ISSN 2250-3153

Query Recommendations for Interactive Database Exploration

Rajendra Desale ,Vaibhav Patil , Tushar Aware

Shatabdi Institute Of Engineering, Agaskhind, Computer Department

Abstract- Traditional DBSMs are suited for applications in which the structure, meaning andcontents of the database, as well as the questions to be asked are already well understood. There is, however, a class of applications that we will collectively refer to as Interactive Data Exploration (IDE) applications, in which this is not the case. IDE is akey ingredient of a diverse set of discovery-oriented applications we are dealing with, including ones from scientific computing, financial analysis, evidence-based medicine, and genomics. The need for effective IDE will only increase as data are being collected at anunprecedented rate. IDE is fundamentally a multi-step, non-linear process with imprecise end-goals. For example, data-driven scientific discovery through IDE often requires non-expert users to iteratively interact with the system to make sense of and to identifyinteresting patterns and relationships in large, amorphous data sets.

To make the most of the increasingly available complex and big data sets, users would need an expert assistant. who would be able to effectively and efficiently guidethem through the data space. Having a human assistant is not only expensive but alsounrealistic. Thus, it is essential that we automate this task. We propose database sys-terms be augmented with an automated database navigator (DBN) service that assistsas a tour guide to facilitate IDE. Just like a car navigation system that offers adviceon the routes to be taken and display points of interest, DBN would similarly steerthe user towards interesting trajectories through the data, while highlighting relevantfeatures. Like any good tour guide, DBN should consider many kinds of information; inparticular, it should be sensitive to a user's goals and interests, as well as common navigation patterns that applications exhibit. We sketch a general data navigation framework and discuss some specific components and approaches that we believe belong to any such system.

I. INTRODUCTION

System Overview

A . Query Fragmentation

The SQL syntax checker checks if the given input query is in SQL syntax. It also checks ifthe given _elds match with that of the tables in the database and veri_es the attributes inthe database table. And then the input SQL query (IQ) is fed to the query fragmentationalgorithm. The given query is split into fragments with respect to the keywords (select,from, where, group by, having, order by). Names are given for the fragmented queries.The fragmented query attributes are stored in the fragment table(Ft) with respect to thefragment name.

B.Query Filter

The query limit (QL) is set to n. The active users query (IQ) is fragmented using theFQ algorithm. The query q is compared with the already recorded fragments in querylog (stack table). If the queries match, the query rating is incremented by 1. If thequeries dont match, the new fragments are updated in the query log. This is done till the number entries in the query log are within the query limit. If the number of entriesis to exceed the query limit, the query log is full. The query is then removed from thequery log accounting to an LRU policy.

C.Query Suggestion Engine

The query suggestion engine gives a set of recommended queries SQ for the given inputSQL query (IQ). The input query IQ is first fragmented and the fragmented query FQis stored in a table t. The query Pro_le QP from the query rating contains fragmentedid and rank of the queries. The fragmented query is compared with the queries in thequery pro_le. If the fragmented query matches with any of the queries in the querypro_le, the rank of the queries in the query pro_le is checked. The top n rank queries arereturned as SQ. If the FQ does not match with any of the queries in the query pro_le, theresult of the input query IQ is returned. Here we consider two di_erent waiting schemes,a binary scheme and a weighted scheme. Both using the queries Q posed by user i. Inbinary scheme all participating fragments receive the same importance weight, regardlessof whether they appear in many queries in the session or only one. In weighted schemefragments that appear more than once in a user session will receive higherweight thanothers. The fragment-based instantiation of the QueRIE framework works in a similarmanner to the tuple based .

