Using Ontology-Based User Preferences to Aggregate Rank Lists in Web Search

Lin Li[1], Zhenglu Yang[1], and Masaru Kitsuregawa[2]

[1] Dept. of Info. and Comm. Engineering, University of Tokyo, Japan
[2] Institute of Industrial Science, University of Tokyo, Japan
{lilin, yangzl, kitsure}@tkl.iis.u-tokyo.ac.jp

Abstract. This paper studies rank aggregation by using ontology-based user preferences in the context of Web search. We introduce a set of techniques to combine the respective rank lists produced by different attributes of user preferences. Furthermore, the learned user preferences are structured as a taxonomic hierarchy (a simple ontology). We use the learned ontology to store the attributes such as, the topics that a user is interested in and the degrees of user interests in these topics. The primary goal of our work is to form a broadly acceptable rank list among these attributes by making use of *rank-based* aggregation. Experiment results on a real click-through data set show that our user-centered rank aggregation techniques are effective in improving the quality of the Web search in terms of user satisfaction.

1 Introduction

Nowadays, it becomes increasingly difficult for users to retrieve desired information due to the continued rapid growth in data volume and the ambiguity of short queries in Web searches. As we know, different users have different intentions for a same query. In order to satisfy the diverse needs of users, search engines should be adaptive to the individual contexts in which users submit their queries. Lawrence et al. [6] addressed an overview of the context of the Web search. User preferences are a kind of useful contexts. Shen et al. [11] developed a client-side Web search agent to perform implicit feedback and inferred user model from short-term search contexts to improve Web searches. The user preferences can be represented by a bag of words or a taxonomic hierarchy. The bag of word representation does not consider term correlations because terms in user preferences are considered in isolation from one another. The taxonomic hierarchy can overcome this drawback and has been widely accepted [2, 7, 10]. It is also the basic structure of modeling our user preferences. Furthermore, user preferences consist of a number of attributes, such as what kind of topics that users are interested in, and how much users are interested in each topic. Each attribute describes a user's favorite in different aspects. In most cases, any individual attribute is deficient in accurately representing user preferences. Combining user knowledge depicted by each attribute can help us understand user preferences well, which finally results in an effective rank mechanism in the Web search. To leverage

T. Washio et al. (Eds.): PAKDD 2008, LNAI 5012, pp. 923–931, 2008.

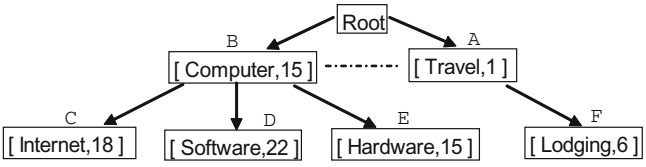

Fig. 1. Hierarchical Model of User Preferences

the rankings produced by the different attributes, rank aggregation intends to form a single rank list supported by a broad consensus among these attributes. There are two approaches: score-based and rank-based. The score-based rank aggregation merges the values of the attributes [2, 12]. However, it is important to observe that if the rank mechanism is score-based, the sequence implied by the scores makes it more meaningful than the actual scores themselves. On the other hand, the rank-based rank aggregation fuses the rank lists produced by the values of the attributes and has been studied and employed in many applications in the last half century [4, 9, 14]. Renda et al. [9] compared rank and score based methods without training data in the context of metasearch, and showed that Markov chain rank-based methods compete with score-based methods. Dwork et al. [4] developed the theoretical groundwork for describing and evaluating rank aggregation methods. Their main point is to effectively combat *spam*. In this paper we introduce methods to effectively improve the Web search in a context-aware manner.

In the rest of this paper, Section 2 describes rank-based rank aggregation, including how to produce and fuse user-centered rank lists. We report the experimental results in Section 3, and draw conclusions in Section 4 . From now on, the term "rank aggregation" means "rank-based" rank aggregation for simplicity.

2 Rank Aggregation

In the following part, we will discuss how to get the respective rank lists from the learned use preferences and the proposed rank aggregation methods.

