Analyse de la robustesse des algorithmes de méta-recherche discriminante

Trang Vu — Patrick Gallinari

Laboratoire d'Informatique de Paris 6 (LIP 6), Université Pierre et Marie Curie
104 ave. du Pdt. Kennedy, 75016 Paris
{vu,gallinar}@poleia.lip6.fr

RÉSUMÉ. *Cet article examine la sensibilité de quatre moteurs de méta-recherche à différents facteurs et contextes d'utilisation. L'accent de l'étude est mis sur les méta-moteurs capables d'apprendre à partir d'exemples. L'apport original de notre travail consiste en une exploration systématique sur des corpus de grande taille des performances et du comportement des méthodes d'apprentissage pour la méta-recherche. D'abord, nous nous intéressons au choix de la représentation des attributs (les scores renvoyés par les moteurs de base). Nous examinons ensuite la performance des méta-moteurs sur différents types de requêtes de test. Nous présentons des expériences montrant l'influence des propriétés et du nombre des données d'apprentissage sur la performance finale sur les données de test. Enfin, nous donnons des résultats préliminaires sur la possibilité de sélectionner des requêtes par apprentissage actif. Toutes ces expériences démontrent que l'apprentissage supervisé de fonctions d'ordonnancement est particulièrement efficace pour la méta-recherche et offre des performances uniformément meilleures que celles obtenues par les moteurs individuels et les heuristiques de combinaison. Ces méthodes sont de plus robustes à des facteurs comme le codage des résultats des moteurs de base et la variabilité de la base d'apprentissage.*

ABSTRACT. *This paper studies the sensitivity of four metasearch engines under different situations. The focus of this analysis is on trainable metasearch engines. Our main contribution is a large scale systematic analysis of the performance and behavior of these methods on several corpora. Firstly, we analyze how the choice and normalization of the relevance score delivered by base search engines influence the performance of metasearch. We then study the effectiveness of the metasearch engines on individual queries. We also analyze the robustness of the metasearch engines with regard to the variability in the training and test corpora. We finally present a preliminary analysis of active learning for query selection. All of these experiments demonstrate that the learned search engines are quite effective since they are uniformly better than both heuristic metasearch techniques and indivual engines. They are also particularly robust to changes in the training data sets and to the encoding of base search engines scores.*

MOTS-CLÉS : *méta-recherche, BordaFuse, RankSVM, RankBoost, analyse de la robustesse.*

KEYWORDS: *metasearch, BordaFuse, RankSVM, RankBoost, robustness analysis.*

1. Introduction

La méta-recherche est une application particulière de combinaison des sources d'information, souvent hétérogènes, en RI. L'objectif est de fusionner les listes renvoyées par plusieurs moteurs de recherche en une liste unique afin d'obtenir un méta-système qui soit plus robuste et plus performant que les moteurs individuels. Dans le cas où il n'y a pas de vérité terrain, notamment dans les applications de science sociale, la performance d'un moteur de méta-recherche est définie comme le consensus des différents systèmes. Lorsqu'il y a une vérité terrain, i.e. un corpus des jugements de pertinence comme dans les évaluations en RI, nous espérons obtenir une liste fusionnée qui est plus proche de la vérité terrain. C'est dans ce cadre que nous situons notre travail. La performance d'un moteur sera mesurée par la précision moyenne non interpolée (AvgP).

Les premières familles de méta-moteurs ont fait appel à des méthodes de fusion heuristiques, par exemple [FOX 94]. Par la suite, plusieurs méthodes ont été proposées pour apprendre à fusionner des listes en employant des techniques issues de l'apprentissage statistique. Les méthodes dites "d'apprentissage de fonction d'ordonnancement" ont fait l'objet de nombreux travaux ces dernières années, par exemple [FRE 03, JOA 02, TAY 06, TSA 07], et apparaissent particulièrement prometteuses dans de nombreux domaines de la RI, qu'il s'agisse du texte, de l'image ou de la vidéo. Ces méthodes nécessitent la constitution de bases de données d'apprentissage pour entraîner les algorithmes d'ordonnancement. Le coût d'étiquetage étant généralement prohibitif, les bases d'apprentissage sont souvent constituées de façon heuristique, par exemple par des méthodes de "pooling" (*cf.* section 3.2). Une question largement ouverte est la robustesse des algorithmes de méta-recherche par apprentissage à la composition de cette base d'apprentissage.

L'efficacité des méthodes d'apprentissage pour l'ordonnancement en RI et plus généralement celle des méthodes de méta-recherche dépendent de nombreux facteurs. L'impact de ces facteurs n'a jamais fait à notre connaissance l'objet d'études systématiques. Plus précisément, nous allons examiner l'impact des facteurs suivants :

– quel codage utiliser pour les listes fournies par les différents moteurs ? Faut-il utiliser des scores réels, et dans ce cas, quel type de normalisation est le plus efficace, ou faut-il utiliser des valeurs relatives comme des rangs de classement ?

– au-delà de la mesure de performance moyenne des méta-moteurs, quel est leur comportement sur les différentes requêtes ?

– quelle est la robustesse des algorithmes d'ordonnancement par rapport à la variabilité entre base d'apprentissage et base d'évaluation, et quel est l'impact des requêtes difficiles ?

– est-il possible d'utiliser des stratégies d'apprentissage actif, où la base d'apprentissage est construite graduellement par interaction avec le juge, dans le but de minimiser le nombre de données à étiqueter ?

Nous étudions pour cela en détail le fonctionnement de quatre moteurs de méta-recherche : deux moteurs qui reposent sur des stratégies de combinaison heuristiques classiques, qui serviront de référence (ces méta-moteurs sont notés par la suite combSUM, combMNZ) et deux méta-moteurs reposant sur des algorithmes d'apprentissage statistique de fonctions d'ordonnancement (il s'agit des algorithmes RankSVM [JOA 02] et RankBoost [FRE 03]). L'objectif principal de notre travail est de tester la robustesse de ces deux dernières méthodes RankBoost et RankSVM dans différents contextes d'apprentissage et d'analyser la qualité de ces méthodes par rapport à des méthodes concurrentes. L'approche que nous proposons est empirique car il n'existe pas de cadre formel pour modéliser l'impact de différentes conditions de fonctionnement. Nous proposons dans cet article une étude systématique des différents facteurs mentionnés plus haut qui conditionnent à notre avis le fonctionnement de ces méta-moteurs. C'est à notre connaissance la première étude réalisée sur un grand nombre de moteurs de recherche de base (de l'ordre d'une centaine) et sur de grands corpus. Nos résultats s'appuient également sur une analyse de la validité statistique des écarts de performances entre les différentes stratégies.