II. WHAT IS RECOMMENDATION SYSTEM ?

Recommender systems or recommendation systems are a subclass of information_ltering system that seek to predict the rating or preference that user would give toan item. Recommender systems have become extremely common in recent years, andare applied in a variety of applications. The most popular ones are probably movies,music, news, books, research articles, search queries, social tags, and products in general.Recommender systems typically produce a list of recommendations in one of two ways through collaborative or content based _ltering. Collaborative _ltering approaches builda model from a users past behaviour as well as similar decisions made by others: thenusethat model to predict items that the user may have an interest in. Content based _lteringapproaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. Recommender system is an active research areain the data mining and machine learning areas.

III. Related Works

A multidimensional query recommendation system is proposed in [4]. In Contextual Database Preferences it suggest that context may express conditions on situations external to the database or related to the data stored in the database. It outlines modelsfor expressing both types of preferences. Then, given a user query and its surrounding context suggests that the size of datasets being collected and analyzed in the industry for business intelligence is growing rapidly, making traditional warehousing solutionsprohibitively expensive. In this paper , we provide a comprehensive presentation ofQUERIE, including an overview of previous work (tuple based instantiation).

System Architecture For Query Recommendation System

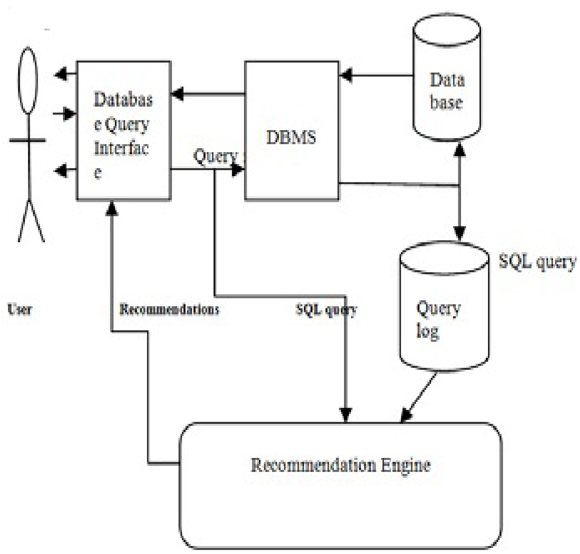

Figure 1.1: System Architecture

IV. Recommendation Algorithms

The queries of each user touch a subset of the database that is relevant to the analysis the user wants to perform. We assume that this subset is modeled as a sessionsummary Si for user i. We use 1, . . . , h to denote the set of past users based on whichrecommendatio-ns are generated and 0 to identify the current user. To generate recommendations, our framework extends the summary S0 of the active user to a predictedsummary Spred 0 . This extended summary captures the predicted degree of interest ofthe active user with respect to all the parts of the database, including those that the user has not explored yet, and thus serves as the seed for the generation of recommendations.To summarize, our framework consists of three components: (a) the construction of asession summary Si for each user i, (b) the computation of a predicted summary Spred 0 for the active user, based on the active users and the past users summaries, and (c) thegeneration of queries based on Spred 0 . Those queries will be presented to the user asrecommendations. The details of each step differ for each recommendation engine. Weprovide a brief overview of both approaches in what follows.

V. Tuple-based Recommendations

Session summaries :-

We define the session summary Si as a vector of tuple weightsthat covers all the database tuples. The weightof each vector element represents the importance of the respective tuple in the exploration performed by user i. For this purpose we employ two different weighting schemes which are detailed in theaccompanying paper

[1]. Using the session summaries of the past users, we can define the conceptual session-tuple matrix that, as in the case of the user-item matrix in web recommender systems,will be used as input in our collaborative filtering process. Computing the predicted summary. Similarly to session summaries, the extended summary Spred 0 is a vector of tuple weights. In order to compute this summary, we assume the existence of a functionsim(Si, Sj) that measures the similarity between two summaries and takes values in [0,1]. Using this function, we compute the extended summary as a weighted sum of theexisting summaries: Spred 0 = P0h(sim(S0, Si)Si). The similarity function sim can berealized with any vector-based metric, such as the cosine similarity measure. Generating recommendations. The final step is to generate queries that cover the interesting tuples in Spred 0 . In order to provide the users with intuitive, easily understandable recommendations, we use the queries of past users. We assign to each past query Q animportance with respect to Spred 0 , computed as rank(Q, Spred0) = sim(SQ, Spred 0). Hence, a query has high rank if it covers the important tuples in Spred 0 . The top ranked queriesare then returned as the recommendation. Accelerating the online computations. To ensure that the aforementioned approach generates real-time recommendations for the active users of a database, we need to compress the session-tuple matrix and to speedup the computation of similarities. For thisreason, we employ the MinHash probabilistic clustering techniquethat maps each session summary Si to a signature h(Si)

