ISSN:0975-9646

K. Padmaja et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (5) , 2015, 4292-4295

Dynamic Query Evaluation using Proximity Ranking

K. Padmaja[#1], B. Prashant[#2],

[#1] *M.Tech (CSE), Eluru College of Engineering & Technology, Eluru, India.*

[#2] *Associate Professor, CSE, Eluru College of Engineering & Technology, Eluru, India.*

Abstract: Search Engines has always been the chosen mode of information retrieval (IR) systems. Users are no longer content with issuing simple navigational queries. A complex query such as travel arrangement has to be broken down into a number of co-dependent steps over a period of time. For instance, a user may first search on possible destinations, timeline, events, etc. After deciding when and where to go, the user may then search for the most suitable arrangements for air tickets, rental cars, lodging, meals, etc. Each step requires one or more queries, and each query results in one or more clicks on relevant pages. Keyword based search engines cannot support this kind of hierarchical queries. So we propose to use Random walk propagation methods that construct user profile based on his credentials from its user search history repositories. Combined with click points driven click graphs of user search behaviors the IR system can support complex queries for future requests at reduced times. Random walk propagation over the query fusion graph methods support complex search quests in IR systems at reduced times. For making the IR Systems effective and dynamic we also propose to use these search quests as auto complete features in similar query propagations. Biasing the ranking of search results can also be provided using any ranking algorithms (top-k algorithms).Supporting these methods yields dynamic performance in IR systems, by providing enriched user querying experience. A practical implementation of the proposed system validates our claim.

Index Terms: Query Clustering, Search Engine, Query Reformulation, Click Graph, Task Identification.

I. INTRODUCTION

AS the size and richness of information on the Web grows, so does the variety and the complexity of tasks that users try to accomplish online. Users are no longer content with issuing simple navigational queries. Various studies on query logs (e.g., Yahoo's and AltaVista's reveal that only about 20% of queries are navigational. The rest are informational or transactional in nature. This is because users now pursue much broader informational and task-oriented goals such as arranging for future travel, managing their finances, or planning their purchase decisions. However, the primary means of accessing information online is still through keyword queries to a search engine.

To improve user's search experience, most major commercial search engines provide query suggestions to help users formulate more effective queries. When a user submits a query, a list of terms that are semantically related to the submitted query is provided to help the user identify terms that he/she really wants, hence improving the retrieval effectiveness. Yahoo's "Also Try" and Google's "Searches related to" features provide related queries for Narrowing search, while Ask Jeeves suggests both more specific and more general queries to the user. One important step towards enabling services and features that can help users during their complex search quests online is the capability to identify and group related queries together. Recently, some of the major search engines have introduced a new "Search History" feature, which allows users to track their online searches by recording their queries and clicks. This history includes a sequence of four queries displayed in reverse Chronological order together with their corresponding clicks. In addition to viewing their search history, users can manipulate it by manually editing and organizing related queries and clicks into groups, or by sharing them with their friends.

Once query groups have been identified, search engines can have a good representation of the search context behind the current query using queries and clicks in the corresponding query group. For example, if a search engine knows that a current query "financial statement" belongs to a {"bank of America", "financial statement"} query group, it can boost the rank of the page that provides information about how to get a Bank of America statement instead of the Wikipedia article on "financial statement", or the pages related to financial statements from other banks.

Your recent history
1: Apple 1(www.apple.com)
Description: Apple is a fruit, this is able to perform relevant description.
2: Apple Ipod(www.appleipod.com)
Description: It is used to provide ipod services

Figure 1: Example of search history feature in Bing.

In this paper we motivate and propose a method to perform query grouping in a dynamic fashion. Our goal is to ensure good performance while avoiding disruption of existing user-defined query groups. We investigate how signals from search logs such as query reformulations and clicks can be used together to determine the relevance among query groups. We study two potential ways of using clicks in order to enhance this process: by fusing the query reformulation graph and the query click graph into a single graph that we refer to as the *query fusion graph*, and by expanding the query set when computing relevance to also include other queries with similar clicked URLs.

II. RELATED WORK

Baeza-Yates et al proposed a query clustering method that groups similar queries according to their semantics. The method creates a vector representation Q or a query q, and the vector Q is composed of terms from the clicked documents of q. Cosine similarity is applied to the query vectors to discover similar queries. More recently, Zhang and Nasraoui presented a method that discovers similar queries by analyzing users' sequential search behavior. The method assumes that consecutive queries submitted by a user are related to each other. The sequential search behavior is combined with a traditional contentbased similarity method to compensate for the high sparsity of real query log data.

