Retourner w
```

ristiques calculées correspondent en partie à des similarités entre requête et document, issues de la littérature, d'autres sont propres à Nomao.

Les proportions de requêtes, de paires requête-document, de préférences et de caractéristiques dans chaque jeu de données sont présentés dans le tableau 1. Certaines caractéristiques ont une valeur nulle pour l'ensemble des documents et des requêtes. Pour chaque jeu de données, nous indiquons également le nombre de caractéristiques non nulles entre parenthèses. Chaque jeu de données est décomposé en échantillons d'apprentissage, validation et test. Une procédure de validation croisée à 5 partitions est réalisée (chaque algorithme est évalué sur 5 répétitions correspondant à des échantillons de test, apprentissage et validation différents). Dans le cas des jeux de données

LETOR, ce découpage en échantillons est fourni. Dans le cas du jeu de données Nomao, nous avons procédé nous-mêmes au partitionnement, selon le même protocole que pour les jeux de données LETOR (Liu, 2011).

Tableau 1. Description des jeux de données

Nom	Caractéristiques	Requêtes	Paires	Préferences
HP2004	64 *(64)*	75	74 409	80 306
NP2004	64 *(64)*	75	73 834	75 747
TD2004	64 *(64)*	75	74 146	1 079 810
Ohsumed	45 *(39)*	106	16 140	582 588
MQ2008	46 *(40)*	784	15 211	80 925
Nomao	101 *(101)*	9 915	124 195	770 520

Nous utilisons deux mesures usuelles en RI pour évaluer la qualité de l'ordonnancement : la moyenne des précisions moyennes (MAP) et le *Normalized Discounted Cumulative Gain* (NDCG). Nous mesurons la capacité des algorithmes à supprimer un nombre important de variables en calculant les ratios de parcimonie des algorithmes, *i.e.* le pourcentage de caractéristiques restantes dans le modèle final. Certaines caractéristiques des jeux de données Ohsumed et MQ2008 sont nulles pour l'ensemble des requêtes, elles ne sont pas prises en compte pour le calcul des ratios de parcimonie. Nous comparons également les méthodes selon les temps d'exécution.

Nous comparons nos trois algorithmes à trois approches de l'état de l'art : Liblinear-ℓ_1 (Fan *et al.*, 2008), FenchelRank (Lai, Pan, Liu *et al.*, 2013) et FSMRank (Lai, Pan, Yong, Yong, 2013). Liblinear-ℓ_1 est une implémentation des SVM en norme ℓ_1 pour la discrimination disponible en ligne dans le package Liblinear [4]. FenchelRank [5] et FSMRank [6] sont les algorithmes de sélection de variables pour l'apprentissage d'ordonnancement les plus performants. Ces méthodes considèrent des SVM parcimonieux, elles sont donc particulièrement adaptées pour effectuer une comparaison.

3.2. Protocole expérimental

Nous avons effectué trois études sur les jeux de données LETOR. Tout d'abord, nous avons analysé conjointement les valeurs de MAP (respectivement de NDCG@10) et les ratios de parcimonie obtenus pour l'ensemble des algorithmes. Ensuite, nous avons comparé conjointement les temps d'exécution et les ratios de parcimonie de nos méthodes et de l'état de l'art. Enfin, nous avons extrait un sous-ensemble de caractéristiques importantes pour l'ordonnancement à partir des modèles obtenus. Sur le jeu de données Nomao, nous présentons les résultats des deux premières études, l'étude sur les caractéristiques étant confidentielle.

4. http://www.csie.ntu.edu.tw/~cjlin/liblinear/
5. Code source disponible sur http://www.scholat.com/~hanjiang
6. Code source fourni sur demande par les auteurs

Dans nos expérimentations, nous avons considéré des seuils de 50 %, 30 % et 10 % de variables à conserver. Comme expliqué dans la section 2, les algorithmes stoppent les itérations quand le nombre de variables du modèle est inférieur ou égal au seuil spécifié. La valeur du paramètre C choisie est celle qui maximise la MAP sur l'échantillon de validation, pour chaque algorithme et jeu de données. Nous avons fixé le seuil de nullité des variables à 10^{-5}. Nous avons choisi RankSVM-Primal (Chapelle, Keerthi, 2010) comme solveur pour les SVM en norme ℓ_2. Les algorithmes que nous proposons sont implémentés en Matlab/Octave. Les expérimentations sur les jeux de données LETOR 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. Les expérimentations sur le jeu de données Nomao ont été réalisées sur des ressources internes à l'entreprise.

4. Analyse préliminaire des corrélations entre caractéristiques

Dans cette section, nous présentons les résultats d'une analyse préliminaire effectuée sur les jeux de données. L'objectif de cette étude est de déterminer si les jeux de données que nous considérons contiennent beaucoup de caractéristiques redondantes.

Une caractéristique est considérée comme redondante si elle peut être expliquée à l'aide d'autres caractéristiques présentes dans le jeux de données, par exemple s'il s'agit d'une combinaison linéaire des autres variables. Une trop grande redondance est généralement à éviter car d'une part, elle peut rendre certaines approches d'apprentissage instables et d'autre part, elle augmente la dimension des modèles sans apporter de caractéristiques porteuses d'information complémentaire. Dans ces cas, la sélection de variables présente un intérêt tout particulier puisqu'elle va permettre de sélectionner un sous-ensemble de caractéristiques les plus utiles et informatives possibles, tout en minimisant leur redondance.

Une approche usuelle pour évaluer la présence potentielle de redondance est d'étudier la corrélation entre les caractéristiques, par exemple via les coefficients de corrélation de Pearson (relation affine entre caractéristiques) ou de Spearman (relation monotone). Des méthodes d'analyse de la redondance peuvent également être appliquées (F. E. Harrell, 2001). Nous présentons ici l'analyse des coefficients de corrélation de Spearman, effectuée sur les préférences. L'analyse a été effectuée à l'aide du logiciel R. Les résultats obtenus avec les coefficients de Pearson et avec une technique d'analyse de la redondance (F. E. J. Harrell *et al.*, 2015) sont similaires et ne sont pas présentés ici.

