ORDONNANCEMENT DES RÉSULTATS SUR LES MOTEURS DE RECHERCHE : PRINCIPES, LIMITES ET APPLICATIONS AU GÉORÉFÉRENCEMENT

Léa LAPORTE (*,**)
laporte@irit.fr, lea@nomao.com

(*)Institut de Recherche en Informatique de Toulouse , 118 Route de Narbonne, 31062 Toulouse Cedex 9, France,
(**)Nomao, 1 Avenue Jean Rieux, 31500 Toulouse, France.

Mots clefs :
Recherche d'information, classement des résultats de recherche, apprentissage, ordonnancement adapté aux requêtes, modèles de pertinence
Keywords :
Information Retrieval, learning to rank, machine learning, query-dependent ranking, relevance models
Palabras clave :
Recuperación de la Información, aprendizaje para clasificar, aprendizaje, adaptación a las consultas, modelo de relevancia

Résumé

Les moteurs de recherche géoréférencés utilisent des algorithmes d'ordonnancement complexes, prenant en compte le contexte d'utilisation, l'*e-reputation* et les réseaux sociaux, afin de classer pertinemment les lieux vis-à-vis d'une requête utilisateur. Parallèlement, contrôler sa visibilité, être correctement référencé, comprendre les critères de sélection des utilisateurs et les critères d'ordonnancement des moteurs sont des points cruciaux pour les entreprises présentes sur ces moteurs. Nous présentons dans cet article le principe général de l'optimisation de l'ordonnancement sur les moteurs de recherche, ainsi que les différentes approches et algorithmes développés au cours de la dernière décennie. Nous montrons que ces algorithmes sont limités, car trop généraux. Nous proposons des pistes d'amélioration sur la façon d'évaluer la pertinence sur certains moteurs spécialisés. Nous présentons également des pistes d'adaptation aux différents types de requêtes, en déterminant différents critères pour l'ordonnancement.

1 Introduction

Au cours des dernières années, un grand nombre de moteurs de recherche d'information géoréférencés ont émergé sur Internet. Ces moteurs proposent aux internautes de rechercher des lieux qui sont classés suivant un grand nombre de critères : l'adéquation avec la requête, la proximité du lieu à la position de l'utilisateur, l'adéquation entre le lieu et les goûts de l'utilisateur ou de son réseau social ou encore l'*e-réputation* du lieu. L'*e-réputation* correspond à la réputation d'un établissement sur Internet, au travers des commentaires laissé par les internautes sur des forums, des réseaux sociaux ou d'autres sites de partage. Pour les utilisateurs, ces moteurs peuvent représentés des outils attractifs. Les résultats correspondent non seulement à leur besoin, mais ils ont aussi été jugés par l'ensemble des internautes et par leur réseau social, ce qui constitue un gage de confiance. Par ailleurs, la recherche personnalisée semble leur garantir des recommandations au plus près de leurs goûts et de leurs attentes. Pour les établissements, ces moteurs constituent une opportunité d'être plus visibles, d'augmenter ou de mieux cibler leur clientèle ou encore de contrôler leur e-réputation. Ainsi, le moteur de recherche géoréférencé Nomao propose aux propriétaires des lieux des outils stratégiques d'aide au référencement et de suivi de leur réputation sur Internet. Plus généralement, la connaissance des mécanismes de référencement et surtout des critères et algorithmes de classement des résultats peuvent être des outils puissants pour la veille stratégique.
Les moteurs de recherche utilisent des algorithmes d'ordonnancement qui leur permettent de classer les résultats suivant leur pertinence à une requête. A l'aide de jeux de données constitués de paires requête-document pour lesquelles la pertinence est connue, les algorithmes apprennent une fonction dite d'ordonnancement permettant de prédire la pertinence et l'ordre des documents. Un grand nombre d'approches et d'algorithmes ont été développés dans ce but au cours de la dernière décennie. Si ces algorithmes sont performants, ils présentent néanmoins des limites. Notamment, ils n'utilisent qu'une seule fonction d'ordonnancement pour l'ensemble des requêtes. Ils considèrent ainsi que les critères de tris sont identiques quelque soit la requête et l'utilisateur. Par ailleurs, la plupart de ces algorithmes ont été développés dans le cadre de moteurs généralistes. Leur utilisation sur des moteurs spécialistes, comme par exemple les moteurs géoréférencés, peut ne pas être adaptée. Il est donc nécessaire de proposer de nouvelles méthodes permettant de mieux prendre en compte les spécificités des requêtes et des utilisateurs, utilisables sur des moteurs généralistes ou spécialistes.
Dans la première partie de cet article, nous introduisons brièvement la Recherche d'Information, puis nous présentons le principe de l'optimisation de l'ordonnancement des résultats sur les moteurs de recherche. Nous détaillons également les différentes approches proposées au cours de la dernière décennie que nous illustrons par les algorithmes de référence correspondants. Puis, dans une deuxième partie, nous présentons deux limites des approches existantes pour lesquelles nous proposons des améliorations.

