IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Vol. 04, Issue 08 (August. 2014), ||V3|| PP 57-59

Evaluating and Analyzing Query Efficiencies along Web Wide Databases

Sonia Gupta
Computer Science Department Iftm University Moradabad
Bharat Bhushan Agarwal Computer Science Department Iftm University Moradabad

Abstract: - The growth of the Web and the Internet leads to the development of an ever increasing number of interesting application classes. Normal recruitment process is now-a-days used most common method. If a company wants an employee immediately, the only way for recruitment is advertising in any media. When the appliction is received from the employee it had to be checked for experience, qualifiction etc. This process requires time.
This paper proposes a method for employee searching by using a user and query dependent ranking. A ranking model is presented which is based on user inputs. This ranking model is acquired from several other ranking functions derived for various user-query pairs. This is based on the intuition that similar users display comparable ranking preferences over the result of similar queries. The idea of how the ranking can be used is provided by this paper.

Keywords: *User Similarity, Query Similarity, Automatic Ranking, Workload, Relational Queries.*

I. INTRODUCTION

The success and growth of the Internet and Web leads to the development of a large number of Web databases for a variety of applications. Only a Boolean query modelis supported by database systems. If query is not selective then too many tuples may be in the answer. It is time consuming to select the most appropriate answer .Web databases simplify this task by sorting the query result. Currently this sorting is done on the values of a single attribute. The ordering based on multiple attribute values would be closer to the Web user's expectation. A simple Boolean query retrieval model is supported by these database, it often leads to situations when too many results are generated in response to a query. In order to find the results that match one's interest, the user has to browse through this large result set – a tedious and time-consuming task. Currently, Web databases simplify this task by displaying these results in a sorted order on the values of a single attribute (e.g., Price, Mileage, etc.). Relevance measurement is crucial to web search and to information retrieval in general. Traditionally, search relevance is measured by using human assessors to judge the relevance of query-document pairs. However, explicit human ratings are expensive and difficult to obtain. Millions of people interact daily with web search engines at the same time, providing valuable implicit feedback through their interactions with the search results. If we could turn these interactions into relevance judgments, we could obtain large amounts of data for evaluating, maintaining, and improving information retrieval systems.

II. PROBLEM DEFINITION AND ARCHITECTURE

The ranking problem can be stated as: "For the query Q_j given by the user U_i, find out a ranking function FU_iQ_j from W". Following are the types of ranking problem:
1. Identifying a ranking function using the similarity model: Given W, determine a user U_x similar to U_i and a query Q_y similar to Q_j such that the function FU_xQ_{ys} exists in W.
2. Generating a workload of ranking functions: Given a user U_x asking query Q_y, based on U_x's preferences towards Q_y's results, determine, explicitly or implicitly, a ranking function FU_xQ_y. W is then established as a collection of such ranking functions learnt over different user-query pairs.

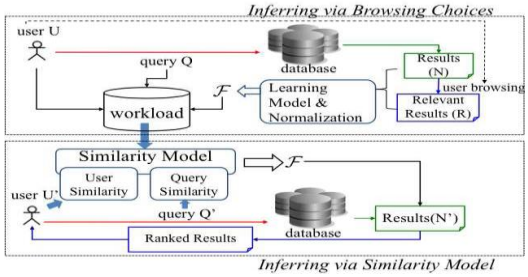

Fig 1: User & Query Dependent Ranking Architecture

III. RANKING ARCHITECTURE

The architecture for our *user*- and *query*-dependent ranking framework (shown in Figure 1) comprises of two components for addressing the subproblems defined above.

In the first component (*Inferring via Browsing Choices*), the user's (U) browsing choices over the query results (N) produces the set of relevant results (R). Both these sets (N and R) are fed to the *Learning Model* that deduces the –

i) significance of each attribute to establish the set of *attribute-weights*, and ii) emphasis given by users to particular values of an attribute. Taking in consideration the preferences associated with the values, a *normalization* scheme translates the values into their corresponding *value-scores*. The attribute-weights and the value-scores then integrate into the ranking function F that assigns a score to every tuple t in N (given by Equation 1):

score(t) = F(t) (1)

F is termed as the inferred ranking function along with the user (U) and the query (Q) are loaded into the workload that forms the nucleus over which the second component of our framework is built. At the time of query (Q′), the user's (U′) browsing choices may not always be available and hence the ranking function ($F_{U'Q'}$) cannot be derived. Address this, the second component (*Inferring via Similarity Model*) is formulated for deriving an appropriate ranking function (to rank Q′'s results for U′). This model takes as input, the user-query combination and determines, from the workload, a ranking function derived for the most similar query (to Q′) given by the most similar user (to U′). This is achieved using the two individual metrics of – *user similarity* and *query similarity*. Based on the function derived, Q′'s results are ranked using Equation 1 and displayed to the user (U′).

IV. SIMILARITY MODEL FOR QUERY RANKING

When ranking functions are known for a small set of user-query pairs, then the concept of similarity-based ranking is aimed. At the time of answering a query asked by a user, if no ranking function is available for this user-query pair, the proposed query and user-similarity models can effectively identify a suitable function to rank the corresponding results.

V. QUERY SIMILARITY

For the user U1 from *Example-1*, a ranking function does not exist for ranking Q1's results (N1). However, from *Example-2*, it is known that a user is likely to have displayed different ranking preferences for different query results. A randomly selected function from U1's workload is not likely to give a desirable ranking order over N1. Meanwhile, the ranking functions are likely to be comparable for queries similar to each other [2].