La section 2 présente les notions de base de méta-recherche nécessaires pour cette étude. Ensuite, chaque section sera consacrée à l'impact d'un des facteurs décrits ci-dessus sur la stabilité des algorithmes de méta-recherche. Dans un premier temps, nous examinons l'impact des stratégies de normalisation des scores d'entrée (section 3). Nous testons ensuite la performance des méta-moteurs sur chaque requête de la base de test (section 4). Les deux sections suivantes concernent la robustesse des algorithmes d'apprentissage d'ordonnancement par rapport à la base d'apprentissage utilisée. La section 5 analyse l'impact de la variabilité entre base d'apprentissage et base d'évaluation. La section 6 montre l'évolution de la performance sur une même base de test en fonction des bases d'apprentissage et de leur taille (déterminée dans nos expériences par le nombre de requêtes). Nous décrivons ensuite une étude préliminaire sur l'apprentissage actif de fonctions d'ordonnancement. Dans la dernière section, nous faisons un bilan du manuscrit et discutons des questions ouvertes.

2. Notions de base de méta-recherche

Nous donnons dans cette section les notions de base de méta-recherche qui sont indispensables pour la présentation de l'article.

2.1. *Méthodes de méta-recherche*

Les méthodes de méta-recherche utilisées dans notre expérimentation sont toutes des méthodes de combinaison linéaire de fonctions de base qui reposent sur des techniques discriminantes. L'intérêt de ce dernier aspect est que l'on n'a pas besoin d'estimer la distribution des scores des moteurs de base. Elles suivent le principe proposé par Borda [BOR 81] : la fusion des résultats consiste à combiner linéairement les scores donnés par chacun des moteurs de base et à ordonner les documents en fonc-

tion de leur score global obtenu après cette combinaison. La méthode de base de Borda repose sur une heuristique simple, appelée *comptage de Borda* (*Borda Count*) et qui consiste simplement à additionner les scores normalisés des documents. En RI, cette stratégie de comptage est aussi appelée *combSUM* [FOX 94].

Plusieurs extensions de cette stratégie ont été proposées, une des plus répandues est la méthode dite *combMNZ*, proposée par Fox et Shaw [FOX 94]. Elle consiste à multiplier, pour chaque document la sortie de combSUM par le nombre de moteurs ayant renvoyé le document dans leur liste de documents pertinents. L'idée de cette pondération est simplement de favoriser les documents de plus hauts scores retournés par plusieurs SRI. Ces deux versions, combSUM et combMNZ, seront appelées BordaFuse ci-après.

Plus récemment, plusieurs travaux se sont attachés à employer des méthodes d'apprentissage automatique pour ajuster les coefficients des combinaisons linéaires en fonction d'un objectif représenté par une fonction de coût. Des familles de méthodes ont été proposées pour l'apprentissage hors ligne et l'apprentissage en ligne. Nous ne considérons ici que le premier cas et nous allons étudier deux méthodes qui figurent parmi les meilleures de cette famille. RankBoost [FRE 03] et RankSVM [JOA 02] sont deux algorithmes d'apprentissage qui optimisent une fonction d'ordonnancement. Des résultats théoriques ont été démontrés concernant leur capacité de généralisation [FRE 03, CLé 05]. Ces deux algorithmes optimisent deux majorations convexes de l'erreur d'ordonnancement qui sont une fonction exponentielle pour RankBoost, et une fonction dite de "hinge loss" pour RankSVM. Dans les deux cas, la fonction d'ordonnancement est une combinaison linéaire de fonctions de base générées à partir des attributs (ici ces attributs sont les scores des moteurs de recherche). On peut considérer ces deux algorithmes comme des variantes avec apprentissage du principe de Borda évoqué plus haut. RankSVM utilise directement comme attributs les scores des moteurs de recherche. RankBoost, lui, utilise des attributs relatifs qui sont les valeurs de position des documents dans la liste renvoyée par les moteurs.

2.2. *Formules de normalisation des attributs en méta-recherche*

Les scores calculés par les différents moteurs de recherche sont sur des échelles variables. Ils sont en général normalisés pour pouvoir être utilisés par les moteurs de méta-recherche. Plusieurs stratégies ont été proposées et seront évaluées dans notre étude. En ce qui concerne les scores réels, deux stratégies communes sont *std-score-norm* qui normalise les scores linéairement pour chaque moteur entre 0 et 1 et *sum-score-norm* pour lequel les scores sont d'abord translatés de façon à ce qu'il n'y ait pas de score négatif. Ils sont ensuite transformés de telle sorte que le score minimal ait la valeur 0 et que la somme des scores soit égale à 1. Pour certains moteurs qui travaillent sur les rangs, ces derniers sont parfois transformés en valeurs réelles avant d'être utilisés par le méta-moteur. La stratégie *std-rank-norm* normalise les rangs entre 0 et 1, elle affecte la valeur 0 aux documents non renvoyés. C'est cette stratégie que nous utiliserons quand nous considérons des rangs en entrée du méta-

moteur. Les scores fournissent une information plus riche que les rangs. Ils permettent par exemple de calculer la distribution de pertinence des documents, qui peut être utile pour l'ordonnancement. Cependant les rangs sont parfois préférés, car, contrairement aux scores, ils sont facilement comparables d'un moteur à l'autre.

3. Analyse de l'impact de la représentation des attributs sur les performances des méta-moteurs

Nous allons explorer l'impact des différentes stratégies de représentation et de normalisation des résultats des moteurs de recherche sur les quatre méta-moteurs de notre étude. Rappelons que l'algorithme RankBoost fonctionne sur les valeurs de position, il n'y a donc pas de résultat sur les codages du score dans ce cas là.