2 Apprentissage d'ordonnancement en Recherche d'Information

La Recherche d'Information (RI) est le domaine de la recherche qui s'intéresse à « la représentation, à l'organisation, au stockage et à la sélection de l'information » [25]. Une des tâches centrales en RI est la restitution de documents pertinents vis-à-vis d'une requête au sein d'un corpus.
En RI, requêtes et documents sont généralement représentés sous forme de vecteurs des occurrences des termes présents dans la requête et le document respectivement. Des mesures représentant la similarité entre la requête et le document sont calculées à partir de ces vecteurs. Elles sont ensuite utilisées pour sélectionner les documents qui sont retournés par le système. Des fonctions d'ordonnancement permettent ensuite de déterminer l'ordre des résultats. Un système de recherche d'information (SRI) est alors évalué sur sa capacité à restituer l'ensemble des documents pertinents et à les classer de façon optimale. L'optimisation automatique des fonctions d'ordonnancement, donc du classement des résultats de recherche, est l'objectif du *learning to rank* en RI.

Nous présentons dans un premier temps les mesures de RI utilisée pour évaluer les SRI. Dans un second, nous introduisons le principe général de l'optimisation automatique des fonctions d'ordonnancement (*learning to rank*). Nous détaillons également les différentes approches utilisées dans les algorithmes de *learning to rank*.

2.1 Mesures d'évaluation en Recherche d'Information

La performance d'un SRI est évaluée sur sa capacité à sélectionner les documents pertinents, comparativement à d'autres systèmes. Deux critères sont étudiés : le *rappel* et la *précision*.
Le *rappel* traduit la capacité d'un SRI à restituer l'ensemble des documents pertinents. Il est défini de la façon suivante:

$$\text{rappel} = \frac{\text{Nombre de documents pertinents sélectionnés}}{\text{Nombre total de documents pertinents}}$$

La *précision* traduit la capacité d'un système à ne sélectionner que des documents pertinents. Elle est définie de la façon suivante :

$$\text{précision} = \frac{\text{Nombre de documents pertinents sélectionnés}}{\text{Nombre total de documents sélectionnés}}$$

La performance des systèmes est généralement évaluée à partir de la précision à la position k P@k, de la précision moyenne AP et de la moyenne de la précision moyenne sur l'ensemble des requêtes MAP, définies ci-dessous.

$$\text{P@k} = \frac{\text{Nombre de documents pertinents sélectionnés jusqu'au rang k}}{\text{Nombre total de documents pertinents jusqu'au rang k}}$$

$$\text{AP} = \frac{\sum_{k=1}^{n} (\text{P@k.rel(k)})}{\text{Nombre total de documents pertinents}} \text{ avec rel(k)} = 1 \text{ si le document au rang est pertinent, 0 sinon et n le nombre total de documents}$$

$$\text{MAP} = \frac{\sum_{i=1}^{Q} AP^{(i)}}{\text{Q}} \text{ où Q est le nombre total de requêtes}$$

Une autre mesure fréquemment utilisée en RI est le NDCG (Normalized Discounted Cumulative Gain). Elle permet d'évaluer la capacité des SRI à renvoyant les documents pertinents en haut de la liste de résultats. Elle est définie de la façon suivante:

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}} \text{ où } \text{DCG@k} = \sum_{i=1}^{k} \frac{2^{\text{rel(k)}} - 1}{\log_2 i} \text{ et IDCG@k est la valeur maximale possible de DCG@k}$$

Certains algorithmes optimisent directement ces mesures pour apprendre les fonctions d'ordonnancement.

2.2 Principe et approches en apprentissage d'ordonnancement

Dans le domaine de l'apprentissage d'ordonnancement, chaque couple requête-document (q_i, d_j) est représenté par un vecteur de variables $x_{i,j} = (x_{i,j}^{(1)}, \cdots, x_{i,j}^{(n)})$ [1] qui traduisent la similarité entre la requête et le document (par exemple le nombre de termes qu'ils ont en commun), ainsi que certaines caractéristiques propres à la requête (nombre de termes, …) ou propres au document (confiance accordée à la source, …). Des jugements de pertinence sont associés à ces couples. Il peut s'agir soit de scores, de classes de pertinence (très pertinent, peu pertinent, non pertinent par exemple) ou de relation d'ordre (document 1 plus pertinent que le document 2) déterminés par des experts humains, soit de scores, de relation d'ordre ou de probabilités de pertinence estimées à partir des clics des utilisateurs sur les documents. L'objectif des algorithmes de *learning to rank* est de prédire correctement ces jugements connaissant les valeurs des variables $x_{i,j}$. Le processus d'apprentissage d'ordonnancement se décompose en deux phases : une phase d'apprentissage et une phase de test. Dans la phase d'apprentissage, ces jeux de données sont utilisés par les algorithmes pour apprendre automatiquement les fonctions d'ordonnancement qui servent de modèles pour la prédiction des jugements de pertinence. Dans la phase de test, ces fonctions sont ensuite utilisées pour ordonner les documents restitués par le SRI lorsque de nouvelles requêtes ont été soumises. Ce principe est illustré à la figure 1.