2.1 Hierarchical Similarity Measures

Our user preferences are structured as a semantic hierarchy shown in Figure 1. Technical details about how to learn and update user preferences from click-through data are in our previous work [7]. For an effective rank mechanism, the more similar a search result is to user preferences, the higher position it will be put in the final rank list. To produce such a new rank list, hierarchical similarity measures are needed to assess the relatedness between user preferences and search results. We choose five content-ignorant measures from [8] because we want to see how much we can benefit from the hierarchical structure. The measures are defined as

$$S_1(i,j) = 2 \cdot M - l\,, \tag{1}$$

$$S_2(i,j) = \alpha S_1(i,j) + \beta h \qquad (\alpha = 0.05, \beta = 1)\,, \tag{2}$$

$$S_3(i,j) = e^{-\alpha \cdot l} \qquad (\alpha = 0.25)\,, \tag{3}$$

$$S_4(i,j) = \frac{e^{\beta \cdot h} - e^{-\beta \cdot h}}{e^{\beta \cdot h} + e^{-\beta \cdot h}} \qquad (\beta = 0.15)\,, \tag{4}$$

$$S_5(i,j) = e^{-\alpha \cdot l} \cdot \frac{e^{\beta \cdot h} - e^{-\beta \cdot h}}{e^{\beta \cdot h} + e^{-\beta \cdot h}} \quad (\alpha = 0.2, \beta = 0.6)\,, \tag{5}$$

where h means the depth of the subsumer (the deepest node common to two nodes), l is the naïve distance (the number of edges or the shortest path length between two nodes), i and j are nodes (topics) in Figure 1, and M is the maximum depth of topic directory possessed by user preferences. The values in parentheses are the optimal values of parameters [8].

2.2 Rank Lists Produced by Ontology-Based User Preferences

The above five content-ignorant measures can evaluate the hierarchical similarity between search results and user preferences. The degree of user preferences has effect on the similarity as well. There are various ways of combining the two kinds of similarity scores dependent on applications. Their product is commonly used in classic IR like our previous work. However, as we mentioned in Section 1, the ranking implied by the scores has more sense than the actual scores themselves. In the following discussion we calculate two user-centered rank lists plus the result list returned by Google for rank-based fusion, as distinguished from the traditional score-based combination.

(1) **Hierarchical Semantic Similarity.** User preferences include a number of topics (nodes) in Figure 1. We further define the semantic similarity between one search result and one user as the maximum value among all the values computed by any one of Equations 1-5. The search results then are re-ranked and form a rank list in order of one attribute of user preferences (i.e., the topics a user is interested in). The priori work [2], however, just selected one of them without analyzing their differences.

The five measures have their own features from their definitions. For example, compared to Equation 1, Equation 3 also uses the naïve distance alone, but makes use of a nonlinear function. Equation 2 is a linear combination of the naïve distance and the depth. Different from Equation 2, Equation 5 transfers the naïve distance and the depth by a nonlinear function, respectively, and then combines them by multiplication. Equation 4 is the transformation of the depth of the subsumer through a nonlinear function. Based on these differences, we think that it is necessary to experimentally compare their performances when they are applied in the context of the Web search and no priori work has done it. The experimental results are reported in Section 3.

(2) **Degree of User Interests.** We find that Equations 1-5 are not round in re-ordering search results. With the help of Figure 1, let us explain the problem clearly. The naïve distance between node A and node C (i.e., 3) is the same as that between node B and node F, and the subsumer of A and C (i.e., "root") is the same as that of B and F as well. As a result, computed by any equation from

Equations 1-5, the similarity score between A and C is equal to that between B and F. In this situation, these measures cannot order the two pairs. Our solution for this problem is intuitive that the degree of the user interests in a topic (node) can alleviate this problem. The more times a user clicks one topic, the more interested the user is in it. The user's clicked times can produce a complementary rank list of search results.

(3) **Google List.** Google applies its patented PageRank technology on the Google Directory to rank the sites. To keep our rank aggregation from missing the high quality Web pages in Google, we also consider the original rank list of Google Directory Search. As we know, there is a PageRank value accompanied with each search result, representing the popularity or authority of results. It certainly could be used to weigh the topics associated with results. Unfortunately, these values are not publicly available for the present, but the ordering of search results can be easily obtained. From this point of view, our rank-based aggregation is suitable in this situation since it is exactly good at processing rank lists. Certainly, it is reasonable for us to guess the approximate values of PageRank if we favor the score-based combination, but this topic is out of the scope of this paper. In our methods the original rank lists as inputs can intactly and unbiasedly reflect Google's standpoint.

2.3 Rank Aggregation Methods

We study the problem of combining sets of rank lists from different attributes of user preferences into a single rank list. Voting provides us with a traditional class of algorithms to determine the aggregated rank list. The most common voting theory, named after its creator, is known as Borda's rule [1] which argues that the majority opinion is the truth, or at least the closest that we can come to determining it [13]. However, the problem with Borda's rule is that it does not optimize any criterion. We make use of Footrule distances [3] to weigh edges in a bipartite graph and then find a minimum cost matching. This method was proved in [4] to approximate the optimal ranking that approximately minimizes the number of disagreements with the given inputs.

