Rapport OUTILEX

Laboratoire d'Informatique de Paris 6
B. Piwowarski

September 18, 2006

1 Introduction

Les systèmes de Recherche d'Information (RI), permettent de rechercher dans de grand corpus électronique les documents les plus pertinents pour des demandes d'information généralement exprimées sous forme de mots-clé par des utilisateurs. Ces systèmes, popularisés par des moteurs de recherche tel que Google ou Yahoo, permettent un accès à une information dont l'unité indivisible la plus petite est le document.

Cette dernière décennie, la notion de document électronique a considérablement évolué. Nous sommes passés en peu de temps d'un monde où le concept dominant était celui du document "plat" – à savoir d'un texte constitué d'une suite de mots sans aucune information de structure – à un monde où le document est devenu un objet plus complexe, structuré, et pouvant comporter différentes médias. Cette évolution a été accélérée par le développement du Web qui propose lui même ces formats structurés pour organiser et décrire ses propres ressources.

Avec cette évolution de la nature des sources d'information, sont apparus de nouveaux besoins qui visent à exploiter la richesse présente dans ces documents. Le format d'un document est aujourd'hui défini par une structure logique décrite par des instances du langages XML. Le format XML permet par exemple de structurer un document de manière logique, par exemple sous forme de chapitres, sections, et paragraphes. Chaque document XML est ainsi défini par une arborescence logique formée d'éléments (e.g. l'information structurelle) et de son contenu (image, texte, etc.). L'arborescence du document donne la possibilité d'accéder à des éléments plus fin que le document entier et permet d'envisager une recherche plus précise et "focalisée".

Des formats particuliers ont été proposés pour standardiser certains types de documents et en faciliter ainsi leur exploitation. Le traitement de ces nouvelles sources d'information transforme profondément les domaines traditionnels de l'accès à l'information. Il faut forger de nouveaux concepts pour la représentation, le stockage, le traitement de cette information, pour l'interaction avec l'utilisateur, pour la manipulation de très grosses masses de données à fort contenu sémantique. Les outils classiques pour le traitement des informations

textuelles sont mal adaptés à l'exploitation de cette information bien plus riche et doivent être adaptés ou plus probablement redéfinis.

Une première initiative internationale (INEX [1], "INitiative for the Evaluation of XML Retrieval") ayant pour but de constituer un corpus de documents XML qui permette l'évaluation de systèmes de Recherche d'Information Structurée (RIS) a vu le jour en avril 2002. Chaque année, un corpus de documents structurés sert de base à la constitution d'un ensemble de questions et des jugements de pertinence associés : ces derniers associent à un ensemble d'éléments du corpus un jugement qui défini la pertinence d'un élément par rapport à la question posée.

Le projet Outilex vise à mettre à la disposition de la recherche, du développement et de l'industrie une plate-forme logicielle de traitement des langues naturelles ouverte et compatible avec l'utilisation de XML, d'automates finis et de ressources linguistiques.

Le travail effectuée par le LIP6 a été le suivant:

- Développement d'un module python permettant d'utiliser certaines fonctions d'Outilex depuis un programme écrit en Python;
- Développement d'une extension de la plateforme SIRXQL (plateforme pour l'accès à l'information structurée développée au LIP6) pour la Recherche d'Information Structurée. Il a été développé les fonctionnalités suivantes:
  - Indexation de la structure des documents XML.
  - Pré-traitements linguistiques utilisant un lemmatiseur (TreeTagger) et Outilex pour détecter les mots-composés.
  - Constitution de l'index permettant une recherche plus rapide dans les documents structurés.
  - Serveur permettant de répondre à des questions composées de mots-clé.

2 Modèle de recherche

Le modèle de recherche utilisé dans notre projet est une version d'Okapi [4] adaptée à la recherche dans des corpus structurés. Okapi est un des modèles de recherche documentaire les plus performants et est basé sur un formalisme probabiliste. Avec ce système de RI, le score attribué à un élément X pour une question q donnée est défini par:

$$\text{Okapi}(q, X) = \sum_{j=1}^{\text{longueur}(q)} \omega_{j,X} \frac{(k_1+1)\text{tf}_{X,j}}{K_X + \text{tf}_{X,j}} \times \frac{(k_3+1)\text{qtf}_j}{k_3 + \text{qtf}_j} \qquad (1)$$

où k_1 et k_3 sont des constantes[1], longueur(q) est le nombre de termes de la question q. L'équation (1) est similaire à celle du modèle Okapi original à