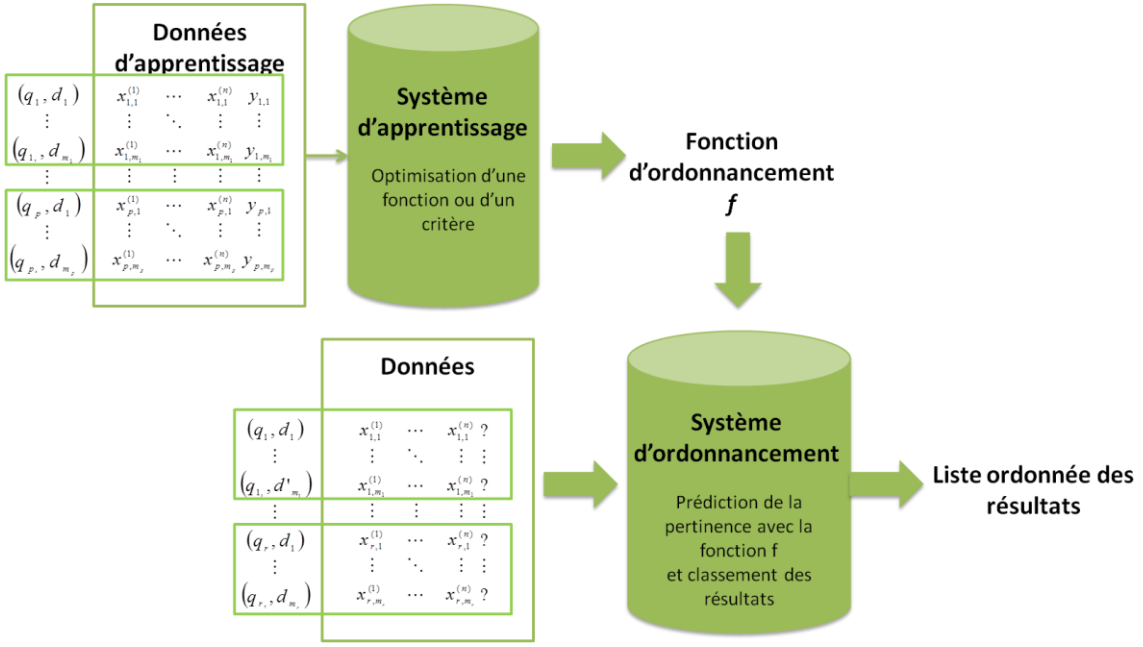

Figure 1 : Les différentes étapes du processus d'ordonnancement [19]

[1] Dans la suite de l'article, nous considérerons la requête fixée et nous intéresserons au document x_i pour la requête considérée afin d'alléger les notations.

Au cours de la dernière décennie, de nombreux algorithmes ont été proposés pour optimiser l'ordonnancement des résultats de recherche. Ils sont généralement répartis en trois grandes catégories : *par point (pointwise)*, *par paire (pairwise)* et par liste (*listwise)*. Ces approches diffèrent sur trois points: leur façon de considérer les données en entrée du système d'apprentissage, le type de variable ou jugement de pertinence à prédire et la modélisation mathématique du problème d'apprentissage.

Dans l'approche par point (*pointwise)*, chaque document x_i est considéré séparément en entrée du système d'apprentissage. Le jugement de pertinence peut être un score entier ou réel, une classe de pertinence non ordonnée (non pertinent, pertinent) ou une classe de pertinence ordonnée (pertinence de niveau 1 < pertinence de niveau 2 <...). Le jugement de pertinence est ici une variable à prédire, dont la valeur permet d'ordonner les documents.
Lorsque le jugement de pertinence est un score entier ou réel, le problème d'apprentissage est généralement considéré comme un problème de *régression linéaire*. La relation qui lie la variable quantitative à expliquer aux variables explicatives est supposée linéaire. Cossock et Zhang [5] ont ainsi proposé un algorithme minimisant une variante de l'erreur de moindres carrés (c'est-à-dire le carré de la différence entre le jugement de pertinence de référence et le jugement prédit) pour optimiser le classement des premiers résultats d'une liste.
Dans le cas où l'on considère des classes de pertinence, le problème d'apprentissage pourra être vu comme un problème de *classification* ou de *régression logistique ordinale*, selon que l'on considère ou non une relation d'ordre entre les classes de pertinence. Dans le cas de la classification, les algorithmes ne prennent pas en compte les éventuelles relations d'ordre pouvant exister entre les classes de pertinence. La littérature distingue des algorithmes traitant des pertinences binaires (le document est pertinent vs non pertinent) et des algorithmes traitant des classes de pertinence multiples (pertinent, moyennement pertinent, non pertinent). Ces algorithmes utilisent des techniques de classification connues en apprentissage automatique Nallapati [22] a ainsi utilisé les machines à support de vecteurs (SVM) pour prédire les classes de pertinence dans le cas de pertinence binaire. Il s'agit ici d'une application directe des SVM proposé par Vapnik [27]. Ceux-ci cherchent à calculer l'hyperplan permettant de correctement séparer les données tout en étant le plus loin possible de chaque observation. Li et al. [17] ont montré que le problème de classification multiple pouvait être résolu par une application directe des méthodes de boosting [9]. Contrairement aux algorithmes de classification, les algorithmes basés sur la régression ordinale prennent en compte les relations d'ordre existant entre les classes dans le processus d'apprentissage. Considérant *k* catégories, l'objectif est de trouver les seuils b_k ($b_1 < b_2 < .. .< b_{K-1}$) qui délimitent les classes et la fonction *f* qui répartit correctement les prédictions dans chaque classe ainsi délimitée. L'algorithme Prank [6] est un algorithme itératif dédié à cette tâche. Considérant un vecteur de poids *w* et des seuils initiaux b_k, alors la quantité $\hat{y}_i = \arg\min_k \{w^T x_i - b_k < 0\}$ est calculée à chaque itération. Elle est alors comparée à la référence y_i. Si l'algorithme a fait une erreur, le vecteur *w* et le seuil sont alors modifiés. Le processus est itéré jusqu'à ce que l'algorithme converge.

