MRM: An Adaptive Framework for XML Searching

Ho Lam Lau
Division of Computer Studies,
The Community College of City University
Hong Kong, China
holamlau@cityu.edu.hk

Wilfred Ng
Department of Computer Science and Engineering,
The Hong Kong University of Science and Technology
Hong Kong, China
wilfred@cse.ust.hk

ABSTRACT
In order to deal with the diversified nature of XML documents as well as individual user preferences, we propose a novel Multi-Ranker Model (MRM), which is able to abstract a spectrum of important XML properties and adapt the features to different XML search needs. The model consists of a novel three-level ranking structure and a training module called Ranking Support Vector Machine in a voting Spy Na¨ıve Bayes Framework (RSSF). RSSF is effective in learning search preference and then ranks the returned results adaptively. In this demonstration, we present our prototype developed from the model, which we call it the MRM XML search engine. The MRM engine employs only a list of simple XML tagged keywords as a user query for searching XML fragments from a collection of real XML documents. The demonstration presents an indepth analyses of the effectiveness of adaptive rankers, tailored XML rankers and a spectrum of low level ranking features.

Categories and Subject Descriptors
H.3.3 [**Information System**]: Information Search and Retrieval, Retrieval models

General Terms
Management, Measurement, Experimentation.

1. INTRODUCTION
The goal of our system is to obtain an effective ranked list of XML fragments as an answer to the user query, taking into account individual preferences on fragments [4]. In this demonstration, we present our work that extends traditional IR theory to rank XML fragments by considering the granularity of XML data and user feedback. The extension includes many important features in XML data such as key-tag proximity, data granularity and contextual paths as basic indexing units. Our prototype, called the MRM engine, is able to cater to the mixed searching needs from different users by providing a set of adaptive rankers at the top level, when given a corpus of diversified XML datasets.

The MRM engine consists of a training module called Ranking Support Vector Machine in a voting Spy Na¨ıve Bayes Framework (RSSF). RSSF refines the rankers in a non-intervening manner by learning users' feedback on the returned search results. The search performance of the trained ranker is shown to be adaptable to the users' preferences in XML searching. We show empirically that the RSSF is able to improve the MRM significantly in the learning process and needs only a small set of training XML fragments. We demonstrate that the trained MRM is able to bring out the strengths of the XML Rankers in order to adapt different preferences and queries.

CIKM'09, November 2-6, 2009, Hong Kong, China.
ACM 978-1-60558-512-3/09/11.

2. THE MULTI-RANKER MODEL (MRM)
As shown in Figure 1, the MRM is composed of three ranking levels. The lowest level consists of two categories of similarity and granularity features. At the intermediate level, we define four tailored XML Rankers (XRs), which are DAT, DOC, DFT and CUS. The XRs consist of different lower level features and have different strengths in searching XML fragments. The search preference of these four XRs is now explained as follows:

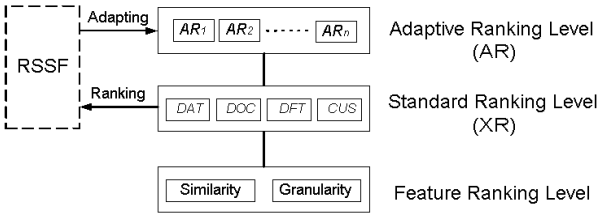

Figure 1. A three level MRM for adaptive XML searching.

The DAT ranker focuses more on the textual similarity than the structural complexity of the results. The DOC ranker targets on the structural part of fragments and aims at supporting those users who have more interests in fragments with similar structures than data values. The DFT ranker considers fairly on the features sets of the DOC and DAT schemes, that is, all similarity and granularity features are equally weighted. The CUS ranker allows the search engine administrator to define a customized ranking scheme, by tuning the lowest level ranking features.

The XRs are trained via a learning mechanism called the Ranking Support Vector Machine in a voting Spy Naïve Bayes Framework (RSSF). The RSSF takes as input a set of labeled fragments and feature vectors and generates as output Adaptive Rankers (ARs) in the learning process. The ARs are defined over the XRs and generated at the top level of the MRM. The RSSF is an enhancement of the Ranking Support Vector Machine (RSVM) algorithm proposed in [1,3], which is essentially a machine learning technique that is applied in our setting to optimize the performance of XML searching via key-tags. In order to discover preferences from analyzing the labeled fragments, we incorporate a novel *Voting Spy Naïve Bayes* (VSNB) algorithm into the RSSF, which generates the adaptive rankers at the top level.

Specifically, VSNB consists of two main components as follows: a spying technique is applied to obtain more accurate negative samples and a voting procedure to consider the information discovered by the spies. Given a set of preference fragments, we need to categorize *unlabeled* fragments in order to discover the Estimated Negative (EN) fragments. The challenge is that we need to make full use of all potential spies, since conventional

naïve Bayes requires both positive and negative examples as training data, while we only have positive examples as indicated by the preference feedback as discussed in [4].

We empirically justified that each XML ranker has its individual strengths in attaining good precision and ranking quality in different XML preferences in [4]. In our experiments, we only used limited training datasets to train the MRM and generate the adaptive rankers for different users in the learning process. The RSSF is able to make the learning process efficient and the adaptive rankers effective, even when the amount of learning data is relatively small and sparse.