Les figures 1 et 2 présentent les matrices des coefficients de corrélations de Spearman pour les jeux de données TD2004 et OHSUMED respectivement. Les figures obtenues sur les quatre autres jeux de données sont très similaires et ne sont donc pas présentées ici. Les points bleus indiquent une corrélation positive, tandis que les points rouges indiquent une corrélation négative. L'intensité du niveau de gris et la taille du point représente l'intensité de la liaison.

Nous constatons que les matrices des coefficients de corrélation sont globalement très « pleines ». Certaines caractéristiques sont corrélées à l'ensemble des autres caractéristiques, bien que l'intensité de la liaison soit variable. Nous observons pour l'ensemble des jeux de données des groupes de caractéristiques extrêmement corrélées, avec des coefficients généralement supérieur à 0,9. Les groupes sont également corrélés entre eux.

De façon générale, pour chaque jeu de données, chaque groupe de caractéristiques identifié correspond aux caractéristiques calculées sur la même partie du document. Par exemple sur OHSUMED, le premier groupe rassemble les caractéristiques calculées entre le titre du document et la requête, le deuxième entre le résumé et la requête et les derniers entre le document complet (titre+résumé) et la requête. Il n'est donc pas surprenant de trouver des corrélations relativement importantes entre les groupes de caractéristiques.

Le nombre élevé de corrélations fortes détectées indique une redondance importante dans les différents jeux de données, ce qui est confirmé par l'analyse des redondances effectuées sous R. L'utilisation d'approches de sélection de variables devrait permettre de supprimer un nombre important de caractéristiques non nécessaires et de simplifier les fonctions d'ordonnancement apprises.

5. Résultats

5.1. Expérimentations sur les jeux de données LETOR

Dans cette section, nous évaluons les trois algorithmes que nous proposons sur les collections internationales de références LETOR. Tout d'abord, nous nous intéressons conjointement à la qualité d'ordonnancement et aux ratios de parcimonie. Ensuite, nous analysons conjointement les temps d'exécution et les ratios de parcimonie. Enfin, nous commentons les variables sélectionnées pour extraire un sous-ensemble de caractéristiques importantes pour l'apprentissage d'ordonnancement.

5.1.1. Ratios de parcimonie et qualité d'ordonnancement

Les figures 3 et 4 présentent conjointement les valeurs de MAP et les ratios de parcimonie obtenus pour les algorithmes que nous proposons et l'état de l'art. Les résultats obtenus avec le NDCG étant similaires, nous ne les présentons pas ici. Le lecteur pourra se référer à (Laporte, 2013) pour l'ensemble des résultats.

Les algorithmes de pondération atteignent des ratios de parcimonie plus faibles que ceux imposés à l'algorithme. En pratique, plusieurs caractéristiques sont parfois retirées simultanément du modèle, ce qui explique ce comportement. Nous constatons que les valeurs de MAP restent stables pour tous les algorithmes et tous les seuils de sélection, à l'exception de RankRWFS-ℓ_0 sur TD2004 au seuil de 10 %. Cette dégradation reste très légère et marginale. Les algorithmes que nous proposons obtiennent ainsi des qualités d'ordonnancement comparables à l'état de l'art sur les jeux de données de référence.

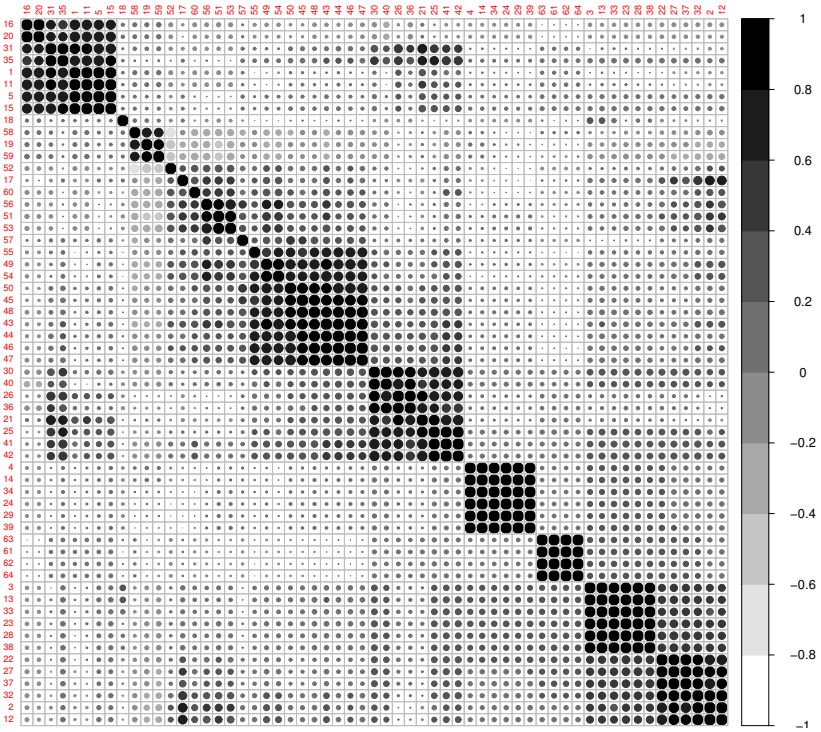

Figure 1. Corrélation de Spearman pour les caractéristiques du jeu de données TD2004

Plus intéressant, nous constatons que nos méthodes permettent d'obtenir des qualités d'ordonnancement équivalentes à l'état de l'art, tout en supprimant beaucoup de caractéristiques. Sur TD2004 et MQ2008, nos approches obtiennent des ratios de parcimonie plus faibles que l'état de l'art quel que soit le seuil choisi. Sur les autres jeux de données, les résultats sont équivalents aux approches de référence pour les seuils de 30 % et 50 %. Par contre, nos algorithmes suppriment beaucoup plus de caractéristiques que l'état de l'art lorsque nous considérons le seuil de 10 %. Les ratios de parcimonie sont globalement similaires pour les trois approches que nous proposons, l'algorithme RankRWFS-ℓ_0 est très légèrement plus parcimonieux que RankRWFS-ℓ_1 et Rankℓ_2-AROM à un seuil de sélection fixé sur la plupart des jeux de données.