Dans l'approche par paire (*pairwise)*, des paires de documents (x_i, x_j) sont considérées en entrée du système d'apprentissage. A chaque paire de documents est associé un jugement de préférence $y_{i,j}$ à valeur dans {-1,1}. Si $y_{i,j} = 1$, alors le document x_i est préféré au document x_j : il doit être classé au dessus de x_j dans la liste de résultat. La préférence est notée $x_i \succ x_j$. Au contraire, si $y_{i,j} = -1$, alors le document x_j est préféré au document x_i et on note $x_i \succ x_j$. Le problème d'apprentissage est ici un problème de classification, dans le cas particulier de paires d'instances. Par conséquent, la plupart des algorithmes de cette approche utilisent des adaptations de classifieurs existants. Ainsi, Burges et al [2] ont proposé RankNet, un algorithme basé sur les réseaux de neurones. Freund et al. [8] ont développé l'algorithme RankBoost, une adaptation d'Adaboost [9] pour les paires de documents. Enfin, Joachims [14] a implémenté Ranking SVM, un algorithme basé sur les SVM.

Dans l'approche par liste (*listwise)*, la liste complète et ordonnée des documents est considérée en entrée du système d'apprentissage. Les algorithmes fournissent en sortie la liste ordonnée des documents ou la liste de leurs scores de pertinence. Les algorithmes sont répartis en deux sous-catégories au sein de cette approche : ceux minimisant une fonction d'erreur définie à partir d'une mesure de RI comme la MAP ou le NDCG et ceux minimisant une fonction de perte non liée à une mesure de RI.
Les travaux de la première catégorie suivent la logique suivante : puisque la performance des algorithmes est évaluée à partir de mesures de RI, les algorithmes doivent apprendre les fonctions en maximisant ces mesures. Considérant une mesure *m*, les algorithmes cherchent ainsi à minimiser soit une approximation de la quantité 1-*m* [26][31], soit une approximation de la borne supérieure de la quantité 1-*m* [29][30]. Les travaux de la deuxième catégorie cherchent quant à eux à minimiser le nombre de permutations entre la liste de référence et la liste restituée par l'algorithme. De nombreuses fonctions de perte ont été proposées dans ce but [3][28].

2.3 Un enjeu : la construction de collections d'apprentissage

Les algorithmes de *learning to rank* sont évalués sur leur capacité à retourner les documents pertinents en haut de la liste, via différentes mesures de RI (MAP, NDCG). L'algorithme le plus performant sera celui pour lequel la valeur de la mesure est la plus grande. Généralement, la performance est mesurée sur les collections d'apprentissage de référence LETOR 3.0 ou LETOR 4.0 [19][20]. Ces collections ont été développées exclusivement pour l'évaluation des algorithmes d'ordonnancement. Les requêtes et documents utilisés sont extraits de deux autres collections de référence en RI : OHSUMED, constituée d'article médicaux et TREC 2003 et 2004, constitué des pages web du domaine .gov. Pour les collections LETOR, l'ensemble des similarités pour chaque couple requête-document est ici pré-calculée et les jugements de pertinence sont donnés par des experts humains. Si ces collections présentent l'avantage de permettre une évaluation et une comparaison aisées des différents algorithmes dans le cadre de moteurs généralistes, elles peuvent ne pas être adaptées lorsque l'on considère certains éléments du contexte, comme par exemple la nouveauté des résultats proposés. Elles sont par ailleurs très longues et coûteuses à obtenir. D'autres méthodes de construction de collections d'apprentissage, basée sur l'utilisation des connexions de recherche, ont ainsi été développées.

Les fichiers de connexion stockent chaque jour l'ensemble des actions réalisées par les utilisateurs sur les moteurs de recherche, notamment les requêtes soumises, les documents restitués et les résultats cliqués. Les clics peuvent être considérés comme des indicateurs implicites de la pertinence. En effet, un utilisateur cliquera sur un résultat s'il est intéressé et si celui-ci répond a priori à son besoin. Le fait que le résultat soit cliqué traduit donc une certaine forme de pertinence. De nombreuses approches ont proposé d'utiliser les clics comme jugements de pertinence pour la création de collection d'apprentissage.