Modified Borda's Rule. Borda's rule is a single winner election method in which voters rank candidates in order of preferences. The winner of an election is determined by giving each candidate a certain number of points corresponding to the position in which she is ranked by each voter. Once all points have been counted, the candidate with the most points is the winner.

Our idea is that we treat each attribute of user preferences as a voter. It means that each attribute re-orders the search results in the same way as each voter selects a list of candidates. Let $A = a_1, a_2, \cdots, a_m$ be the set of positions in the rank list, and let the attributes of user preferences plus the result list of Google be named by elements of n (i.e., n voters in an election). We shall assume for the present that every element of n can be expressed by a linear order in the position set A. We denote a linear order by a sequence $A_i = a_{i_1}, a_{i_2}, \cdots, a_{i_m}$ where for $j < k$, a_{i_j} is preferred to a_{i_k}. For each voter, the ranked results

should be given some points. The closer a search result is to the top of the list, the more points it will be given. Especially in the context of the Web search, the top search results have much higher possibility to be clicked than others. Most Web search users just browse the top 10 or 20 results. If they do not find the desired information, they will modify their queries to start a new search, instead of continuing checking the results. Therefore, modified Borda's rule is applied here. The voter awards the first-ranked candidate with one point (i.e., 1). The second-ranked candidate receives half of a point (i.e., 1/2), the third-ranked candidate receives on a third (i.e., 1/3), etc. This kind of point distribution gives more weights to the top results. When all elements of n have been counted, and each A_i can be thought of as a position vector, we sort the search results by several formulas, defined as

$$L_1(a_k) = \sum_{i=1}^{n} 1/a_{i_k}\,, \quad L_2(a_k) = \sqrt{\sum_{i=1}^{n} (1/a_{i_k})^2}\,, \tag{6}$$

$$GM(a_k) = (\prod_{i=1}^{n} 1/a_{i_k})^{1/n}\,. \tag{7}$$

Equation 6 represents the L_1 norm and the L_2 norm of these position vectors, and the geometric mean of the n points is expressed in Equation 7. We take into consideration the median of the n points as well. Borda's rule is commonly classified as a positional voting system because from each voter, candidates receive a certain number of points. Computationally it is very easy, as it can be implemented in linear time.

Bipartite Graph. Borda's rule does not assure us that it can find the optimal rank list because it does not optimize any criterion. A graph theory based method is proposed here, to approximate the optimal ranking. We define a weighted balanced bipartite graph $G = (V_1 \cup V_2, W)$. $V_1 = r_1, r_2, \cdots, r_m$ is a set of search results to be ranked. $V_2 = p_1, p_2, \cdots, p_m$ is the m available positions in the rank list. For any two vertices $r \in V_1$ and $p \in V_2$, rp is an edge in G; thus G is also a complete bipartite graph. The weight $W(r, p) =$ is the total distance of a ranking value that places r at position p. The task of rank aggregation is to minimizes the number of disagreements with the respective lists. Therefore, if all the search results are put in proper positions, the total distance (i.e., the number of disagreements) should be the smallest. Now we meet two difficulties in achieving this goal. One is how to compute the distance. The other one is what kind of approaches can minimize the distance.

To weigh the edges in G, according to Diaconis et al. [3], the two distance measures that we consider are:

$$Footrule\_D(\pi, \sigma) = \sum_{i=1}^{n} \mid \pi(i) - \sigma(i) \mid\,,\ Footrule\_S(\pi, \sigma) = \sum_{i=1}^{n} (\pi(i) - \sigma(i))^2\,, \tag{8}$$

where π and σ are regarded as rank lists. Diaconis et al. [3] also suggest two other measures. One roughly seems similar to $Footrule\_D$, and the other is unsuitable for general use, having very small variance about a mean that is very close to its maximum value. Therefore, we choose $Footrule\_D$ and $Footrule\_S$ here. We then adjust the two measures to compute the total distance that is the weight in an edge, now defined as $\sum_i^n \mid A_i(r) - p \mid$ or $\sum_i^n (A_i(r) - p)^2$. Minimizing the total distance to n could be solved by the well-known Hungarian algorithm that finds a minimum cost perfect matching in the bipartite graph. A matching in a graph is a set of edges where no two of which share an endpoint. The most similar work to ours is Dwork et al. [4] who only used $Footrule\_D$ as the distance measure. However, our experiments compared the two measures and observed that $Footrule\_D$ performed the worst among all the methods, even inferior to the score-based method. The largest improvement is reached by $Footrule\_S$. In addition, their main application is to effectively combat *spam* while we study the rank aggregation in terms of user preferences to improve the Web search.