3.1. *Questions d'expérimentation*

Nous allons utiliser pour nos expériences, des bases TREC (*cf.* section 3.2) sur 3 années successives, de 1997 à 1999. Dans notre cas, ces bases sont constituées des réponses des moteurs ayant participé aux compétitions pour ces 3 années, sur l'ensemble des requêtes. Nous nous intéressons à cinq questions sur l'efficacité des moteurs de méta-recherche :

1) est-ce que les moteurs de méta-recherche excèdent significativement les moteurs de base ? (Q1).

2) est-ce que les algorithmes de méta-recherche avec apprentissage sont significativement supérieurs aux méthodes heuristiques du type BordaFuse ? (Q2).

3) est-ce que les méthodes "rank-based" sont significativement inférieures aux méthodes "'score-based"' ? (Q3).

4) y a-t-il une différence significative entre les stratégies "score-based" en fonction de la normalisation utilisée, i.e. std-score-norm vs. sum-score-norm ? (Q4).

5) y a-t-il une différence significative entre les stratégies de combinaison combMNZ et combSUM ? (Q5).

A notre connaissance, il n'existe pas d'étude systématique de ces questions. Les analyses réalisées dans la littérature de méta-recherche sont toujours effectuées à petite échelle pour une dizaine de moteurs de base comme [MON 01]. Nous proposons ici des expériences à large échelle, prenant en compte les listes proposées par une centaine de moteurs. Cela permet d'assurer une bien meilleure robustesse aux résultats et à leur significativité statistique. Nous proposons également dans ces expériences une comparaison systématique entre les méthodes heuristiques de combinaison de modèles et les méthodes avec apprentissage RankSVM et RankBoost, ainsi qu'une analyse du comportement de RankSVM selon la représentation des attributs.

3.2. *Conception des expérimentations*

Nous utilisons 3 bases de test de TREC : TREC 6[1], 7, 8. Le nombre de moteurs de base varie entre 78 et 129 pour ces différentes expériences. Nous comparons le méta-moteur entraîné par RankSVM ou par RankBoost avec les deux variantes de Borda-Fuse combSUM et combMNZ. Nous nous intéressons aussi aux comparaisons entre les méta-moteurs avec le meilleur moteur parmi les moteurs de base, pour savoir l'intérêt de la fusion par rapport à la sélection a posteriori du meilleur individu. Nous utilisons les trois stratégies de normalisation évoquées précédemment : std-score-norm, sum-score-norm et std-rank-norm.

Concernant les données d'entraînement pour RankBoost et RankSVM, nous utilisons le protocole suivant. Pour chaque requête, nous retenons comme base d'apprentissage les 5 premiers documents renvoyés pour chaque SRI, i.e. nous prenons la base générée par la méthode de "pooling" à profondeur 5. Le nombre de documents dans la base d'apprentissage varie alors entre 115 et 144 selon la requête. Nous estimons la performance en test en utilisant la stratégie de validation croisée "leave-one-out" sur la base des requêtes. Nous utilisons l'algorithme RankSVM dans le package SVM$^{\text{light}}$ [JOA 02], et la version de RankBoost adaptée à la RI (avec la pondération entraînée pour les requêtes) décrite dans [USU 05].

					rank-based				*score*-based					
									standard			sum		
TREC	#	max	min	moy	MNZ	SUM	SVM	RB	MNZ	SUM	SVM	MNZ	SUM	SVM
6	78	,463	,001	,169	,328	,331	,492	**,544**	,342	,333	,539	,400	,396	**,545**
7	103	,370	,012	,199	,308	,311	,463	**,542**	,320	,321	,501	,349	,355	,501
8	129	,469	,003	,235	,325	,329	,550	**,627**	,350	,356	,568	,417	,431	,572

Tableau 1 – MAP des méta-moteurs : combMNZ ("MNZ"), combSUM ("SUM"), RankSVM ("SVM"), RankBoost ("RB") avec trois types de normalisation des données : std-rank-norm ("*rank*-based"), std-score-norm ("standard *score*-based") et sum-score-norm ("sum *score*-based"). Nous montrons aussi le nombre des moteurs de base ("#") et les statistiques de MAP sur les moteurs de base : la valeur du meilleur moteur de base ("max"), du moteur le plus faible ("min"), et la moyenne ("moy").

La performance d'un SRI sera mesurée par la précision moyenne non interpolée (AvgP) et la moyenne arithmétique par défaut (MAP pour "Mean Average Precision") sur l'ensemble des requêtes de test. Pour tester la différence statistique entre deux moteurs, nous utilisons deux tests statistiques classiques (de type "paired", "1-sided", avec 95% confiance) : le t-test et le test de Wilcoxon. Nous utilisons aussi 2 règles pragmatiques traditionnelles en RI : si la différence de performance moyenne (comme MAP) de deux moteurs est égale ou plus petite que $0,05$ alors les performances sont

1. Nous avons éliminé la liste appelée "nmsu1" dans ce corpus TREC-6 qui n'a aucune réponse pour 32 requêtes sur 50 au total.

considérées équivalentes. Deux types de différence sont généralement considérées, la différence absolue δ_a ou la différence relative δ_r. Un autre critère basé sur la différence relative a été proposé par Sanderson et Zobel [SAN 05] : si la différence relative de MAP est supérieure à 25%, elle est considérée comme significative.