The proposed paper advances the hypothesis that if Q1 is most similar to query Qy (in U1's workload), U1 would display similar ranking preferences over the results of both queries; thus, the ranking function (F1y) derived for Qy can be used to rank N1. Similar to recommendation systems, this framework can utilize the aggregate function, composed from the functions corresponding to the *top-k* most similar queries to Q1, to rank N1 [2]. This proposal of *query similarity* into two alternative models: i) *query condition* similarity, and ii) *query-result* similarity.

Query condition similarity: By comparing the attribute values in the query conditions, the similarity between two queries can be determined. Given two queries Q and Q', each with the conjunctive selection conditions, respectively of the form "WHERE $A_1=a_1$ AND • • • AND Am=am" and "WHERE $A_1=a_1$' AND • • • AND Am=am'.

$$\text{Similarity}(Q, Q') = \prod_{i=1}^{m} \text{sim}(Q[A_i = a_i], Q'[A_i = a_i']) \quad (2)$$

Query result similarity: If two queries are similar, the results are likely to greater similarity. Similarity between a pair of queries is evaluated as the similarity between the tuples in the respective query results. Given two queries Q and Q', let their query result be N and N' . The similarity of query- result between Q and Q' is then computed as the similarity between the result sets N and N', given by Equation 2.

similarity(Q,Q') =sim(N,N') (3)

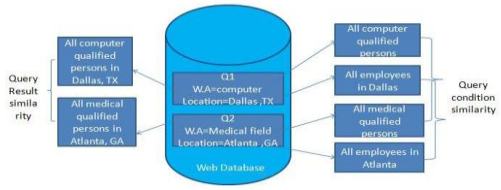

Fig 2: Query similarity model summarized view

The above figure shows the computation of similarity for the two models.

VI. USER SIMILARITY

It is known from Example-1 that different users may display different ranking preferences towards the same query. Here put forward the hypothesis that if U_1 is similar to an existing user U_x, then, for the results of a given query (say Q_1), both users will show similar ranking preferences; therefore, U_x's ranking function (F_{x1}) can be used to rank Q_1's results for U_1 as well. Given two users U_i and U_j with the set of common queries – {Q_1, Q_2, ..., Q_r}, for which ranking functions ({F_{i1}, F_{i2}, ..., F_{ir}} and {F_{j1}, F_{j2}, ..., F_{jr}}) exist in W, the user similarity between U_i and U_j is expressed as the average similarity between their individual ranking functions for each query Q_p (shown in Equation 3):

$$\text{Similarity}(U_i, U_j) = \frac{\sum_r sim(F_{ip}F_{jp})}{r} \qquad (4)$$

VII. CONCLUSION

This paper proposes a model for employee searching by using a user- and query-dependent ranking method. By using this method, it solves the Many-Answers Problem which leverages data and workload statistics and correlations. The design and maintenance of an appropriate workload that satisfies properties of similarity-based ranking is very challenging.

REFRENCES

[1] Google. Google base. http://www.google.com/base.
[2] AdityaTelang, Chengkai Li, Sharma Chakravarthy, "One size Does Not Fit All: Towards User- and Query Dependent Rasnking For Web Databases".
[3] G. Koutrika and Y. E. Ioannidis. Constrained optimalities in query personalization. In SIGMOD Conference, pages 73–84, 2005.
[4] S. Amer-Yahia, A. Galland, J. Stoyanovich, and C. Yu. From del.icio.us to x.qui.site: recommendations in social tagging sites. In SIGMOD Conference, pages 1323–1326, 2008.
[5] A. Penev and R. K. Wong. Finding similar pages in a social tagging repository. In WWW, pages 1091–1092, 2008
[6] T. C. Zhou, H. Ma, M. R. Lyu, and I. King. Userrec: A user recommendation framework in social tagging systems. In *AAAI*, 2010.
[7] B. He. Relevance feedback. In Encyclopedia of Database Systems, pages 2378–2379, 2009.
[8] Y. Rui, T. S. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in mars. In IEEE International Conference on Image Processing, pages 815–818, 1997.
[9] L. Wu and C. F. et. al. Falcon: Feedback adaptive loop for content-based retrieval. In VLDB, pages 297–306, 2000.
[10] Google. Google base. http://www.google.com/base.
[11] A. Marian, N. Bruno, and L. Gravano. Evaluating top-k queries over web-accessible databases. ACM Transactions of Database Systems, 29(2):319–362,2004.
[12] S. Gauch and M. S. et. al. User profiles for personalized information access. In Adaptive Web, pages 54–89, 2007.
[13] A. Penev and R. K. Wong. Finding similar pages in a social tagging repository. In WWW, pages 1091–1092, 2008.
[14] T. C. Zhou, H. Ma, M. R. Lyu, and I. King.Userrec: A user recommendation framework in social tagging systems. In AAAI, 2010.
[15] H. Yu, S.-w.Hwang, and K. C.-C. Chang. Enabling soft queries for data retrieval. Information Systems, 32(4):560–574, 2007.
[16] A. Telang, C. Li, and S. Chakravarthy. One size does not fit all: Towardsuser- and query-dependent ranking for web databases. Technical report,UT Arlington, ttp://cse.uta.edu/research/Publications/Downloads/CSE-2009-6.pdf,2009.