Joachims et al. [14][15] ont ainsi testé différentes stratégies d'extraction de préférences dans le cadre d'algorithmes. Parmi celles-ci, la plus connue, car la plus performante, est la stratégie *SkipAbove*. Elle stipule que, considérant un ensemble de résultats restitués par le système et l'ensemble de résultats cliqués correspondants, le document cliqué de rang le plus faible est préféré à l'ensemble des documents non cliqués de meilleur rang. Ainsi, si $(d_1, d_2, d_3, d_4, d_5)$ est la liste des documents restitués et si (d_1, d_4) est la liste des documents cliqués, nous obtenons les préférences suivantes : $d_4 \succ d_2$ et $d_4 \succ d_3$. Ces préférences sont directement utilisables par les algorithmes basés sur l'approche par paires. Radlinski et al. [23] ont étendu ces stratégies dans le cadre de chaînes de requêtes, c'est-à-dire des successions de requêtes traduisant le même besoin mais tour à tour reformulées, généralisées ou spécialisées. La stratégie *SkipEarlierQuery* considère qu'un document cliqué est préféré à l'ensemble des documents non-cliqués à la requête précédente, dans la même chaîne de requête. Dans le cas où aucun document n'avait été cliqué pour la première requête, les auteurs considèrent qu'un document cliqué est préféré aux deux premiers documents de la requête précédente (stratégie *TopTwoEarlierQuery*).

Les approches basées sur les clics peuvent être biaisées par la présentation initiale des résultats. En effet, les utilisateurs sont généralement plus influencés par la position des résultats de recherche que par leur pertinence. Ainsi, ils ont tendance à cliquer sur les premiers résultats restitués par le système et à délaisser des documents situés plus bas dans la liste, même si ceux-ci sont les plus pertinents [20]. Des approches ont été proposées afin de prendre en compte ce biais de position dans la modélisation des clics. La pertinence est alors inférée à partir de l'ensemble des sessions utilisateurs. Les premiers modèles proposés ont été le modèle de position et le modèle en cascade [7]. Le modèle de position (« Position Model ») considère que l'utilisateur clique sur un seul document qui est alors le document pertinent. La probabilité qu'un document soit cliqué dépend uniquement de sa position dans la liste de résultat et diminue avec celle-ci. Le modèle en cascade (« Cascade Model ») suppose que l'utilisateur regarde les résultats du haut vers le bas de la liste. Pour chaque résultat, il choisit de cliquer sur le document qui est alors le seul document pertinent ou de passer au résultat suivant. L'utilisateur clique ainsi sur un seul document qui lui paraît pertinent compte tenu des résultats précédents. Il est important de noter que ces deux modèles postulent qu'il n'y a qu'un seul document pertinent, donc un seul clic, par session de recherche. Or, en pratique, les utilisateurs cliquent généralement sur plusieurs résultats au cours d'une même session. D'autres modèles ont ainsi été développés pour généraliser le modèle en cascade aux sessions pour lesquelles plusieurs résultats sont cliqués. Le modèle de clics dépendants (« Dependent click model ») considère que l'utilisateur consulte les résultats du haut vers le bas de la liste. Comme pour le modèle en cascade, l'utilisateur choisit à chaque étape de cliquer sur le document ou de passer au résultat. Mais, contrairement au modèle en cascade où il s'arrête une fois le premier document cliqué, l'utilisateur peut revenir sur la liste de résultat et consulter d'autres documents. La probabilité qu'un clic ait lieu sur le document d_i à la position i est définie de la façon suivante [12] :

$$c_{d_i,i} = r_{d_i} \prod_{j=1}^{i-1} \left(1 - r_{d_j} + \lambda_j r_{d_j}\right)$$

Les estimateurs de la pertinence r_d du document et du paramètre λ sont alors :

$$r_d = \frac{\text{Nombre de clics sur d}}{\text{Nombre de clics sur d avant la position du document considéré}}$$

$$\lambda_i = 1 - \frac{\text{Nombre de sessions pour lesquelles le dernier document cliqué est à la position i}}{\text{Nombre de sessions pour lesquelles la position i est cliquée}}$$

Le modèle bayésien dynamique (« Dynamic Bayesian Model ») [4] généralise également le modèle en cascade aux sessions à clics multiples et apporte une nouvelle contribution en définissant les notions de pertinence perçue et de pertinence effective. La pertinence perçue a_u correspond à la probabilité que le document soit cliqué. Elle traduit le fait que l'utilisateur ait été attiré par le résultat avant consultation du contenu du document. La pertinence effective s_u traduit la satisfaction de l'utilisateur après consultation du contenu du document. Soient les évènements E_i : le résultat est vu dans la liste, C_i : le résultat est cliqué, A_i : l'utilisateur est attiré par le résultat et S_i : l'utilisateur est satisfait par le document, le modèle est alors défini [4] par les équations suivantes (Tableau 1).

Tableau 1 : Modèle Bayésien Dynamique

E_i=1, $A_i = 1 \leftrightarrow C_i = 1$	Un résultat est cliqué si l'utilisateur a vu le résultat et a été attiré par celui-ci
$P(A_i = 1) = a_u$	La pertinence perçue est la probabilité que l'utilisateur soit attiré par le document
$P(S_i=1\|C_i=1)=s_u$	La pertinence effective est la probabilité que l'utilisateur soit satisfait sachant qu'il a cliqué sur le document
$S_i=1 \rightarrow E_{i+1} = 0$	Si un utilisateur est satisfait par un document, il arrête sa recherche
$C_i = 0 \rightarrow S_i = 0$	L'utilisateur ne peut être satisfait que s'il a cliqué et consulté le document
$P(E_{i+1} \mid E_i, S_i = 0) = \gamma$	L'utilisateur retourne sur la liste de résultat s'il n'a pas été satisfait avec une probabilité γ (il arrête sa recherche avec une probabilité 1- γ)
$E_i = 0 \rightarrow E_{i+1} = 0$	L'utilisateur ne regarde le résultat suivant que s'il a traité le résultat en cours

La pertinence globale du document est alors définie comme le produit de la pertinence perçue et de la pertinence effective $a_u s_u$. Elle est utilisée pour la construction des échantillons d'apprentissages. Liu et al. [18] ont également proposé un modèle bayésien, spécialement adapté à l'apprentissage de préférences dans les algorithmes d'ordonnancement de paires d'instances.