3.3. *Résultats et observations*

Le Tableau 1 présente la valeur de performance MAP pour ces différentes configurations. Le Tableau 2 présente le résultat de différents tests statistiques de la performance de plusieurs paires de moteurs. Sur le tableau 1, les moteurs à base d'apprentis-

hypothèses	TREC-6				TREC-7				TREC-8			
	δ'_a	δ_r	t	W	δ'_a	δ_r	t	W	δ'_a	δ_r	t	W
SUMr > MNZr	0,3	0,9	∼	*	0,3	0,9	*	*	0,5	1,5	∼	*
SVMr > SUMr	**16,1**	**32,7**	*	*	**15,2**	**32,8**	*	*	**22,0**	**40,1**	*	*
SVMr > best	2,9	5,9	*	∼	**9,3**	20,1	*	*	**8,1**	14,7	*	*
best > SUMr	**13,2**	**28,5**	*	*	**5,9**	15,9	*	*	**13,9**	**29,7**	*	*
SUMstd > MNZstd	-0,9	-2,6	-*	-*	0,1	0,3	∼	∼	0,6	1,7	*	*
SVMstd > SUMstd	**20,6**	**38,2**	*	*	**18,0**	**35,9**	*	*	**21,1**	**37,3**	*	*
SVMstd > best	**7,6**	14,1	*	*	**13,1**	**26,2**	*	*	**9,8**	17,3	*	*
best > SUMstd	**13,0**	**28,1**	*	*	4,9	13,2	*	*	**11,3**	24,1	*	*
SUMsum > MNZsum	-0,4	-1,0	∼	-*	0,6	1,7	∼	∼	1,4	3,3	*	*
SVMsum > SUMsum	**14,9**	**27,3**	*	*	**14,6**	**29,1**	*	*	**14,2**	24,7	*	*
SVMsum > best	**8,2**	15,1	*	*	**13,1**	**26,2**	*	*	**10,3**	18,0	*	*
best > SUMsum	**6,7**	14,5	*	*	1,5	4,1	∼	∼	3,8	8,1	∼	∼
SUMsum > SUMstd	**6,3**	15,9	*	*	3,4	9,6	*	*	**7,5**	17,4	*	*
MNZsum > MNZstd	**5,8**	14,5	*	*	2,9	8,3	*	*	**6,7**	15,9	*	*
SVMsum > SVMstd	0,6	1,1	∼	∼	0,0	0,0	∼	∼	0,5	0,9	∼	∼
SUMsum > SUMr	**6,5**	16,4	*	*	4,4	12,4	*	*	**10,1**	23,5	*	*
MNZsum > MNZr	**7,2**	18,0	*	*	4,1	11,8	*	*	**9,2**	21,9	*	*
SVMsum > SVMr	**5,3**	9,7	*	*	3,8	7,6	*	*	2,2	3,9	*	*
RB > SVMsum	-0,1	-0,2	∼	∼	4,1	7,6	*	*	**5,5**	8,8	*	*
RB > SVMr	**5,2**	9,6	*	*	**7,9**	14,6	*	*	**7,7**	12,3	*	*
RB > MNZr	**21,6**	**39,7**	*	*	**23,4**	**43,17**	*	*	**30,2**	**48,2**	*	*
RB > SUMr	**21,3**	**39,2**	*	*	**23,1**	**42,62**	*	*	**29,7**	**47,5**	*	*
RB > best	**8,1**	14,9	*	*	**17,2**	**31,73**	*	*	**15,8**	**25,2**	*	*

Tableau 2 – Tests empiriques et statistiques pour mesurer la différence entre deux SRI. "*" symbolise une différence significative, "∼" une différence faible, "-*" qu'une hypothèse alternative est recommandée. Les chiffres sont en **gras** quand un critère heuristique ($\delta'_a = \delta_a.10^{-2}$ ou $\delta_r(\%)$) est satisfait. Les suffixes "r", "std", "sum" dans la première colonne signifient respectivement trois types de normalisation : std-rank-norm, std-score-norm et sum-score-norm. "best" est le moteur de base le meilleur ("max" dans le Tableau 1). t et W correspondent au t-test et au test de Wilcoxon.

sage présentent de meilleures performances que les méthodes heuristiques de fusion

de Borda. Ces résultats sont confirmés par les tests statistiques. Nous observons une corrélation forte entre les deux tests statistiques et le critère du seuil de la différence absolue $\delta_a \geq 0,05$. Le critère $\delta_r \geq 25\%$ est trop restrictif et ne coïncide pas avec les autres tests. Nous l'écartons de l'analyse et discutons les hypothèses testées sur la base des deux tests statistiques et sur le critère absolu $\delta_a \geq 0,05$. Concernant les questions posées en section 3.1, nous pouvons énoncer les réponses suivantes :

1) Q1 : méta-recherche vs. moteur de base. Les stratégies combSUM et comMNZ agrègent mal les SRI hétérogènes, la performance des méta-moteurs qui utilisent ces stratégies est toujours inférieure à celle du meilleur moteur. RankBoost et RankSVM sont significativement meilleurs que tous les SRI individuels. Ces résultats démontrent ensemble l'importance du choix de la stratégie de fusion par rapport à la sélection posterieure du meilleur moteur de base.

2) Q2 : méthodes à apprentissage vs. BordaFuse (combSUM, combMNZ). Pour des stratégies de normalisation identiques, RankBoost et RankSVM excèdent toujours les méthodes heuristiques combSUM et combMNZ. Ces résultats démontrent alors l'intérêt réel de l'apprentissage supervisé par rapport aux règles de fusion heuristiques.

3) Q3 : "rank-based" vs. "score-based". Pour BordaFuse, utiliser des valeurs de score mène en général à une amélioration par rapport aux valeurs de rang. Pour RankSVM, il n'y a pas de différence nette entre score et rang. RankBoost qui ne fonctionne que sur les valeurs de position est aussi bon que des variantes de RankSVM sur les valeurs de position ou de score.

4) Q4 : méthodes de combinaison combSUM vs. combMNZ. Il n'y a pas une différence suffisante pour privilégier combSUM ou combMNZ. Ces résultats concordent avec ceux de Montague et Aslam [MON 01] qui avaient été effectués sur une dizaine de moteurs de base.

5) Q5 : sum-score-norm vs. std-score-norm. Pour combSUM et combMNZ, sum-score-norm est meilleur que std-score-norm. Il n'y a pas de différence entre ces deux codages quant à RankSVM.

En résumé, pour les données d'apprentissage que nous avons utilisées, RankBoost et RankSVM représentent de bonnes solutions pour agréger un grand nombre de SRI hétérogènes. Ils améliorent significativement les résultats du meilleur moteur et de la meilleure combinaison heuristique dans chaque cas. De plus, RankSVM est très peu sensible à la stratégie de normalisation de données.