3. IMPLEMENTATION OF MRM

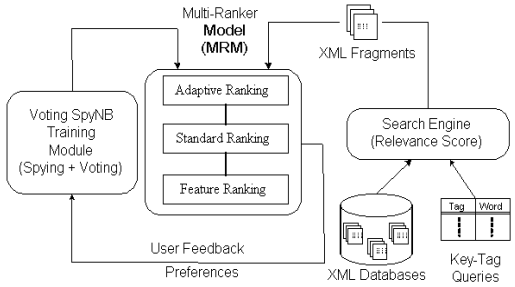

Figure 2. A simplified architecture of the MRM prototype.

Figure 2 shows the basic ideas and the logistics of how we search XML documents by using key-tags within the MRM prototype. When the user submits a key-tag query, the search engine will search the XML databases by matching the submitted key-tags. The query results (i.e. a set of XML fragments) are obtained based on the relevance scores which are detailed [4]. The obtained results (i.e. a set of labelled XML fragments) are then passed to the MRM module for ranking and sent to the user. The user's feedback is then sent to the Voting SpyNB Training Module for adapting the standard rankings. For each fragment in the returned result, we measure the relevance scores and pass them to the MRM. The MRM collects user preferences and uses the RSSF algorithm to optimize the adaptive ranking functions in the top level by assigning different weights to the features of XRs in the middle level. The result of adaptive ranking constitutes an adaptive ranking towards the preferences tailored to the user.

4. DEMONSTRATION PLAN

In this demonstration, we plan to set up the prototype in order to illustrate the principle and the effectiveness of our new strategies. First, we show the details of the spy operations with some search queries in VSNB in a walking manner. This will give interesting insights of the effect of spying and voting. Second, we demonstrate the effectiveness of the MRM by showing the contributions of the rankers at the three levels. This will show the mechanism of adaptive ranking in our approach. We prepare 30 queries [7] for the DBLP dataset [5] and the queries for INEX 2006 [2] are modified from the 125 INEX 2006 topics. Specifically, we have four specific tasks detailed as follows.

1. We plan to illustrate step by step how *Voting Spy Naïve Bayes* (VSNB) algorithm in the RSSF is involved in the ranking process. The query results for a key-tag query are shown in Figure 3. Users can select their preferences by clicking the checkboxes next to the preferred fragments. Then, the detailed information of the selected fragments, such as the scores of all low-level features can be found by clicking the "Show Details" button next to the checkboxes. The process continues when the user clicks on the "Spy" button which further displays the spying and voting behind the adaptive process.
2. We show the effectiveness of our adaptive ranker by comparing it with the four standard rankers (XRs) by getting the participants involved in searching our XML datasets. The effectiveness can be measured the two precision parameters used in INEX [6].

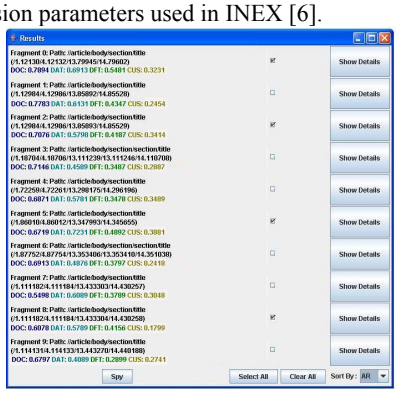

Figure 3. The query results and the XR ranker contributions in the MRM System.

3. We prepare several simulated user profiles adapted to various preferences such as academic, literature, science and commercial ones. For details of defining such profiles the readers may refer to [4]. We compare the adapted ranking results with each other using the same set of queries. We also show the effectiveness of our adaptive ranker by comparing the ranking results using a new user with unknown profile as the baseline.
4. In order to demonstrate the effectiveness of low level features in each XR of the MRM system, we allow the user to add or remove or customize low level features to form XRs through the "Low Level Feature Customization" service. By combining various similarity and granularity features, we illustrate the importance of the low level features adopted in each XR using the MAP and prec@10 as metrics to compare the baseline, adaptive rankers and a ranker that is composed of various selected low-level features.

5. REFERENCES

[1] Bennet, K., and Demiriz, A. (1998). Semi-supervised support vector machines. In: *Proc. of Conf. on Advances in neural information processing systems*, 268–374.

[2] Denoyer, L. and Gallinari, P. (2006). The wikipedia XML corpus. In *WORKSHOP SESSION: INEX reports.* ACM Press, New York, 64–69.

[3] Joachims, T. (2002). Optimizing search engines using clickthrough data. In: *Proc. of ACM SIGKDD*, 133–142.

[4] Lau, H., and Ng, W. (2008). A multi-ranker model for adaptive XML searching. *The International Journal on Very Large Data Bases*, Volume 17, Number 1, 57-80.

[5] Ley, M. (2004). Digital Bibliography & Library Project. (DBLP) In: http://dblp.uni-trier.de/, 2004.

[6] Fuhr, N., Lalmas, M., and Malik, S. (2006). INitiative for the Evaluation of XML Retrieval (INEX). In: http://inex.is.informatik.uniduisburg.de/2006/index.html.

[7] Lau, H. (2009). Full list of queries. In http://personal.cityu.edu.hk/~holamlau/mrm/.