[1] Nous avons utilisé les valeurs standard des paramètres d'Okapi (k_3 =1.2, k_3 = 7 et b= 0.75)

l'exception de l'indice X qui est présent dans les termes $\omega_{j,X}$, K_X et $\mathrm{tf}_{X,j}$: Okapi utilise différentes statistiques liées à la collection de documents et à la question pour calculer le score entre un document et une question. Parmi celles-ci, la longueur moyenne d'un document et le nombre de documents dans lequel apparaît un terme doivent être adaptés à la recherche structurée. Les termes $\omega_{j,X}$ et K_X ont donc été défini de la manière suivante :

- $\omega_{j,X} = \log(\frac{N-n_j+0.5}{n_j+0.5})$. Pour Okapi N est le nombre de documents dans la collection et n_j le nombre de documents contenant le terme j. Il y a différentes possibilités pour adapter n_j à la recherche structurée: il est possible de compter le nombre de documents, mais également le nombre d'éléments ou de la calculer pour les éléments d'un type donné (par exemple, calculer une statistique pour les paragraphes, une autre pour les sections, etc.).
- $K_X = k_1((1-b) + b\frac{dl}{avdl})$ où b est une constante, dl la longueur du document et $advdl$ la longueur moyenne des documents. Comme pour ω, la statistique $advdl$ peut être calculée de différentes manières dans un corpus structuré : il est possible de calculer la longueur moyenne des éléments, des éléments de même type ou bien encore des éléments qui sont apparaissent au même niveau de la hiérarchie.

Les résultats obtenus avec le corpus d'INEX 2004 ont montré qu'un des meilleurs modèles combinait :

- Le calcul de n_j comme le nombre d'éléments dans lequel apparaît un terme donné.
- Le calcul de $advdl$ comme la moyenne des éléments de même type.

C'est ce modèle qui a été développé dans SIRXQL. Notons qu'il serait facile d'intégrer des variations de ce modèle au sein de ce projet, ainsi que de proposer des modèles plus complexes – comme par exemple des modèles capables d'apprendre leurs paramètres.

Pour le pré-traitement, nous avons utilisé "tree tagger" [3], un lemmatiseur basé sur un modèle d'arbre de décision probabiliste. Outilex a été utilisé pour la détection de mots composés : la Figure 1 est une représentation graphique de la (très simple) grammaire utilisée pour la détection de mots-clé. Dans le futur, il sera facile de détecter d'autres formes composées utiles pour la recherche d'information.

3 Implémentation

Le module décrit dans la section précédente a été dans l'environnement SIRXQL, une plateforme pour l'accès à l'information structurée qui s'appuie sur un ensemble de scripts qui manipulent des tables stockées dans une base de données (BDD) telle que MySQL. Voici l'ensemble des scripts développés dans le cadre du projet Outilex:

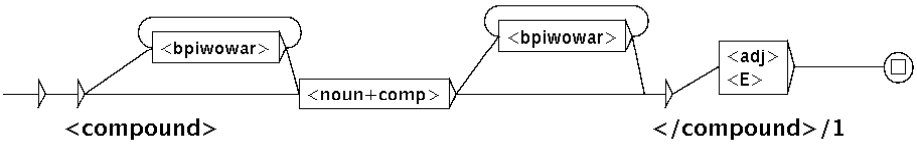

Figure 1: Grammaire WRTN utilisée pour détecter les mots-clé

1. Copie un ensemble de fichiers XML dans la BDD.
2. Indexation structurelle des fichiers XML en employant le codage "pre-size", où chaque élément du corpus XML est identifié par un entier. Ce codage permet une identification rapide des éléments qui contiennent ou contenant un ou des éléments de référence, permettant ainsi une recherche plus rapide.
3. Indexation linguistique, en utilisant TreeTagger et optionnellement Outilex pour la détection des mots-clefs. Une interface en Python a été spécialement développée afin de permettre l'appel aux fonctions de l'API d'Outilex.
4. Recherche structurée: l'algorithme Okapi décrit dans la sections précédente.
5. Serveur XMLRPC: Un serveur permettant de répondre à des appels XML-RPC, permettant ainsi d'avoir un serveur de recherche structuré.

4 Expériences

Des expériences ont été conduites pour déterminer l'influence de l'utilisation d'Outilex pour détecter les mots-clé sur les performances. La collection utilisée est celle d'INEX 2005, constituée de 16819 documents de revues de IEEE pour une période allant de 1995 à 2004, pour un total de 11 million d'éléments et de 764Mo. Les documents sont de tailles variées et leur contenu couvre l'intégralité des domaines de recherche en informatique.