4. Performance par requête

Nous avons comparé la performance globale des moteurs de méta-recherche sur un ensemble de requêtes. Du point de vue de l'utilisateur, la performance des moteurs sur les requêtes individuelles est également importante. Dans cette section, nous examinons donc la performance des moteurs par apprentissage RankBoost et RankSVM sur les requêtes de la base de test. Nous comparons ces moteurs avec les meilleurs moteurs de base. Le protocole d'expérimentation dans cette section est le même que dans

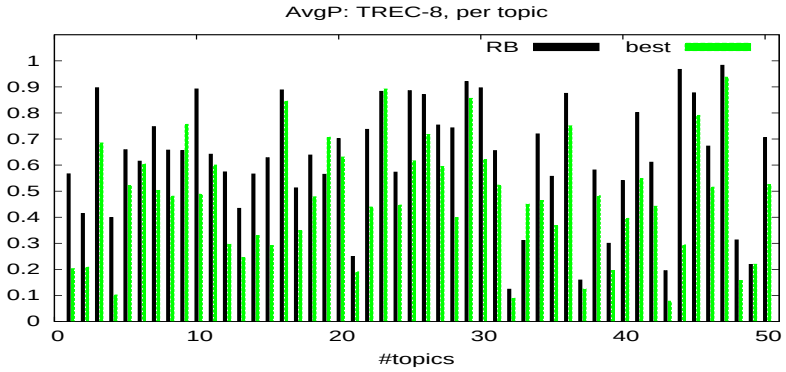

Figure 1 – La différence de précision moyenne sur chaque requête du corpus TREC8 de RankBoost par rapport au meilleur SRI de base du corpus. Les requêtes sont ordonnées selon l'ordre croissant de leurs numéros.

la section 3. Faute de place, nous ne montrons que la comparaison entre l'algorithme RankBoost et le meilleur système de TREC-8 qui est "READWARE2" (Figure 1). Notons que ce système est partiellement manuel, c'est-à-dire qu'il y a une intervention humaine dans certaines étapes du processus de recherche (l'utilisateur donne des retours de pertinence sur les documents dans le haut des listes de réponses intermédiaires). On voit d'après les graphiques de la Figure 1 que RankBoost fait mieux que le meilleur SRI de base pour 47/50 requêtes. Nous observons le même résultat avec RankSVM (non montré ici faute de place). Cela signifie que RankBoost et RankSVM ont non seulement une bonne performance moyenne mais aussi de bonnes performances individuelles sur la plupart des requêtes.

Il est par ailleurs connu que certaines requêtes sont particulièrement difficiles pour les moteurs de recherche. Par exemple, dans le corpus TREC 8, 17 des 50 requêtes sont considérées comme difficiles pour des SRI courants. Cette difficulté est détectée en se basant sur l'énoncé des requêtes elles-même ainsi que sur les résultats obtenus sur les systèmes participant à TREC. Nous donnons en Figure 2 la comparaison entre les algorithmes RankBoost et RankSVM (avec le codage *std-rank-norm*) sur ces requêtes difficiles. Il n'apparaît pas de différences significatives entre ces deux méta-moteurs pour ces requêtes.

5. Sensibilité des moteurs de méta-recherche à la base d'apprentissage

Nous examinons dans cette section l'impact des requêtes utilisées pour l'apprentissage sur la performance du modèle en test.

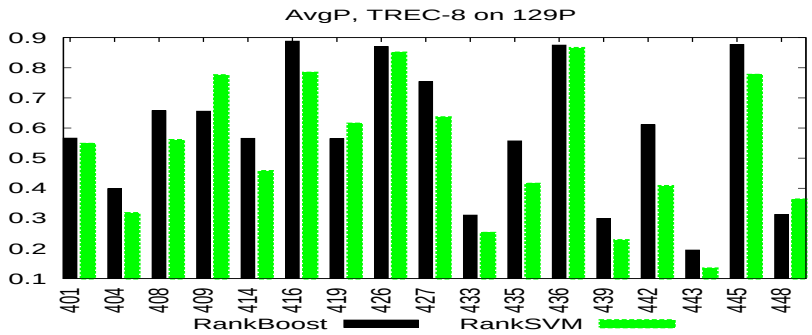

Figure 2 – Précision moyenne (AvgP) de RankBoost et de RankSVM sur 17 requêtes difficiles du corpus TREC8. Le numéro des requêtes est indiqué en abscisse.

5.1. *Introduction*

La robustesse des moteurs de recherche aux différents types de corpus et types de requêtes a fait l'objet de plusieurs études. Par exemple les résultats des campagnes de TREC révèlent que des SRI existants fonctionnent selon le principe "une seule formule couvre tous les cas" ("one size fits all"), cela signifie que la performance des meilleurs systèmes varie toujours largement d'une requête à l'autre. Ces études révèlent aussi que les performances sur les requêtes difficiles sont faibles et que la performance médiane est souvent nettement inférieure à la performance moyenne.

Dans le cadre du projet "Reliable Information Access" [HAR 04], 150 requêtes des corpus TREC 6, 7 et 8 ont été analysées pour les meilleurs SRI des campagnes de TREC. L'objectif de ce projet est l'analyse de la fiabilité des moteurs. Trois questions principales ont été étudiées : (1) quelles sont les mesures qui permettent de récompenser la fiabilité d'un SRI, (2) comment développer des méthodes pour prédire la performance ou la difficulté d'une requête et (3) comment concevoir des stratégies de RI appropriées pour des requêtes difficiles ? Il y a également eu récemment des expériences dont l'objectif était simplement d'examiner l'impact du nombre de requêtes d'apprentissage sur la performance en test [TAY 06, TSA 07]. Dans ces articles, les requêtes sont toujours supposées choisies de façon aléatoire. Nous analysons ici une autre forme de robustesse qui concerne la sensibilité des méta-moteurs de recherche qui apprennent à partir des données. Les questions que l'on se pose concernent les performances des méta-moteurs quand il y a une forte variabilité entre l'apprentissage et les conditions opérationnelles de test. Nous nous intéressons en particulier à analyser l'impact de la prise en compte de requêtes difficiles en apprentissage, cette analyse permettrait de savoir par exemple dans quelle mesure il est possible d'apprendre des requêtes difficiles. Ces questions rejoignent en partie celles sur la robustesse des moteurs qui n'utilisent pas l'apprentissage.