3 Experiments

3.1 Dataset and Evaluation Metrics

Given a query, Google API offered us the top 20 search results. In order to collect the real click-through data, we randomized the order of the results before returning them to 12 invited users and asked them to evaluate whether the clicked results are relevant or not. After the data were collected over a ten-day period (From October 23nd, 2006, to November 1st, 2006), we had a log of about 300 queries averaging 25 queries per subject and about 1200 records of the clicked Web pages in total. The evaluation metrics are listed as follows.

(1) **AvgRank** indicates the average rank of search results, defined as:

$$AvgRank(q) = \sum_{p \in S} R(p)/|S| \,. \tag{9}$$

Here S denotes the set of search results selected by a subject for query q, $R(p)$ is the position of p in the result list, and $|S|$ is the cardinality of the set S. A smaller $AvgRank$ represents a better quality.

(2) **DCG** [5] gives more weight to highly ranked search results, defined as:

$$DCG(i) = \begin{cases} G(1) & \text{if } i = 1 \\ DCG(i-1) + G(i)/log(i) & \text{otherwise} \end{cases} \tag{10}$$

By averaging over a set of test queries, the average performance of our methods can be analyzed. In the experiments, we used $G(i) = 2$ for highly relevant Web pages, $G(i) = 1$ for relevant Web pages, and $G(i) = 0$ for non-relevant search results. A larger DCG means a better quality.

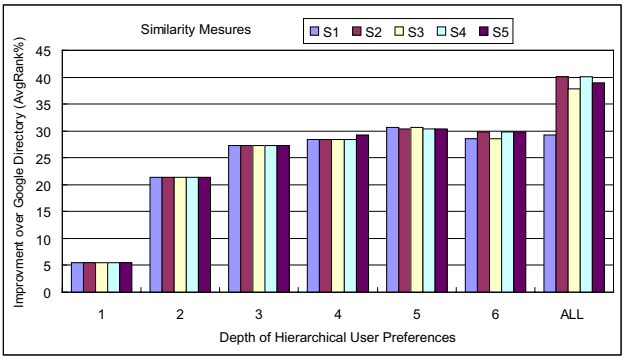

Fig. 2. Depth of Hierarchical User Preferences

3.2 Experimental Results

Depth of Topic Directory for User Preferences. The first step is to determine how deep the depth of topics is when modeling user preferences. We did a preliminary performance analysis on different depths. The re-ranking mechanism is addressed in [7]. Measured by *AvgRank* in Equation 9, Figure 2 illustrates the improvement over Google Directory Search per similarity measure versus the depths considered in learning user preferences.

It shows that the deeper the topic directory we process, the bigger improvement is generally reached. If our algorithm stores the whole topic directory, the biggest improvement is over 40%. In addition, we observed that when the depth is set to 1 (2 or 3), the five similarity measures performed almost the same. The reason is that in our dataset, most of the relevant and non-relevant search results share the same subsumer in a very low depth of the hierarchy. We need to store the deeper topic directory to tell the relevant results from the non-relevant ones. Furthermore, from Figure 2 when the depth of topic directory increases to 3, the improvement is big, from 5% to above 25%. However, when the depth is increased continually from 3 to 6, the improvement changes slowly. Due to this observation and the large size of the whole Google Directory[1], only the top 4 topic directory is encoded into the user preferences in the following experiments, which is a trade-off between accuracy and storage memory.

Effect of Similarity Measures and Rank Aggregation Methods. Figure 3 illustrates the performances of rand aggregation methods and the five similarity measures defined in Equations 1- 5. The *Score-Based* method in the figure is the same as that in [7]. From this figure, the highest improvement over Google Directory Search is about 13%, produced by $Footrule\_S$. L_1 norm, L_2 norm, and $Footrule\_S$ performed better than *Score-Based*, while the qualities of *Median*

[1] Google uses ODP as basis for its Google Directory service, and ODP has more than 590,000 categories.