Time	Query
10:51:45	Saturn Value
10:54:27	Hybrid Saturn value description
11:21:07	Will GameStop
12:22:22	Sprint Latest Model

Figure 2: User time results based on searching process.

Figure 3: User processing results with semantic group results.

Our goal is to automatically organize a user's search history into query groups, each containing one or more related queries and their corresponding clicks. Each query group corresponds to an atomic information need that may require a small number of queries and clicks related to the same search goal. For example, in the case of navigational queries, a query group may involve as few as one query. One major problem with the click through-based method is that the number of common clicks on URLs for different queries are limited. This is because different queries will likely retrieve very different result sets in very different ranking orders.

Dynamic Query Grouping: One approach to the identification of query groups is to first treat every query in a user's history as a singleton query group, and then merge these singleton query groups in an iterative fashion (in a k-means or agglomerative way [8]). However, this is impractical in our scenario for two reasons. First, it may have the undesirable effect of changing a user's existing query groups, potentially undoing the user's own manual efforts in organizing her history. Second, it involves a high computational cost, since we would have to repeat a large number of query group similarity computations for every new query.

III. PRELIMINARIES

Our objective is to instantly arrange a user's look for record into question categories, each containing one or more appropriate concerns and their corresponding mouse clicks. Each question team matches to an nuclear information need that may require some concerns and mouse clicks appropriate to the same look for objective. For example, in the case of navigational concerns, a question team may involve as few as one question and one click (e.g., "cnn" and www.cnn.com).

Query Group: A question team is an or- dered list of concerns, qi, together with the corresponding set of visited URLs, clki of qi. A question team is denoted as s = h{q1, clk1}, . . . , {qk, clkk}i.

The specific ingredients of our problem is as follows:

Given: a set of present question categories of a user, S = {s1, s2, . . . , sn}, and her present question and mouse clicks, {qc, clkc}, Find: the question team for {qc, clkc}, which is either one of the present question categories in S that is most appropriate to, or a new question team sc = {qc, clkc} if there does not are available a question team in S that is not completely appropriate to {qc, clkc}.

Below, we will encourage the powerful characteristics of this ingredients, and give an introduction to the remedy. The primary of the remedy is a evaluate of importance between two concerns (or question groups). We will further encourage the need to go beyond guideline importance actions that depend promptly or written text, and instead recommend a importance evaluate based on alerts from look for records. Dynamic Query Collection. One strategy to the recognition of question categories is to first cure every question in a user's record as a singleton question team, and then combine these singleton question categories in an repetitive fashion (in a k-means or agglomerative way). However, this is incorrect in our situation for two reasons. First, it may have the unwanted effect of modifying a user's present question categories, possibly undoing the user's own guide initiatives in planning her record. Second, it includes a high computational cost, since we would have to do it again a huge variety of question team likeness calculations for every new question.

Query (or Query Group) Relevance: To ensure that each question team contains carefully appropriate and appropriate concerns and mouse clicks, it is important to have a appropriate importance evaluate sim between the present question singleton team sc and an present question team si 2 S. There are a variety of possible techniques to determine the importance between sc and si. Below, we summarize a variety of different importance analytics that we will later use as baselines in tests. We will also talk about the benefits and drawbacks of such analytics as well as our suggested strategy of using look for records. Time. One may believe that sc and si are somehow appropriate if the concerns appear close to each other soon enough in the user's record. In other words, we believe that customers generally issue very similar concerns and clicks within a few months frame.

IV. PROBLEM DEFINATION

Personalized Concept-Based Clustering: We now explain the essential idea of our personalized concept-based clustering algorithm with which ambiguous queries can be clustered into different query clusters. Personalized effect is achieved by manipulating the user concept preference profiles in the clustering process. In contrast to BB's agglomerative clustering algorithm, which represents the same queries submitted from different users by one query node, we need to consider the same queries submitted by different users separately to achieve personalization effect. In other words, if two given queries, whether they are identical or not, mean different things to two different users, they should not be merged together because they refer to two different sets of concepts for the two users. Therefore, we treat each individual query submitted by each user as an individual vertex in the bipartite graph by labeling each query with a user identifier.