5.2. *Protocole expérimental*

Nous utilisons le corpus "TREC Robust 2004" (249 requêtes) [VOO 05], le seul corpus à notre connaissance dédié aux études empiriques de la robustesse des SRI. Pour cette série d'expérimentations, nous n'utilisons que les 10 meilleurs SRI (selon leur valeur MAP sur le corpus entier) parmi 110 SRI. Nous utilisons les 10 premiers documents de chacun de ces 10 SRI pour constituer une base d'apprentissage, et les valeurs de score renvoyées par ces 10 SRI comme attributs pour la méta-recherche. Ces scores sont normalisés par la méthode sum-score-norm.

Nous utilisons la précision moyenne AvgP pour mesurer l'efficacité d'un SRI sur une requête. Dans cette série d'expérimentation, nous utilisons le moyennage arithmétique (MAP) ainsi que le moyennage géométrique (gMAP, c'est-à-dire le produit et pas la somme des valeurs données) pour la performance moyennée sur un ensemble de requêtes de test. Ce dernier est recommandé par la campagne "TREC Robust" pour mieux récompenser la robustesse d'un SRI sur des requêtes de différents niveaux de difficulté.

Nous avons aussi testé 2 algorithmes d'ordonnancement supervisé : RankSVM et RankBoost. Dans nos expérimentations, RankBoost est légèrement meilleur que RankSVM mais l'écart n'est pas statistiquement significatif. Nous ne montrerons que les résultats de RankSVM dans la suite.

Pour évaluer l'impact des requêtes d'apprentissage sur la performance du modèle en test, nous avons divisé l'ensemble des requêtes du corpus Robust04 en sous-ensembles de différentes tailles[2] : 200 requêtes ("200old" = "150tr678" $\cup$ "50rb03"), 150 requêtes ("150tr678" pour les requêtes des corpus TREC 6, 7 et 8), 50 requêtes ("50hard" pour 50 requêtes difficiles extraites de "150tr678", "50rb03" pour 50 requêtes créées dans la campagne "TREC Robust 2003") et 49 requêtes ("49rb04" pour 49 requêtes créées dans la campagne "TREC Robust 2004"). Le détail de ces sous-ensembles se trouve dans [VOO 05]. La relation entre ces sous-ensembles est la suivante : $200old \supset 150tr678 \supset 50hard$, $200old \supset 50rb03$ et $200old \cap 49rb04 = \emptyset$. Nous analysons tout particulièrement le sous-ensemble $50hard$ qui contient 50 requêtes que les SRI des campagnes de TREC trouvent difficiles.

5.3. *Résultats*

Le Tableau 3 présente les valeurs gMAP et MAP de différents couples d'apprentissage et de test. La procédure est la suivante : on entraîne le méta-moteur RankSVM sur un sous-ensemble de requêtes ("train set" dans le tableau 3) et on le teste sur un autre sous-ensemble ("test set" dans le tableau 3). On peut voir que les performances des moteurs sur les bases de test 50rb04 et 49rb04 sont assez similaires (gMAP est approximativement 0.3) quelles que soient les bases d'apprentissage utilisées. La performance sur la base de test 50hard est beaucoup moins bonne que sur les bases de test

2. Les notations utilisées sont celles du corpus.

50rb03 et 49rb04 (gMAP ≈ 0.12 vs. 0.3, MAP ≈ 0.2 vs. 0.4). Nous notons aussi que la performance d'apprentissage de l'ensemble 50hard est légèrement inférieure aux performances obtenues en test quand ce moteur est entraîné avec une des deux bases 50rb03 et 49rb04. Ce phénomène n'est pas observé dans les autres expériences.

	test set (gMAP)			test set (MAP)		
train set	50*hard*	50rb03	49rb04	50*hard*	50rb03	49rb04
200old	-	-	0,339	-	-	0,428
150tr678	-	0,387	0,338	-	0,448	0,427
50*hard*	0,125	0,361	0,319	0,181	0,435	0,417
50rb03	0,128	0,391	0,314	0,190	0,448	0,415
49rb04	0,135	0,364	0,349	0,187	0,433	0,425
combSUM	0,145	**0,394**	**0,347**	0,198	**0,458**	**0,431**
combMNZ	**0,146**	0,392	0,342	**0,199**	0,456	0,428
best	0,144	0,376	0,336	0,195	0,440	0,423

Tableau 3 – Performance gMAP (gauche) et MAP (droite) sur différents couples (base d'apprentissage, base de test). Les lignes de "200old" à "49rb04" donnent les performances du méta-moteur RankSVM, les deux lignes suivantes combSUM et combMNZ donnent les performances des techniques de combinaison heuristiques de Borda, et la ligne "best" la performance du meilleur moteur individuel sur ces données. Les chiffres en **gras** donnent les meilleures performances. Les performances d'apprentissage sont soulignées.

Nous avons utilisé des règles pragmatiques ($\delta_a \geq 0,05$ ou/et $\delta_r \geq 5\%$) ainsi que des outils statistiques (intervalle de confiance par Bootstrap et tests de paires) pour tester si les différences de performance des colonnes MAP et gMAP du Tableau 3 sont statistiquement significatives. Les tests montrent que les différences de performance avec les différentes bases d'apprentissage ne sont pas statistiquement significatives. De même les différences de performance entre les méthodes BordaFuse (combSUM ou combMNZ) et RankSVM ne le sont pas non plus. Globalement RankSVM apparaît peu sensible aux conditions d'apprentissage. En particulier, les performances en test et apprentissage ne sont pas significativement différentes. Il est possible d'apprendre sur une base de requêtes et d'obtenir de bonnes performances sur des requêtes très différentes. Les performances obtenues sur les sous-ensembles de requêtes de cette nouvelle base sont similaires pour les méthodes à apprentissage et les méthodes de combinaison de Borda.