3 Limites en apprentissage d'ordonnancement

Les algorithmes d'apprentissage de fonctions d'ordonnancement et les modèles de clics sont généralement considérés comme performants. Néanmoins, la grande majorité des méthodes a été développée dans le cadre de moteurs généralistes. Elles peuvent donc ne pas être adaptées pour une utilisation sur d'autres types de données, ou sur des moteurs spécialisés. De nouvelles pistes d'adaptation doivent donc être explorées. Nous détaillons plus précisément dans cette partie deux limites rencontrées et présentons des solutions possibles.

3.1 Des approches peu adaptées aux spécificités des requêtes

Les algorithmes d'apprentissage d'ordonnancement présentés apprennent une fonction d'ordonnancement unique pour l'ensemble des requêtes. Ils supposent donc que toutes les requêtes peuvent être traitées de la même manière. Or les requêtes peuvent grandement différer à plusieurs niveaux. Elles peuvent utiliser des sémantiques différentes, être longues ou courtes, fréquentes ou rares, avoir ou non une dimension géographique, être plus ou moins ambiguës ou encore traduire des buts utilisateurs distincts. De nombreuses études se sont intéressées aux différentes façons de regrouper les requêtes.

Certains travaux ont proposés de classer les requêtes suivant le but supposé de l'utilisateur. Broder [1] a ainsi proposé une typologie regroupant les requêtes en trois grandes catégories: *navigationnelles*, *informationnelles* et *transactionnelles*. Les requêtes *navigationnelles* correspondent à la recherche d'un site internet spécifique. Par exemple, un utilisateur qui soumet la requête *« air france »* cherche probablement la page d'accueil du site *www.airfrance.fr* de la compagnie ferroviaire française. L'utilisateur attend a priori un seul résultat. Les requêtes *informationnelles* correspondent à la recherche d'une information pouvant être présente sur une ou plusieurs pages web. Par exemple, l'utilisateur qui soumet la requête *« sites touristiques Dordogne »* peut être uniquement intéressé par le site du comité départemental du tourisme de Dordogne ou par l'ensemble des sites des offices de tourisme de Dordogne. Enfin, les requêtes *transactionnelles* traduisent que l'utilisateur recherche un site Internet dans le but d'y effectuer une transaction ultérieure, par exemple un achat, un téléchargement ou une réservation. Ainsi, la requête *« billet de train Toulouse Sarlat »* semble indiquer que l'utilisateur cherche un site internet pour acheter un billet de train. Il s'agit donc d'une requête transactionnelle. Rose et al. [24] puis Jansen et al. [13] ont également proposés des typologies de requêtes reprenant les trois grandes catégories de Broder. Néanmoins, certains travaux ont mis en évidence que ces typologies peuvent ne pas être représentatives des requêtes sur des moteurs spécialisés, par exemple des moteurs de recherche de blogs [21]. D'autres études se sont plus particulièrement intéressées aux requêtes portant sur des informations géoréférencées. Notamment, Gan et al. [10] ont mené une analyse des requêtes géographiques extraites des logs du moteur de recherche AOL et ont proposé de regrouper les requêtes géographiques en 23 catégories combinant le but utilisateur et le thème de la recherche. Ils ont également évalué des méthodes permettant de séparer les requêtes géographiques des requêtes non géographiques d'une part, et de séparer les requêtes géographiques navigationnelles des requêtes géographiques informationnelles d'autre part.

Parallèlement, l'apprentissage de fonctions d'ordonnancement dépendantes des requêtes ou de catégories de requêtes est vu comme un moyen d'améliorer le classement des résultats restitués par le système [19][20]. Kang et al. [16] ont ainsi classifié les requêtes en s'inspirant de la typologie de Broder et utilisé des fonctions distinctes crées manuellement pour classer les documents. Ils ont montré que cette approche permettait d'améliorer le classement. Geng et al. [11] ont proposé une approche basée sur l'algorithme des k plus proches voisins pour regrouper les requêtes et apprendre des fonctions spécifiques. Pour chaque requête de l'échantillon d'apprentissage, ils sélectionnent les k plus proches requêtes présentes dans l'échantillon. Un modèle est automatiquement appris grâce à l'algorithme Ranking SVM sur le sous-ensemble de données ainsi constitué. Lorsqu'une nouvelle requête est soumise au système, ses k plus proches voisins sont sélectionnés. Ce groupe de requêtes est alors comparé à chaque sous-ensemble pré-calculé afin de déterminer celui qui contient le plus de requêtes communes. La fonction d'ordonnancement correspondante est alors utilisée pour classer les résultats.