Deux expériences ont été conduites, l'une en utilisant uniquement TreeTagger pour l'indexation, l'autre utilisant TreeTagger et Outilex. Parmi l'ensemble des questions d'INEX 2005, celles composées de mots-clé uniquement ont été utilisées.

La mesure d'évaluation utilisée pour comparer les performances des deux systèmes de recherche d'information ainsi obtenus est la mesure officielle d'INEX 2005 décrite dans [2]: il s'agit d'une métrique spécifique à la recherche d'information structurée. Les évaluations effectuées avec cette métrique se représentent sous forme de courbes effort - gain. Le gain (entre 0 et 1, en ordonnée) correspond au pourcentage d'éléments pertinents retrouvés par le moteur de recherche.

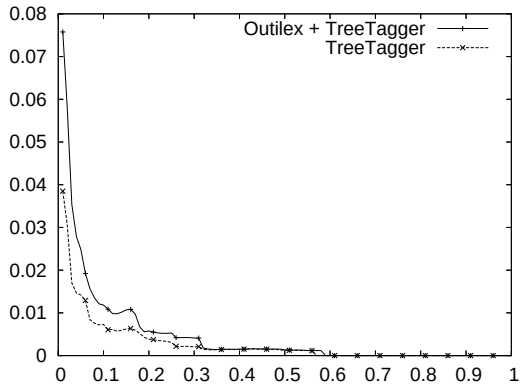

Figure 2: Comparaison des performances du système de recherche avec et sans l'utilisation d'Outilex pour la détection des mots clefs. La courbe est un graphe effort (abcisse)-gain (ordonnée)

L'effort, entre 0 et 1 également, correspond à la proportion d'effort supplémentaire nécessaire par rapport à un système idéal. Par exemple, le point de gain 0.2 et d'effort 0.05 signifie que pour retrouver 20% des éléments pertinents pour la demande d'information de l'utilisateur, celui-ci devra considérer 20 fois plus d'items dans la liste retourner que si celui-ci avait utilisé le meilleur de recherche qu'il est possible d'imaginer.

La figure 2 compare les performances de deux systèmes dont la seule différence tient dans l'indexation pour une des tâches d'INEX où le but est de retourner les éléments pertinents, en plaçant ceux qui ont la meilleure taille en premier (ainsi, un paragraphe ne contenant que du texte pertinent doit être placé avant la section qui le contient). Les courbes montrent une légère différence à l'avantage de l'indexation des mots-clé, en particulier pour des niveaux de gains faibles: il y a bien là un gain dans la précision des éléments renvoyés.

Nous avons également effectué un test statistique afin de vérifier la différence entre les deux modèles au niveau de l'effort moyen (équivalent de la précision moyenne utilisée couramment en recherche d'information), mais sans arriver à aucune conclusion définitive. Le test utilisé est le t-test de Student avec un niveau de confiance de 0,95. Il faudrait plus de questions avant de pouvoir affirmer que statistiquement la détection des mots-clé est un avantage au niveau de l'effort moyen. Avant de mener des expériences plus approfondies, il serait bon d'affiner les paramètres du modèle afin de l'amener au niveau des meilleurs modèles de recherche d'information structurée.

5 Conclusions

Ce rapport a présenté le travail effectué par le LIP6 dans le cadre du projet Outilex. L'apport principal du LIP6 est le développement d'un moteur de recherche

d'information capable de travailler avec des corpus de documents structurés tels que les documents XML. Des expériences permettant d'étudier l'apport de traitements linguistiques plus poussés, tel la détection de mots-clé, ont été conduites. Les premières expériences montrent que ce type de pré-traitement améliore la précision du moteur de recherche. Des expériences supplémentaires sont nécessaires pour valider statistiquement l'amélioration.

References

[1] Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Gabriella Kazai, editors. *INEX 2005 Proceedings*, 2005.

[2] Gabriella Kazai, Lalmas Mounia, and Arjen de Vries. Reliability tests for the xcg and INEX 2002 metrics. In Norbert Fuhr, Mounia Lalmas, Saadia Malik, and Zoltán Szlávik, editors, *INEX 2004 Proceedings*, 2004.

[3] Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *International Conferenceon New Methods in Language Processing*, Manchester, UK, 1994.

[4] S. Walker and Stephen E. Robertson. Okapi/keenbow at TREC-8. In E. M. Voorhees and D. K. Harman, editors, *NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC-8)*, Gaithersburg, Maryland, USA, November 1999.