6. Sélection graduelle des requêtes d'apprentissage

6.1. *Stratégies de sélection*

Le dernier aspect de la robustesse des méta-moteurs que nous étudions concerne la possibilité d'apprendre à partir d'un faible nombre de données. L'acquisition des ju-

gements manuels de pertinence est coûteuse et minimiser le nombre de données d'apprentissage est un enjeu important. Une des techniques développées en apprentissage statistique est l'apprentissage actif qui permet de sélectionner incrémentalement les exemples d'apprentissage les plus utiles à un instant donné pour effectuer l'apprentissage. Nous examinons dans cette section quelles sont les possibilités d'apprentissage actif pour la méta-recherche. Ce problème revient à sélectionner à un instant donné quelle est la requête la plus intéressante pour effectuer l'apprentissage. Notons que ce problème est différent de celui de la sélection active des documents par requête qui a été examiné dans le contexte de l'ordonnancement par [VU 07]. Nous supposons que le pool des documents à juger par requête est fixé. Ce scénario est répandu dans le contexte de la RI internet par exemple où l'utilisateur ne s'intéresse qu'aux premiers résultats renvoyés par les moteurs de recherche.

Pour sélectionner une requête parmi un ensemble de requêtes qui n'ont pas encore été jugées, il faut associer à chacune de ces requêtes une valeur qui mesure l'intérêt de la requête pour le système d'apprentissage. Le problème est actuellement largement ouvert et n'a pas encore fait l'objet de travaux. Il y a 3 principales familles de critères, selon qu'ils sont définis à partir du score de chaque document, à partir de la différence de scores entre deux documents (des adaptations du critère utilisé par [YU 05, BRI 04] dans le cas de classification de multi-classe) ou à partir de la distance entre deux listes ordonnées des documents (voir [AMI 06] pour le détail de ce nouvel axe). Nous introduisons dans la suite des critères de sélection basés sur les scores des documents.

A partir du score de chaque document pour une requête donnée, nous calculons une mesure d'incertitude associée à la requête. La requête la plus incertaine sera choisie pour l'apprentissage du méta-moteur. L'expert donnera des jugements de pertinence sur les documents pour cette requête. La base d'apprentissage sera enrichie de ces jugements. Nous distinguons deux types de mesure d'incertitude :

– une mesure probabiliste, l'*entropie* d'information correspond à l'incertitude liée à cette information. Ce cadre exige que le score de chaque document soit calibré de façon à pouvoir être interprété comme une probabilité. Nous calculons alors l'entropie de la pertinence associée à chaque document. L'entropie d'une requête sera définie comme la somme des entropies de tous les documents jugés pertinents pour cette requête. Elle est notée SumEnt. La moyenne de cette valeur SumEnt calculée en divisant SumEnt par le nombre de documents pertinents pour la requête est notée AvgEnt.

– une mesure adaptée de la classification par machine à vecteurs supports (SVM). Les poids du méta-moteur définissent un hyperplan. Comme pour les SVM utilisées en classification, on peut définir une marge à l'aide des exemples d'apprentissage. On peut alors sélectionner la requête qui est plus proche de l'hyperplan. On note ce critère "Margin-based". Notons que cette règle est valide si nous avons déjà une quantité suffisante des données d'apprentissage pour tracer l'hyperplan de discrimination.

6.2. *Protocole expérimental*

Pour cette série d'expérimentations, nous avons besoin d'une base de données avec un grand nombre de requêtes. Nous avons utilisé le corpus PWSC [BEI 05] créé à partir des 10 premières réponses de 10 moteurs de base pour des requêtes posées par des utilisateurs au moteur commercial AOL. Ce corpus se compose de 896 requêtes jugées. Nous enlevons 66 requêtes atypiques pour lesquelles l'ensemble de documents jugés contient soit trop peu de documents pertinents, soit aucun document jugé comme non pertinent.

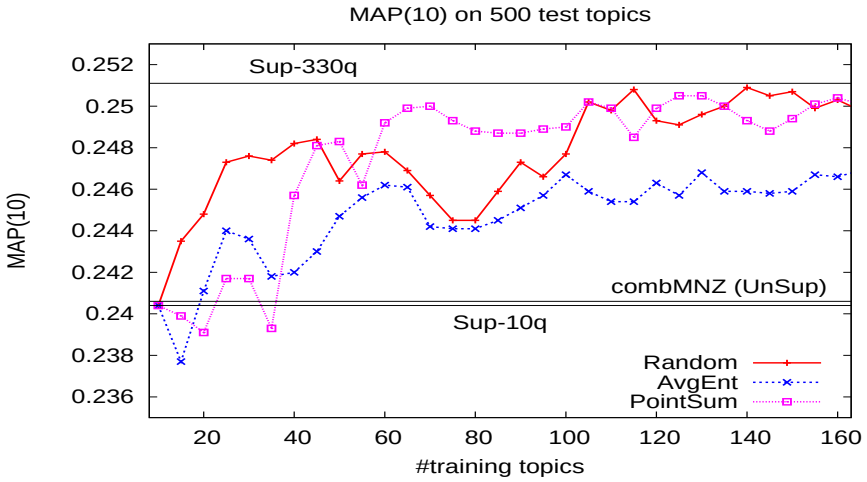

Figure 3 – La valeur de MAP sur la base de test. La base d'apprentissage est enrichie suivant différentes stratégies.

Nous divisons les données de la façon suivante : les 10 premières requêtes forment la base d'apprentissage initiale, les 320 suivantes seront utilisées pour l'apprentissage actif et les 500 dernières constituent la base de test. Pour chaque requête sélectionnée pour l'apprentissage, nous utilisons tous les documents jugés pour l'apprentissage, c'est-à-dire les 10 premiers documents renvoyés par 10 moteurs. Pour les attributs d'entrée, nous utilisons les scores fournis avec le corpus, qui sont les inverses de la position des documents dans la liste. La performance utilisée est la mesure de précision moyenne MAP.

Quant aux règles de sélection des requêtes d'apprentissage, nous présenterons les résultats des critères suivants dans la Figure 3 : une sélection au hasard ("Random"), un critère basé sur la valeur d'entropie ("AvgEnt"), et un critère de type "Margin-based" adapté de la classification par SVM ("PointSum"). Nous avons testé plusieurs autres règles de sélection de requête évoquées plus haut qui ne seront pas présentées ici par souci de lisibilité.