Sur TD2004, nous observons que les algorithmes de pondération utilisent 6 à 10 fois moins de caractéristiques que les algorithmes de référence, pour apprendre des modèles de qualité d'ordonnancement équivalente. Le constat est similaire sur les autres jeux de données. Comparativement aux algorithmes de référence, les approches que nous proposons suppriment 3 à 7 fois plus de caractéristiques sur Ohsumed, 4 à 6 fois plus sur MQ2008 et 4 à 5 fois plus sur NP2004. HP2004 est le jeu de données pour lequel l'écart entre les méthodes que nous proposons et l'état de l'art est le

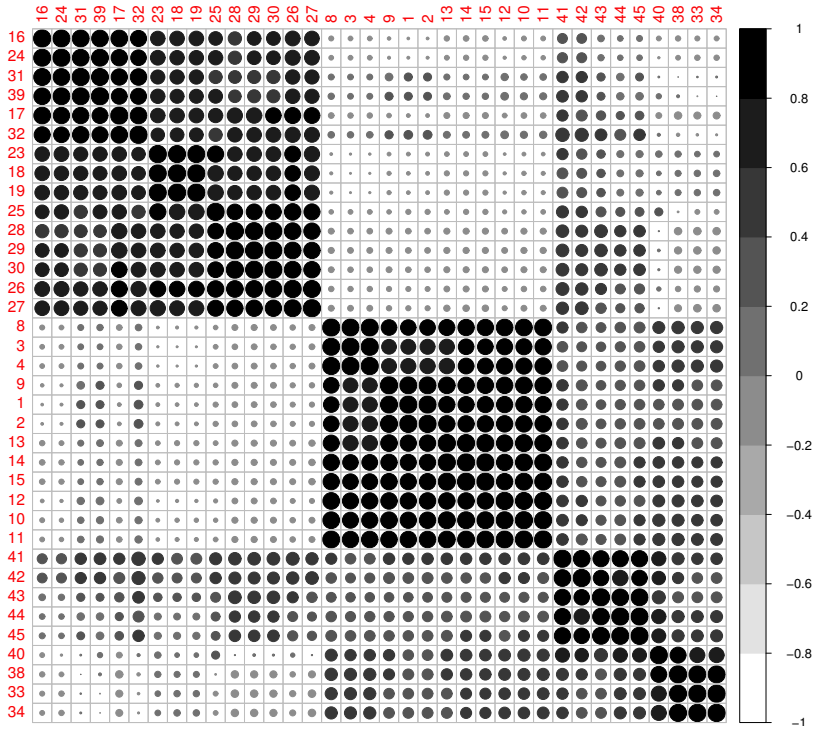

Figure 2. Corrélation de Spearman pour les caractéristiques du jeu de données OHSUMED

moins marqué, même s'il reste important : nos approches suppriment 2 à 3 fois plus de caractéristiques que les méthodes de référence.

Les algorithmes de pondération sont donc beaucoup plus efficaces que l'état de l'art pour effectuer la sélection de variables. En effet, elles obtiennent des qualités d'ordonnancement similaires, tout en supprimant jusqu'à 10 fois plus de caractéristiques. En ce sens, elles sont bien plus performantes que les approches de sélection de variables existantes en discrimination et en apprentissage d'ordonnancement.

5.1.2. Temps d'exécution et ratios de parcimonie

Les figures 5 et 6 présentent conjointement les temps d'exécution et les ratios de parcimonie obtenus pour les algorithmes que nous proposons et l'état de l'art. Notons que nous ne fournissons pas les valeurs pour FenchelRank, qui est le seul algorithme à ne pas être implémenté en Matlab.