V. SYSTEM ANALYSIS

A user queries a search engine Search Engine tries to construct user profile based on his ipaddress/login credentials from its user search history repositories. If the user already exists, the search engine checks from its user search history repositories up to a certain threshold whether the user already queried the same query previously If the user did, then search engine further retrieves click points from user search history repositories and reformulates query results by generating click graphs. Click graphs contain useful information on user behavior when searching online. This step is called query fusion graph. Uses random walk propagation over the query fusion graph instead of time-based and keyword similarity based approaches. This entire process is called organizing user search histories into query groups. This approach helps users to pursue complex search quests online.

VI. QUERY REKEVANCE USING SEARCH LOGS

We now develop the machinery to define the *query relevance* based on Web search logs. Our measure of relevance is aimed at capturing two important properties of relevant queries, namely: (1) queries that frequently appear together as reformulations and (2) queries that have induced the users to click on similar sets of pages

Find the relevance
Input: QFG, factor, given query, q.
Output: Relevance vector for given query.
Step 1: Initially rel=0
Step 2: random walk propagation, number of visits.
Step 3: for each user processing results are displayed based on numVisits
Step 4: above two steps are repeated to every user processing in search process.

Figure 4: Algorithm for calculating the query relevance by simulating random walks over the query fusion graph.

6.1 Search Behavior Graphs

We derive three types of graphs from the search logs of a commercial search engine. The *query reformulation graph*, QRG, represents the relationship between a pair of queries that are likely reformulations of each other. The *query click graph*, QCG, represents the relationship between two queries that frequently lead to clicks on similar URLs.

Query Reformulation Graph: One way to identify relevant queries is to consider *query reformulations* that are typically found within the query logs of a search engine. If two queries that are issued consecutively by many users occur frequently enough, they are likely to be reformulations of each other.

Query Click Graph: A different way to capture relevant queries from the search logs is to consider queries that are likely to induce users to click frequently on the same set of URLs. For example, although the queries “ipod” and “apple store” do not share any text or appear temporally close in a user's search history, they are relevant because they are likely to have resulted in clicks about the iPod product.

Query Fusion Graph: The query reformulation graph, QRG, and the query click graph, QCG, capture two important properties of relevant queries respectively.

VII. QUERY GROUPING USING THE QFG

In this area, we summarize our suggested likeness operate simrel to be used in the on the internet question collection procedure described. we sustain a question picture, which symbolizes the importance of other concerns to this question. For each question team, we sustain a perspective vector, which aggregates the images of its member concerns to form an overall reflection. We then recommend a likeness operate simrel for two question categories based on these ideas of perspective vectors and question pictures. Note that our suggested explanations of query reformulation chart, question pictures, and perspective vectors are crucial components, which offer significant unique to the Markov sequence procedure for identifying importance between concerns and question categories.

Context Vector: For each question team, we sustain a perspective vector which is used to calculate the likeness between the question team and the user's latest singleton question team. The perspective vector for a question team s, denoted cxts, contains the importance ratings of each question in VQ to the question team s, and is obtained by aggregating the combination importance vectors of the concerns and mouse clicks in s. If s is a singleton question team containing only {qs1 , clks1}, it is determined as the combination importance vector rel(qs1,clks1). For a question team s = h{qs1 , clks1}, . . . , {qsk , clksk}i with k > 1, there are a number of different ways to determine cxts. For example, we can determine it as the combination importance vector of the most recently added question and mouse clicks, rel(qsk ,clksk). Other opportunities include the average or the calculated sum of all the combination importance vectors of the concerns and mouse clicks in the question team.

Query Image: The combination importance vector of a given question q, relq, catches the degree of importance of each question q0 2 VQ to q. However, we observed that it is not effective or effective to use relq itself as a importance measure for our on the internet question collection. For example, let us consider two relevant concerns, “financial statement” (“fs”) and “bank of america” (“boa”).

Some programs such as question recommendation may be assisted by fast on-thefly collection of user concerns. For such programs, we can avoid performing the unique walk calculations of combination importance vector for every new question in real-time, and instead pre-compute and storage cache these vectors for some concerns in our chart. This works especially well for the popular concerns. In this case, we are basically trading off hard drive storage area for run-time performance. We calculate that to storage cache the combination importance vectors of 100 million concerns, we would require hard drive storage area space in the hundreds of gb. This additional storage area space is unimportant comparative to the overall storage area requirement of a on the internet search engine.