Ces approches donnent de bons résultats, mais n'ont jamais été évaluées, à notre connaissance, sur des jeux de données issus de moteurs géoréférencés. Nous proposons ainsi d'adapter l'apprentissage des fonctions d'ordonnancement au type de la requête, dans le cadre particulier des moteurs de recherche d'information géoréférencé. Nous proposons de tester deux approches et de les confronter. La première pourra consister à définir une typologie de requêtes géoréférencées et à apprendre des fonctions spécifiques à chaque groupe de requêtes. Cette typologie pourra, dans une première étape, distinguer les requêtes navigationnelles des requêtes informationnelles, puis dans une seconde étape, séparer les requêtes suivant le type de lieux recherché (restaurant, hôtels, etc). La deuxième approche pourra consister à adapter la méthode basée sur les k plus proches voisins au cas géoréférencé, pour les requêtes informationnelles.

3.2 Des collections d'apprentissage non représentatives

Les modèles présentés à la section 2.3 sont performants pour prédire correctement les clics et donc la pertinence. Néanmoins, ils restent limités à une utilisation sur des moteurs de recherche généralistes. Nous entendons par généralistes des moteurs qui traitent des documents non spécifiques et qui présentent leurs résultats sous forme de liste d'URLs. L'utilisateur qui souhaite consulter un résultat clique sur le lien vers le document. Un seul clic est possible pour chaque référence. Or, de plus en plus de moteurs proposent de visualiser les résultats sous forme de fiches comportant plusieurs éléments cliquables. Plusieurs actions sont alors possibles pour un

même résultat et chaque clic peut avoir une importance différente concernant la pertinence. Nous pouvons citer en exemple le moteur de recherche du site « The European Library »[2] qui permet de consulter les collections des bibliothèques nationales européennes. Chaque résultat est présenté sous la forme d'une fiche bibliographique comportant plusieurs liens cliquables permettant l'accès au document complet en ligne, l'accès au document au format pdf ou encore le téléchargement du document. L'utilisateur peut réaliser plusieurs actions sur un même résultat et ces actions peuvent refléter des degrés distincts de satisfaction de l'utilisateur. Ainsi, nous pourrions considérer qu'un document qui a affiché au format pdf puis téléchargé est plus pertinent qu'un document qui a été seulement affiché. Les modèles de clics actuels ne sont pas conçus pour prendre en compte ces nuances. Nous travaillons sur une nouvelle approche pour la modélisation de la pertinence. Cette approche considère la pertinence d'un document comme la somme pondérée de l'ensemble des clics enregistrés sur le résultat. Une étude à grande échelle est en cours pour analyser de façon plus précise la contribution des différents types de clics. Nos premiers résultats mettent en évidence que chaque clic n'a effectivement pas la même représentativité pour l'utilisateur. A terme, cette approche permettra de mieux évaluer la pertinence et pourra être utilisée comme une méthode générique permettant de créer des jeux de données d'apprentissage adaptés aux spécificités du moteur considéré.

4 Conclusion

Nous avons présenté le principe général de l'ordonnancement des résultats sur les moteurs de recherche. Nous avons également introduit les principales approches et algorithmes existant dans la littérature. Nous avons constaté que les approches actuelles étaient limitées car elles ne peuvent pas être généralisées aux moteurs spécialisées. En effet, elles ne prennent pas en compte les spécificités des moteurs, des utilisateurs et des requêtes. Nous proposons des pistes d'adaptations et d'améliorations basées sur une meilleure connaissance des comportements des utilisateurs et sur la détermination de critères propre à différents scénarios de requêtes. Nous proposons d'appliquer ces méthodes dans le cadre des moteurs géoréférencés. Plus particulièrement, nous pourrons les adapter dans le cadre spécifique du moteur de recherche géoréférencé Nomao. En améliorant l'ordonnancement des résultats par l'utilisation de critères spécifiques aux catégories de requêtes, des études devraient permettre d'augmenter la performance du moteur et la satisfaction utilisateur. Elles pourront aussi fournir des indications importantes pour les entreprises afin d'améliorer leur compréhension des processus de choix des utilisateurs. Notre étude sur les différents types de clics permettra ainsi de mieux appréhender ces mécanismes utilisateurs et surtout, de créer des jeux de données d'apprentissage adaptés à chaque moteur. L'application de ces méthodes devrait ainsi conduire à la création d'une base d'apprentissage dédiée aux moteurs géoréférencés, qui pourra être ouverte à la communauté scientifique.

5 Bibliographie

[1] BRODER A., *A taxonomy of web search*, SIGIR Forum, 2002, vol. 36, n° 2, p. 3-10.

[2] BURGES C.J.C., SHAKED T., RENSHAW E., LAZIER A., DEEDS M., HAMILTON N., HULLENDER G., *Learning to rank using gradient descent*, Proceedings of the 22nd International Conference on Machine Learning (ICML), 2005, p. 89-96.

[3] CAO Z., QIN T., LIU T.Y., TSAI M.F., LI H., *Learning to rank : From pairwise approach to listwise approach*, Proceedings of the 24th International Conference on Machine Learning (ICML), 2007, p. 129-136.

[4] CHAPELLE O., ZHANG Y., *A dynamic bayesian network click model for web search ranking*, Proceedings of the 18th International Conference on World Wide Web (WWW), 2009, p. 1-10.