Nous observons que les temps d'exécution des algorithmes de pondération que nous proposons sont globalement stables aux seuils de 50 % et 30 %, mais qu'ils augmentent lorsque le seuil de 10 % est considéré. Ce comportement pourrait traduire le

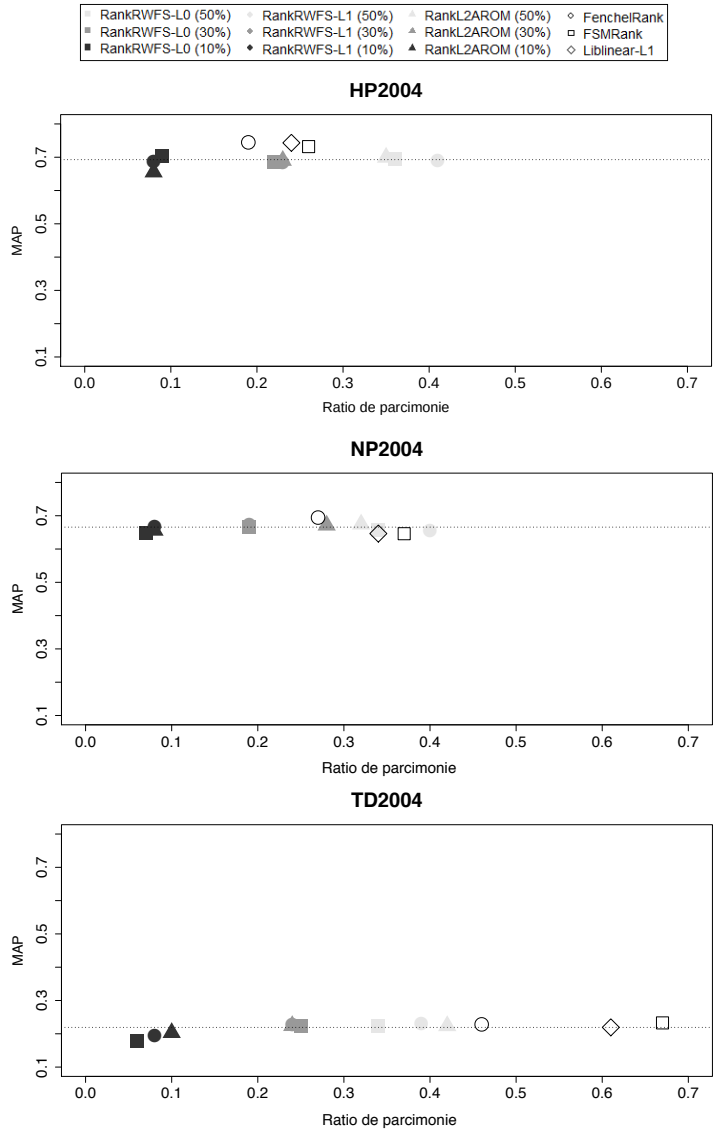

Figure 3. MAP vs. ratios de parcimonie pour chaque algorithme sur les trois jeux de données de référence issus de la collection GOV de LETOR 3.0. La ligne en pointillé représente la valeur de MAP moyenne pour l'ensemble des méthodes. Les algorithmes situés à gauche sont les plus parcimonieux. Les algorithmes que nous proposons sont équivalents en matière de MAP et meilleurs en matière de parcimonie

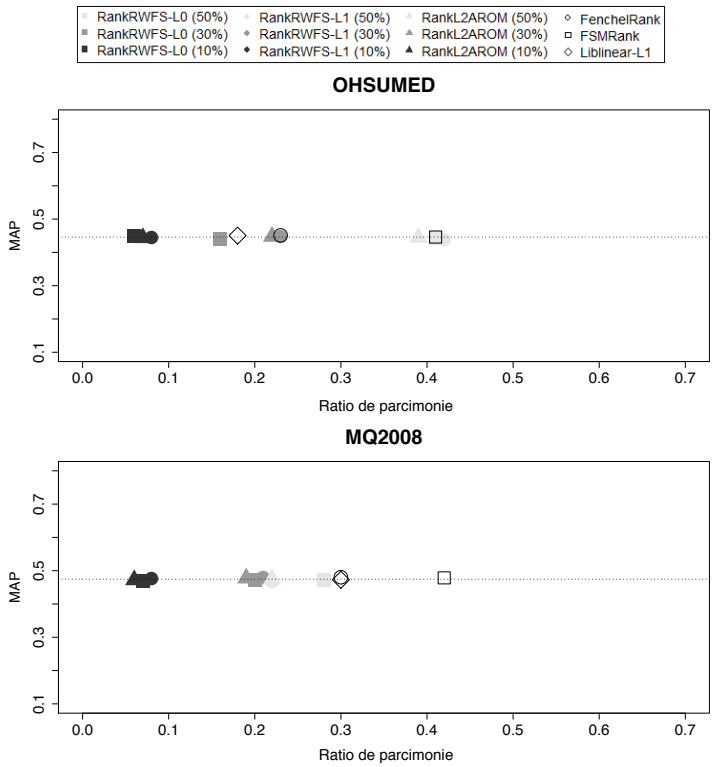

Figure 4. MAP vs. ratios de parcimonie pour chaque algorithme sur les jeux de données de référence OHSUMED et MQ2008. La ligne en pointillé représente la valeur de MAP moyenne pour l'ensemble des méthodes. Les algorithmes situés à gauche sont les plus parcimonieux. Les algorithmes que nous proposons sont équivalents en matière de MAP et meilleurs en matière de parcimonie

fait que la sélection devient plus difficile quand le nombre de caractéristiques restantes devient très faible.

Nous remarquons que les algorithmes de pondération que nous proposons sont plus performants que les méthodes de l'état de l'art. En effet, lorsque nous considérons les plus petits seuils de parcimonie, nos approches sont soit plus parcimonieuses, soit à la fois plus parcimonieuses et plus rapides.

Nous nous plaçons au seuil de 10 %. Sur le jeu de données TD2004, Rankℓ_2-AROM, RankRWFS-ℓ_1 et RankRWFS-ℓ_0 sont respectivement 2.5, 6 et 7 fois plus rapides que Liblinear-ℓ_1, tandis que RankRWFS-ℓ_1 et RankRWFS-ℓ_0 sont respectivement 2 et 3 fois plus rapides que FSMRank. Sur Ohsumed, les méthodes que nous proposons sont en moyenne 4 fois plus rapides que FSMRank et 2 fois plus rapides que Liblinear-ℓ_1. Sur HP2004, NP2004 et MQ2008, nos algorithmes sont plus lents que FSMRank, mais plus rapides que Liblinear-ℓ_1, tout en sélectionnant moins de ca-

ractéristiques que ces derniers. Plus précisément, sur MQ2008, nos algorithmes sont 2 à 4 fois plus rapides que Liblinear-ℓ_1. Ils sont 2 fois plus lents que FSMRank dans le pire des cas, mais sélectionnent environ 75 % de caractéristiques en moins. Sur HP2004, nos algorithmes sont tous 2 fois plus rapides que Liblinear-ℓ_1, 4 à 5 fois plus lents que FSMRank, tous en supprimant 2 à 3 fois plus de caractéristiques. Sur NP2004, RankRWFS-ℓ_0 et Rankℓ_2-AROM respectivement 2 et 3 fois plus rapides que Liblinear-ℓ_1. RankRWFS-ℓ_1 et Rankℓ_2-AROM sont 2 à 3 fois plus lents que FSMRank, mais suppriment 4 à 5 fois plus de caractéristiques. Parmi les algorithmes de pondération que nous proposons, RankRWFS-ℓ_0 est le plus rapide sur tous les jeux de données.