VIII. SYSTEM IMPLEMENTATION

Experimental Setup:

We study the behavior and performance of our algorithms on partitioning a user's query history into one or more groups of related queries. For example, for the sequence of queries "Caribbean cruise"; "bank of America"; "expedient"; "financial statement", we would expect two output partitions: first, {"Caribbean cruise", "expedia"} pertaining to travel-related queries, and, second, {"bank of America", "financial statement"} pertaining to money-related queries.

Using Search Logs

Our query grouping algorithm relies heavily on the use of search logs in two ways: first, to construct the query fusion graph used in computing query relevance, and, second, to expand the set of queries considered when computing query relevance. We start our experimental evaluation, by investigating how we can make the most out of the search logs. In our first experiment, we study *how we should combine* the query graphs coming from the query reformulations and the clicks within our query log.

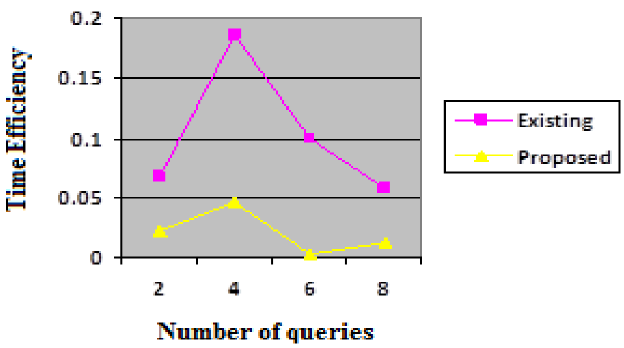

Figure 5: Varying query results in both existing and proposed approaches.

Above graph describes the horizontal axis represents _ (i.e., how much weight we give to the query edges coming from the query reformulation graph), while the vertical axis shows the performance of our algorithm in terms of the RandIndex metric.

IX. CONCLUSION

The Query formulations based on click graphs contain useful information on user behavior when searching online. For this process we are using different informative techniques like page rank operations for analyzing the user histories. In this paper we propose to develop the efficient data extraction based on click graph results. We also find value in combining our method with keyword similarity-based methods, especially when there is insufficient usage information about the queries. As future work, we intend to investigate the usefulness of the knowledge gained from these query groups in various applications such as providing query suggestions and biasing the ranking of search results.

REFERENCES

[1] Shiwen Cheng, Arash Termehchy, and Vagelis Hristidis "Efficient Prediction of Difficult Keyword Queries over Databases" in proc. 2014
[2] http://www.dmoz.org/, 2008.
[3] http://www.google.com/, 2008.
[4] http://www.sigkdd.org/kdd2005/kddcup.html, 2008.
[5] D. Beeferman and A. Berger, "Agglomerative Clustering of a Search Engine Query Log," Proc. ACM SIGKDD, 2000.
[6] R. Jones and K. L. Klinkner, "Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs," in *CIKM*, 2008.
[7] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna, "The query-flow graph: Model and applications," in *CIKM*, 2008.
[8] R. Baeza-Yates and A. Tiberi, "Extracting semantic relations from query logs," in *KDD*, 2007.
[9] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.
[10] P. Anick, "Using terminological feedback for web search refinement: A log-based study," in *SIGIR*, 2003.
[11] B. J. Jansen, A. Spink, C. Blakely, and S. Koshman, "Defining a session on Web search engines: Research articles," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 6, pp. 862–871, 2007.

AUTHORS

K.Padmaja received her B.Tech degree in CSE from JNTUH SCDE, Hyderabad, India in 2011. Presently, she is pursuing M.tech in CSE from ELURU College of Engineering and technology, Eluru, West Godavari Dt, Andhra Pradesh, India. Her research interests are data mining and Network Security.

B.Prashant received his B.Tech Degree in EEE from Bapatla Engineering College, Bapatla, Guntur(Dt), Andhra Pradesh, India in 2002, M.Tech Degree in CSE from Nova College of Engineering and Technology, Jangareddygudem, West Godavari Dt, Andhra Pradesh, India in 2011. He has 10 years of experience in teaching. Currently he is working as Associate Professor in Department of CSE in Eluru College of Engineering and Technology, Eluru, West Godavari Dt, Andhra Pradesh, India. His areas of research interests are Artificial Intelligence and Neural Networks, Computer Organisation, Data Mining and Data Base Management Systems