6.3. *Résultats empiriques*

La Figure 3 trace la performance (MAP) sur la base de test avec les stratégies incrémentales correspondant à différentes règles de sélection des requêtes. Selon ce résultat, la base d'apprentissage finale (330 requêtes) produit logiquement une meilleure performance que celle de la base d'apprentissage initiale (10 requêtes). Cet écart n'est cependant pas extrêmement élevé, mais il est significatif selon le t-test et le test de Wilcoxon. On voit globalement que les stratégies de sélection n'apportent pas d'information véritablement pertinente par rapport à une stratégie aléatoire. Les performances sont assez similaires pour toutes les stratégies. Ce comportement recoupe en partie celui observé en section 5.3 sur la robustesse des moteurs de recherche aux données d'apprentissage. Ces courbes montrent également la difficulté du problème de la sélection de requêtes pour les méta-moteurs.

7. Conclusion et discussion

L'objectif global de notre travail est d'analyser la robustesse et la qualité des algorithmes de méta-recherche. Nous avons présenté des expériences sur les données standard de TREC et sur un corpus de données réelles de recherche de pages web. Ces expériences ont permis une analyse systématique et à grande échelle du comportement des algorithmes de méta-recherche par apprentissage. Une analyse statistique de validité vient renforcer la portée des résultats obtenus. Nous avons tout d'abord comparé différentes stratégies de méta-recherche et analysé la sensibilité des algorithmes à différents types et normalisations des données d'entrée. Les principaux résultats sont : une nette supériorité des moteurs de méta-recherche à apprentissage sur les méta-moteurs heuristiques et sur le meilleur moteur de base, une forte robustesse des méthodes RankSVM au codage des données d'entrée et des résultats similaires pour le traitement des scores et des rangs. Ces résultats qui sont moyennés sur l'ensemble des requêtes se retrouvent sur les requêtes individuelles et l'on peut conclure que les méta-moteurs étudiés sont uniformément meilleurs que les approches heuristiques et les moteurs individuels. Nous avons également étudié la robustesse à la variabilité de la base d'apprentissage. Les méthodes étudiées sont peu sensibles à cette variabilité. Enfin, nous avons décrit une analyse prospective concernant la possibilité de sélectionner des requêtes par apprentissage actif.

Les principales extensions et limites de ce travail sont à notre avis relatives au cadre TREC utilisé dans notre protocole expérimental. La stratégie d'apprentissage utilisée repose sur le pooling qui fournit une proportion raisonnable de documents pertinents. Le cadre du web est très différent car la proportion de documents pertinents est en général bien plus faible. L'effet du déséquilibre des classes sur les performances des méthodes est bien connu en classification, le problème reste ouvert dans le cas des méthodes d'ordonnancement. De plus, les corpus sont de taille finalement assez petite, en terme de requêtes jugées, par rapport aux corpus privés utilisés dans [TAY 06, TSA 07]. Notre projet à court terme est d'analyser la robustesse des méta-moteurs entraînés sur de grands corpus de la campagne "Million Query" [ALL 07]

qui contiennent des milliers de requêtes et où les documents jugés peuvent être choisis adaptativement.

8. Bibliographie

[ALL 07] Allan J., Carterette B., Aslam J. A., Pavlu V., Dachev B., Kanoulas E., « Million Query Track 2007 Overview », *Proc. TREC-17*, 2007.

[AMI 06] Amini M., Usunier N., Laviolette F., Lacasse A., Gallinari P., « A Selective Sampling Strategy for Label Ranking », *Proc. ECML'06*, 2006, p. 18–29.

[BEI 05] Beitzel S. M., Jensen E. C., Frieder O., Chowdhury A., Pass G., « Surrogate scoring for improved metasearch precision », *Proc. SIGIR'05*, 2005, p. 583–584.

[BOR 81] Borda J.-C., « Mémoire sur les élection au scrutin », 1781.

[BRI 04] Brinker K., « Active Learning of Label Ranking Functions », *ICML'04*, 2004.

[CLé 05] Clémençon S., Lugosi G., Vayatis N., « Ranking and scoring using empirical risk minimization », *Proc. COLT 2005*, 2005.

[FOX 94] Fox E. A., Shaw J. A., « Combination of multiple searches », *Proc. TREC-2*, 1994, p. 243–249.

[FRE 03] Freund Y., Iyer R., Schapire R. E., Singer Y., « An efficient boosting algorithm for combining preferences », *JMLR*, vol. 4, 2003, p. 933–969.

[HAR 04] Harman D., Buckley C., « The NRRC reliable information access (RIA) workshop », *Proc. SIGIR'04*, 2004, p. 528–529.

[JOA 02] Joachims T., « Optimizing search engines using clickthrough data », *Proc. KDD'02*, 2002, p. 133–142.

[MON 01] Montague M., Aslam J. A., « Relevance score normalization for metasearch », *Proc. CIKM'01*, 2001, p. 427–433.

[SAN 05] Sanderson M., Zobel J., « Information Retrieval System Evaluation : Effort, Sensitivity, and Reliability », *Proc. SIGIR'05*, 2005, p. 162–169.

[TAY 06] Taylor M., Zaragoza H., Craswell N., Robertson S., Burges C., « Optimisation methods for ranking functions with multiple parameters », *Proc. CIKM'06*, 2006, p. 585–593.

[TSA 07] Tsai M.-F., Liu T.-Y., Qin T., Chen H.-H., Ma W.-Y., « FRank : A Ranking Method with Fidelity Loss », *Proc. SIGIR'07*, 2007.

[USU 05] Usunier N., Amini M., Gallinari P., « Combinaison de fonctions de préférence par Boosting pour la recherche de passages dans les systèmes de question/réponse », *Extraction et Gestion de Connaissances (EGC'05)*, 2005.

[VOO 05] Voorhees E. M., « The TREC Robust retrieval track », *SIGIR Forum*, vol. 39, n° 1, 2005, p. 11-20.

[VU 07] Vu H.-T., Gallinari P., « Apprentissage Statistique pour la Constitution de Corpus d'évaluation », *Revue I3 (Information-Interaction-Intelligence)*, vol. 7, n° 1, 2007.

[YU 05] Yu H., « SVM selective sampling for ranking with application to data retrieval », *Proc. KDD'05*, 2005, p. 354–363.