Les méthodes de sélection par pondération de la norme ℓ_2 que nous proposons sont donc plus performantes que les approches de l'état de l'art en apprentissage d'ordonnancement et en discrimination, puisqu'elles conservent une qualité d'ordonnancement similaire, tout en étant plus parcimonieuses et plus rapides. Par ailleurs, RankRWFS-ℓ_0 est globalement la plus rapide des trois approches proposées, ainsi que la plus parcimonieuse. Cela justifie d'une part, l'utilisation d'une approche de pondération de la norme ℓ_0 différente des approches de type ℓ_2-AROM et, d'autre part, l'utilisation de la norme ℓ_0.

5.1.3. Caractéristiques sélectionnées

Dans cette étude, nous nous sommes intéressés aux caractéristiques sélectionnées par les modèles, dans le but d'extraire un ensemble de critères importants pour l'ordonnancement. Nous considérons les modèles appris au seuil de 10 %.

Les algorithmes de pondération sélectionnent globalement les mêmes caractéristiques pour un même jeu de données. Ils se comportent donc de façon cohérente. Sur MQ2008, les algorithmes ont sélectionné les caractéristiques 23 et 39 pour l'ensemble des répétitions ainsi que les critères 32 et 37 pour la majorité des modèles. Toutes ces caractéristiques correspondent aux scores obtenus via des modèles de langue, connus en RI pour être très performants. Sur Ohsumed, les caractéristiques 3, 4 (basées sur la fréquence des termes), 11, 41 (score BM25 sur le titre et le document complet) et 43 (score modèle de langue) sont les plus fréquemment sélectionnées. Sur la collection Gov, les caractéristiques sélectionnées varient légèrement suivant le jeu de données. Sur HP2004, les algorithmes sélectionnent majoritairement les caractéristiques 23 (score BM25 du titre), 46 (hyperlink based feature propagation) et 52 (HostRank). Sur NP2004, les caractéristiques 23 et 46 sont également sélectionnées, ainsi que la caractéristique 22 (score BM25 de l'ancre). Par contre, la caractéristique 52 n'est généralement pas conservée. Enfin, sur TD2004, les algorithmes sélectionnent une combinaison des caractéristiques 22, 23, 52 et ponctuellement 46 suivant la répétition considérée. Toutes ces mesures sont connues pour être hautement informatives.

Les algorithmes proposés dans ces travaux sont donc capables d'apprendre des fonctions d'ordonnancement facilement interprétables, cohérentes et de bonne qualité.

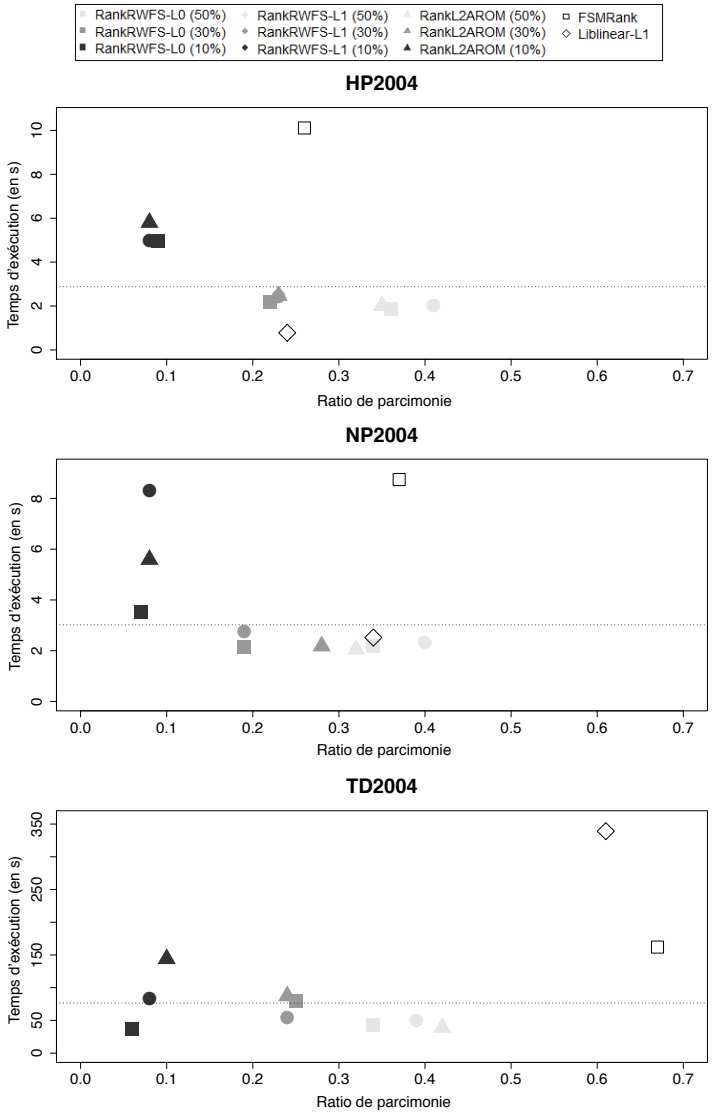

Figure 5. Temps d'exécution vs. ratios de parcimonie pour chaque algorithme sur les trois jeux de données de référence issus de la collection Gov de LETOR 3.0. La ligne pointillée représente le temps d'exécution moyen pour l'ensemble des méthodes. Les algorithmes situés à gauche et en bas sont les plus parcimonieux et les plus rapides. Les algorithmes que nous proposons sont meilleurs à la fois en matière de parcimonie et de rapidité d'exécution