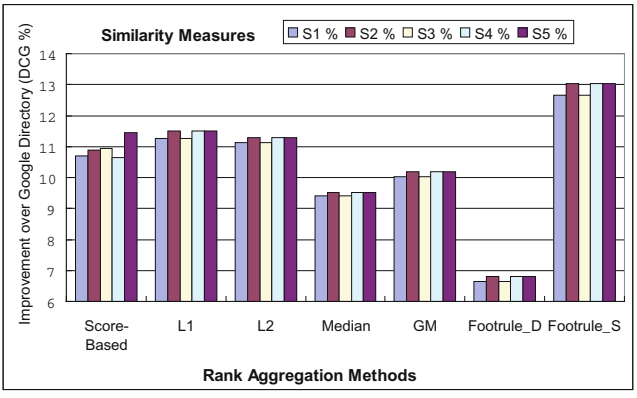

Fig. 3. Effect of Similarity Measures and Rank Aggregation Methods

and *Footrule_D* are inferior to that of *Score-Based*. Although Borda's Rule neither optimizes any criterion nor satisfies the Condorcet property [13], this kind of method outperformed the score-based combination. The Hungarian algorithm based on the Footrule distance that finds a minimum cost perfect matching in the bipartite graph showed the best results obtained by the distance measure *Footrule_S* in Equation 8. In addition, we know that S2, S4, and S5 perform similarly, while S1 and S3 perform similarly as well. The reason is that the former three measures give much more weight on depth than length, and the latter two measures only consider length. Given the same length and depth, the five measures will compute different values due to different transformation functions. Thus the score-based method is easily influenced by the selected function. On the other hand, the rank-based methods are robust. Even the transformation function is different, as long as the measures take into account the same information, they will produce similar performance. Moreover, in the rank-based methods, S2, S4, and S5 performed slightly better than S1 and S3, which tells us that the depth of subsumbers carry more useful information than the naïve distance in our dataset. Note that S4 uses the depth alone, but competes with S2 and S5. In the score-based method, however, S5 is the winner, much better than the other measures.

4 Conclusions

In this paper we proposed a set of techniques for rank aggregation. Experimental results on a real click-through data set demonstrate the effectiveness of our methods. We observed that some rank-based aggregation methods performed better than the *Socre-Based* method and the *Footrule_S* method performed best in our evaluation. Furthermore, we analyzed the influence of the topic depth of the ontology-based user preferences on the quality of the Web search, and compared the performances of five similarity measures. If the measures utilize similar

information from users, they will perform similarly regardless of what kind of transformation functions is being used. But the score-based combination is sensitive to the selected function. In the future we plan to put these methods into larger datasets, and further mine more user-centered information and optimize Web searches in terms of user's satisfaction.

References

[1] Borda, J.: Mémoire sur les élections au scrutin. Comptes rendus de l'Académie des sciences 44, 42–51 (1781)
[2] Chirita, P.A., Nejdl, W., Paiu, R., Kohlschütter, C.: Using ODP metadata to personalize search. In: Proc. of SIGIR 2005, Salvador, Brazil, pp. 178–185 (2005)
[3] Diaconis, P., Graham, R.L.: Spearman's footrule as a measure of disarray. Journal of the Royal Statistical Society. Series B (Methodological) 39(2), 262–268 (1977)
[4] Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank aggregation methods for the web. In: Proc. of WWW 2001, Hong Kong, China, pp. 613–622 (2001)
[5] Jarvelin, K., Kekalainen, J.: IR evaluation methods for retrieving highly relevant documents. In: Proc. of SIGIR 2000, Athens, Greece, pp. 41–48 (2000)
[6] Lawrence, S.: Context in web search. IEEE Data Eng. Bull. 23(3), 25–32 (2000)
[7] Li, L., Yang, Z., Wang, B., Kitsuregawa, M.: Dynamic adaptation strategies for long-term and short-term user profile to personalize search. In: Proc. of AP-Web/WAIM 2007, Huang Shan, China, pp. 228–240 (2007)
[8] Li, Y., Bandar, Z., McLean, D.: An approach for measuring semantic similarity between words using multiple information sources. IEEE Trans. Knowl. Data Eng. 15(4), 871–882 (2003)
[9] Renda, M.E., Straccia, U.: Web metasearch: Rank vs. score based rank aggregation methods. In: Proc. of SAC 2003, Melbourne, USA, pp. 841–846 (2003)
[10] Schickel-Zuber, V., Faltings, B.: Inferring user's preferences using ontologies. In: Proc. of AAAI 2006, Boston, Massachusetts, USA (2006)
[11] Shen, X., Tan, B., Zhai, C.: Implicit user modeling for personalized search. In: Proc. of CIKM 2005, Bremen, Germany, pp. 824–831 (2005)
[12] Speretta, M., Gauch, S.: Personalized search based on user search histories. In: Proc. of WI 2005, Compiegne, France, pp. 622–628 (2005)
[13] Young, H.P.: Condorcet's theory of voting. American Political Science Review 82(4), 1231–1244 (1988)
[14] Zhu, S., Fang, Q., Deng, X., Zheng, W.: Metasearch via voting. In: Liu, J., Cheung, Y.-m., Yin, H. (eds.) IDEAL 2003. LNCS, vol. 2690, pp. 734–741. Springer, Heidelberg (2003)