[2] www.theeuropeanlibrary.org

[5] COSSOCK D., ZHANG T., *Subset ranking using regression*, Proceedings of the 19th Conference on Learning Theory (COLT), 2006, p. 605-619.

[6] CRAMMER K., SINGER Y, *Pranking with ranking*, Advances in Neural Information Processing Systems, 2001, vol. 14, p. 641-647.

[7] CRASSWELL N., ZOETER O., TAYLOR M., RAMSEY B., *An experimental comparison of click position-bias models*, Proceedings of the 1st International Conference of Web Search and Data Mining (WSDM), 2008, p. 87-94.

[8] FREUND Y., IYER R., SCHAPIRE R., SINGER Y., *An efficient boosting algorithm for combining preferences*, Journal of Machine Learning Research, 2003, vol.4, p. 170-178.

[9] FREUND Y., SCHAPIRE R., *A decision-theoretic generalization of online learning and an application to boosting*, Journal of Computer and System Sciences, 1995, vol. 55, n°1, p. 119-139.

[10] GAN Q., ATTENBERG J., MARKOWETZ A., SUEL T., *Analysis of geographic queries in a search engine log*, Proceedings of the 1st international workshop on Location and the web (LOCWEB), 2008, p. 49-56.

[11] GENG X.B., LIU T.Y., QIN T., LI H., SHUM H.Y., *Query-dependant ranking using k-nearest neighbor*, Proceedings of the 31st annual international SIGIR Conference on Research and Development in Information Retrieval, 2008, p. 115-122.

[12] GUO F., LIU C., WANG Y.M., *Efficient multiple-clicks models in web search*, Proceedings of the 1st International Conference of Web Search and Data Mining (WSDM), 2008, p. 124-131.

[13] JANSEN B.J., BOOTH D.L., SPINK A., *Determining the informational, navigational and transactional intent of web queries*, Information Processing and Management, 2008, vol. 44, n° 3, p. 1251-1266.

[14] JOACHIMS T., *Optimizing search engines using clickthrough data*, Proceedings of the 8th ACM Conference on Knowledge Discovery and Data Mining (KDD), 2002, p. 133-142.

[15] JOACHIMS T., GRANKA L., PAN B., HEMBROKE H., GAY G., *Accurately interpreting clickthrough data as implicit feedback*, Proceedings of the 28th annual international SIGIR Conference on Research and Development in Information Retrieval, 2005, p. 154-161.

[16] KANG I., KIM G., *Query classification for web document retrieval*, Proceedings of the 26th annual international SIGIR Conference on Research and Development in Information Retrieval, 2003, p. 64-71.

[17] LI F., BURGES C., WU Q., *McRank : learning to rank using multiple classification and gradient boosting*, Advances in Neural Information Processing Systems (NIPS), 2008, p. 845-852.

[18] LIU C., FALOUTSOS C., *BBM : Bayesian Browsing Model for petabyte-scale data*, Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2009, p. 537-546.

[19] LIU T.Y., Learning to Rank for Information Retrieval, Springer, 2011

[20] LIU T.Y., XU J., QIN T., XIONG W., LI H., *LETOR : Benchmark dataset for research on learning to rank for information retrieval*, Proceedings of SIGIR 2007 Worshop on Learning to Rank for Information Retrieval.

[21] MISHNE G., DE RIJKE M., *A study of blogs search*, Proccedings of the 28th European Conference on Information Retrieval (ECIR), 2006, p. 289-301.

[22] NALLAPATI R., *Discriminative models for information retrieval*, Proceedings of the 27th annual international SIGIR Conference on Research and Development in Information Retrieval, 2004, p. 64-71.

[23] RADLINSKI F., JOACHIMS T., *Query chains : Learning to rank from implicit feedback*, Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2005, p. 239-248.

[24] ROSE D.E., LEVINSON D., *Understanding user goals in web search*, Proceedings of the 13th International Conference on the World Wide Web (WWW), 2004, p. 13-19.

[25] SALTON G, Automatic Information Organization and Retrieval, McGraw Hill Text, 1968.

[26] TAYLOR M., GUIVER J., *SoftRank : optimising non-smooth rank metrics*, Proceedings of the 1st International Conference on Web Search and Data Mining, 2008, p. 77-86.

[27] VAPNIK V.N., Statistical learning theory, Wiley Inter Science, 1999.

[28] XIA F., LIU T.Y., WANG J., ZHANG W., LI H., *Listwise approach to Learning to rank – Theory and Algorithm*, Proceedings of the 25th International Conference on Machine Learning (ICML), 2008

[29] XU J., LI H., *AdaRank : A boosting algorithm for information retrieval*, Proceedings of the 30th annual international SIGIR Conference on Research and Development in Information Retrieval, 2007, p.107-114.

[30] YUE Y., FINLEY T., RADLINSKI F., JOACHIMS T., *A support vector method for optimising average precision,* Proceedings of the 30th annual international SIGIR Conference on Research and Development in Information Retrieval, 2007, p. 271-278.

[31] ZOETER O., TAYLOR M., SNELSON E., GUIVER J., CRASWELL N., SZUMMER M., *A decision-theoretic framework for ranking using implicit feedback*, SIGIR Workshop on Learning to Rank for Information Retrieval, 2008.