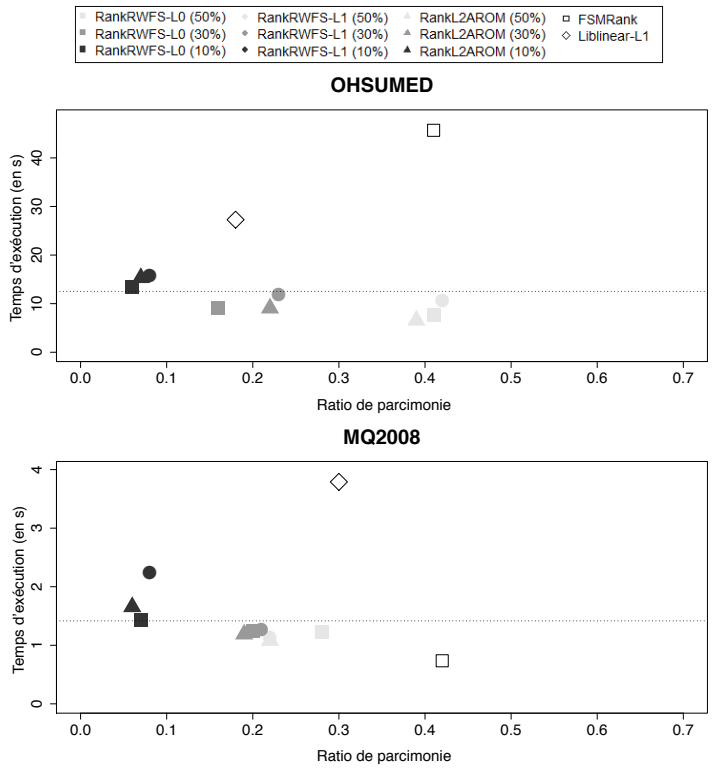

Figure 6. Temps d'exécution vs. ratios de parcimonie pour chaque algorithme sur les jeux de données OHSUMED et MQ2008. La ligne pointillée représente le temps d'exécution moyen pour l'ensemble des méthodes. Les algorithmes situés à gauche et en bas sont les plus parcimonieux et les plus rapides. Les algorithmes que nous proposons sont meilleurs en matière de parcimonie et de rapidité d'exécution

5.2. Expérimentations sur un jeu de données à plus large échelle

Dans cette section, nous évaluons nos algorithmes sur un jeu de données plus volumineux, dans un contexte d'application industrielle sur un moteur de recherche commercial. Nous nous intéressons à la qualité d'ordonnancement, aux ratios de parcimonie et aux temps d'exécution. Nous nous sommes concentrés ici sur l'évaluation et la comparaison des quatre algorithmes RankRWFS-ℓ_1, RankRWFSℓ_0, Rankℓ_2-AROM et FSMRank.

La figure 7 présente le lien entre MAP et ratio de parcimonie sur le jeu de données industriel à plus large échelle. La ligne en pointillé représente la valeur moyenne de MAP obtenue par les différentes méthodes.

Nous observons que les algorithmes présentent des qualités d'ordonnancement globalement similaires. L'algorithme le moins "bon" dégrade la MAP de 0,16 % com-

parativement à FSMRank, tandis que le "meilleur" l'améliore de 1 %. Les valeurs de MAP restent donc stables pour tous les algorithmes et tous les seuils de sélection.

De la même façon que sur les jeux de données LETOR, nous constatons que nos méthodes permettent de supprimer plus de caractéristiques que l'état de l'art, tout en conservant la même qualité d'ordonnancement. Ainsi, les algorithmes RankRWFS-ℓ_1, RankRWFS-ℓ_0 et Rankℓ_2-AROM obtiennent des ratios de parcimonie au seuil de 30 % similaire à l'algorithme de référence FSMRank ($0,26$, $0,2$ et $0,18$ contre $0,22$). Ils sont capables de supprimer de 3 à 4 fois plus de caractéristiques que FSMRank au seuil de 10 %. Ils sont ainsi beaucoup plus efficaces que l'algorithme de l'état de l'art pour apprendre des fonctions d'ordonnancement parcimonieuses.

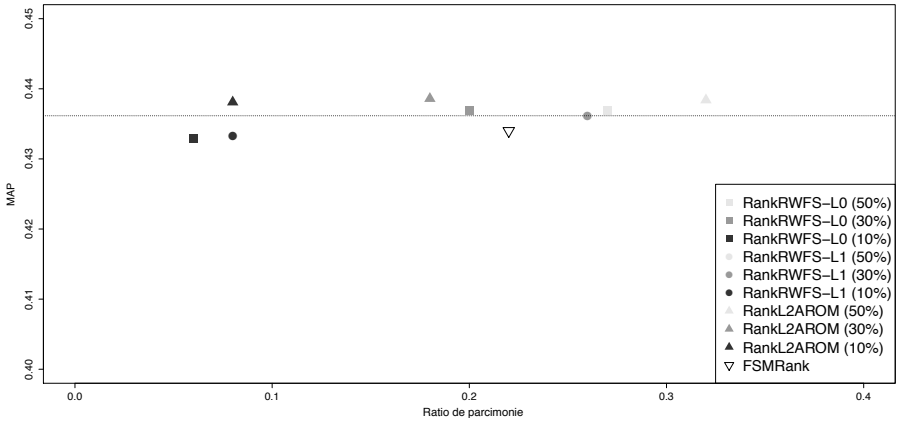

Figure 7. MAP vs. ratios de parcimonie pour chaque algorithme sur le jeu de données NOMAO. La ligne en pointillé représente la valeur de MAP moyenne pour l'ensemble des méthodes. Les algorithmes situés à gauche sont les plus parcimonieux. Les algorithmes que nous proposons sont équivalents en matière de MAP et meilleurs en matière de parcimonie

La figure 8 présente conjointement les temps d'exécution et les ratios de parcimonie obtenus pour les algorithmes que nous proposons et FSMRank, méthode de référence. La ligne en pointillé représente le temps d'exécution moyen, exprimé en secondes, pour les différentes méthodes. Les algorithmes les plus parcimonieux et/ou les plus rapides sont situés à gauche et en bas de la figure.

Nous observons que l'algorithme le plus rapide est l'algorithme de référence FSMRank, avec une durée d'exécution d'environ 20 secondes. Notons que 75 % du temps d'exécution de cet algorithme est à imputer au calcul de la matrice des corrélations entre caractéristiques qui est utilisée pour effectuer la sélection.

Parmi les algorithmes que nous proposons, l'algorithme RankRWFS-ℓ_1 est globalement le plus rapide, avec des temps d'exécution compris entre 20 et 35 secondes suivant le seuil de sélection considéré. Il semble également être le plus stable. En effet, les temps d'exécution aux seuils de 10 % et 30 % sont globalement similaires (30 et

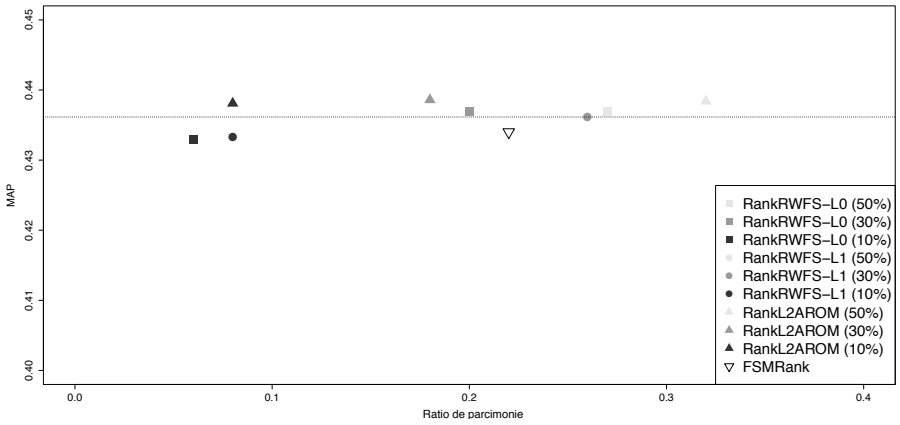

Figure 8. Temps d'exécution vs. ratios de parcimonie pour chaque algorithme sur le jeu de données Nomao. La ligne en pointillé représente le temps d'exécution moyen pour l'ensemble des méthodes. Les algorithmes situés à gauche et en bas sont les plus parcimonieux et les plus rapides. Les algorithmes que nous proposons sont plus lents que l'état de l'art, mais plus parcimonieux

35 secondes) tandis que ceux de RankRWFS-ℓ_0 et Rankℓ_2-AROM sont multipliés par deux entre le seuil à 30 % et le seuil à 10 %. Ce comportement avait déjà été observé sur les jeux de données LETOR.

Nous constatons que les algorithmes que nous proposons permettent de supprimer de 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 - nombre de caractéristiques supprimées serait ainsi plus favorable pour les algorithmes que nous proposons.

En conclusion, les algorithmes que nous proposons utilisent de 5 à 8 caractéristiques dans les fonctions d'ordonnancement, contre 22 en moyenne pour l'algorithme de référence, et 101 initialement, sans dégrader la qualité d'ordonnancement et dans des temps d'exécution qui restent très raisonnables.

6. Conclusion

Dans cet article, nous nous sommes intéressés à l'adaptation et à l'analyse des méthodes de pondération des SVM en norme ℓ_2 pour la sélection de variables via des SVM parcimonieux en norme ℓ_1 et ℓ_0. A notre connaissance, il s'agit des premiers travaux à proposer l'utilisation de la norme ℓ_0 et des méthodes de pondération de la norme ℓ_2 en sélection de variables pour l'apprentissage d'ordonnancement.

Nos expérimentations ont montré que :

1. Les méthodes de pondération de la norme ℓ_2 que nous proposons sont plus efficaces que l'état de l'art pour supprimer un grand nombre de caractéristiques des fonc-

tions d'ordonnancement. Elles conservent une qualité d'ordonnancement équivalente aux méthodes de référence, tout en utilisant de 3 à 10 fois moins de caractéristiques.

2. Les méthodes proposées obtiennent des qualités d'ordonnancement comparables à l'état de l'art, tout en étant de 2 à 7 fois plus rapides.

3. Les algorithmes que nous proposons permettent de sélectionner un sous-ensemble de caractéristiques très informatives pour chaque jeu de données. Ces sous-ensembles sont cohérents.

4. Les algorithmes que nous proposons permettent de supprimer plus de 90 % des caractéristiques sur l'ensemble des jeux de données considérés. Ce chiffre n'est pas nécessairement étonnant vu la grande redondance constatée dans les jeux de données et présentée dans cet article.

Les méthodes de pondération de la norme ℓ_2 que nous proposons constituent des approches performantes en sélection de variables pour l'apprentissage d'ordonnancement. Les performances obtenues concernant les ratios de parcimonie sont importantes, car la réduction du nombre de caractéristiques, en plus de simplifier les modèles appris, permet de gagner du temps lors de l'extraction des valeurs des caractéristiques. Les trois algorithmes Rank-ℓ_2-AROM, RankRWFS-ℓ_1 et RankRWFS-ℓ_0 présentent des performances globalement similaires, bien que RankRWFS-ℓ_0 semble plus rapide et légèrement plus parcimonieux. Par ailleurs, l'approche générique RankRWFS est plus flexible, puisqu'elle permet d'utiliser différentes régularisations parcimonieuses par une modification de la règle de mise à jour. Notons également que n'importe quel algorithme de résolution des SVM en norme ℓ_2 peut être incorporé au sein de ces méthodes. D'autres ré-écritures des normes (Chartrand, Yin, 2008 ; Zhang, Kingsbury, 2010) pourraient également être utilisées dans le cadre de cette approche. Leur apport fera l'objet de travaux futurs. Nous prévoyons également d'intégrer de nouvelles ré-écritures pour approcher la norme ℓ_0 par pondération de SVM en norme ℓ_1 et de comparer la performances de notre approche à des méthodes considérant d'autres régularisations parcimonieuses (Laporte, Flamary *et al.*, 2014).

Remerciements

Les auteurs remercient l'entreprise Nomao ainsi que la Région Midi-Pyrénées, qui ont contribué au financement de ces travaux (financement 10009018).

Bibliographie

Burges C., Shaked T., Renshaw E., Lazier A., Deeds M., Hamilton N. *et al.* (2005). Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on machine learning*, p. 89–96.

Cao Z., Qin T., Liu T.-Y., Tsai M.-F., Li H. (2007). Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on machine learning*, p. 129–136.

Chapelle O., Keerthi S. S. (2010). Efficient algorithms for ranking with svms. *Information Retrieval*, vol. 13, n° 3, p. 201–215.

Chartrand R., Yin W. (2008). Iteratively reweighted algorithms for compressive sensing. In *Icassp*, p. 3869-3872.

Dang V., Croft B. (2010). Feature selection for document ranking using best first search and coordinate ascent. In *Sigir workshop on feature generation and selection for information retrieval.*

Fan R.-E., Chang K.-W., Hsieh C.-J., Wang X.-R., Lin C.-J. (2008). Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, vol. 9, p. 1871–1874.

Freund Y., Iyer R., Schapire R. E., Singer Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, vol. 4, p. 933–969.

Geng X., Liu T.-Y., Qin T., Li H. (2007). Feature selection for ranking. In *Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval*, p. 407–414.

Harrell F. E. (2001). *Regression modeling strategies: With applications to linear models, logistic regression, and survival analysis*. Springer.

Harrell F. E. J., Dupont C., many others. (2015). Hmisc: Harrell miscellaneous Manuel de logiciel. Consulté sur http://CRAN.R-project.org/package=Hmisc (R package version 3.15-0)

Hastie T., Tibshirani R., Friedman J. H. (2003). *The Elements of Statistical Learning* (Corrected éd.). Springer. Hardcover.

Hua G., Zhang M., Liu Y., Ma S., Ru L. (2010). Hierarchical feature selection for ranking. In *Proceedings of the 19th international conference on world wide web*, p. 1113–1114.

Joachims T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, p. 133-142.

Kira K., Rendell L. A. (1992). A practical approach to feature selection. In *Proceedings of the ninth international workshop on machine learning*, p. 249–256.

Kohavi R., John G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, vol. 97, n° 1-2, p. 273–324.

Kujala J., Aho T., Elomaa T. (2009). A walk from 2-norm svm to 1-norm svm. In *Proceedings of the 2009 ninth ieee international conference on data mining*, p. 836–841.

Lai H., Pan Y., Liu C., Lin L., Wu J. (2013). Sparse learning-to-rank via an efficient primal-dual algorithm. *IEEE Transaction on Computers*, vol. 62, n° 6, p. 1221–1233.

Lai H., Pan Y., Yong T., Yong R. (2013). Fsmrank: A feature selection algorithme for learning-to-rank. *IEEE Transactions on Neural Networks and Learning Systems*.

Lai H., Tang Y., Luo H.-X., Pan Y. (2011). Greedy feature selection for ranking. In *Proceedings of the 15th international conference on computer supported cooperative work in design*, p. 42-46.

Laporte L. (2013). *La sélection de variables en apprentissage d'ordonnancement pour la recherche d'information : vers une approche contextuelle.* Thèse de doctorat, Université Paul Sabatier, Toulouse, France.

Laporte L., Déjean S., Mothe J. (2013). Multiple Clicks Model for Web Search of Multi-clickable Documents (short paper). In *International Conference on Enterprise Information Systems (ICEIS), Angers, 04/07/13-07/07/13*, p. (electronic medium). http://www.scitepress.org/.

Laporte L., Déjean S., Mothe J. (2014). Séparateurs à Vaste Marge pondérés en norme L2 pour la sélection de variables en apprentissage d'ordonnancement (regular paper). In *Conférence francophone en Recherche d'Information et Applications (CORIA), Nancy, 19/03/2014-21/03/2014*, p. 295–310.

Laporte L., Flamary R., Canu S., Déjean S., Mothe J. (2014). Non-convex Regularizations for Feature Selection in Ranking with Sparse SVM. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, n° 6, p. 1118–1130.

Liu T.-Y. (2011). *Learning to rank for information retrieval*. Springer.

Pahikkala T., Airola A., Naula P., Salakoski T. (2010). Greedy rankrls: a linear time algorithm for learning sparse ranking models. In *Sigir 2010 workshop on feature generation and selection for information retrieval*, p. 11-18.

Pan F., Converse T., Ahn D., Salvetti F., Donato G. (2009). Feature selection for ranking using boosted trees. In *Proceedings of the 18th ACM conference on information and knowledge management*, p. 2025–2028.

Sun Z., Qin T., Tao Q., Wang J. (2009). Robust sparse rank learning for non-smooth ranking measures. In *Proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval*, p. 259–266.

Weston J., Elisseeff A., Schölkopf B., Tipping M. (2003). Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, vol. 3, p. 1439-1461.

Xu J., Li H. (2007). Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval*, p. 391–398.

Yu H., Oh J., Han W.-S. (2009). Efficient feature weighting methods for ranking. In *Proceedings of the 18th ACM conference on information and knowledge management*, p. 1157–1166.

Zhang Y., Kingsbury N. (2010). Fast l0-based sparse signal recovery. In *Machine learning for signal processing (mlsp), 2010 ieee international workshop on*, p. 403-408.