[2]. The Jaccard similarity between vectors is thus reduced to the similarity of their signatures:JaccardSim(Si, S0) = sim(h(Si), h(S0)).

Fragment-based recommendations

Session summaries :-

This approach is based on the pair-wise similarity of query fragments (attributes, tables, joins and predicates). We need to identify fragments thatco-appear in several queries posed by different users. The session summary vector Sifor a user i consists of all the query fragments of the users past queries. Let Qi represent the set of queries posed by i and F represent the set of distinct query fragmentsrecorded in the query logs. For a given fragment 2 F, its importance in session Si isrepresented by Si[] and depends on its importance in the session. We can define SQ[] asa weighted or binary variable that represents the importance of in a sessions query Q.

Then, Si is defined respectively as a sum (Si = PQ2Qi SQ), or O -ed (Si = WQ2Qi SQ). Computing the predicted summary. Using the session summaries of the past users anda vector similarity metric, we construct the (|F| |F|) fragment-fragment matrixthat contains all similarities sim(-, i), -, i 2 F. The recommendation seed, modeled bySpred 0 , represents the estimated importance of each query fragment with regard to theactive users behavior S0. Similarly to the item-to-item collaborative filtering approachof web recommendersystems, we employ the fragment-to-fragment similarities that arecomputed in the previous step: Spred 0 [] = P-2R S0[-]-sin(-,i) P-2R sin(-,i)

VI. IDENTIFY, RESEARCH AND COLLECT IDEA

Collaborative Framework with User Personalization for Efficient web Search : Mining approach

Introduction

User personalization becomes more important task for web search engines. We develop a unified model to provide user personalization for efficient web search. We collect implicitfeedback from the users by tracking their behavior on the web page based on their actionson the web page. We track actions like save, copy, bookmark, time spent and logging intodata base, which will be used to build unified model. Our model is used as a collaborativeframework using which related users can mine the information collaboratively with littleamount of time. Based on the feedback from the users we categorize the users and searchquery. We build the unified model based on the categorized information, using which weprovide personalized results to the user during web search. Our methodology minimizesthe search time and provides more amount of relevant information.Methodology.The user interface is a tabbed web browser, which is a part of the system. Through thisbrowser the user can provide short-term query simultaneously in multiple tabs for hisinformation need. The user interacts with the system to give search query, to view theranked results and to view the re ranked results. The re ranking is done based on the pastsearch behavior of the user with the system. The browser also supports for providing actions like SAVE, COPY, PRINT AND E-MAIL, which depicts the importance of the web page for his need. The browser also projects the re-ranked results in an interactive graphlike structure rather than list based representations. Each user-visited web page is represented by a set of index words that comes under top list. The usage time of each searchquery and usage-time of each visited page is calculated transparently without disturbingthe user. Based on the search query, index word and usage-times, the User ConceptualIndex (UCI) is calculated. The UCI can be represented mathematically as a function ofweights of above parameters. The usage time directly indicates whether previous searchresults were relevant or irrelevant to the users information need. The search queries thathave similar or related meaning are categorized to a group using worddictionary in-orderto avoid inconsistencies that arise in above strategies. The visited-pages that have similaror related index words are also categorized to recommend the pages for a novice user. Theusers with similar search behavior are categorized to a group to improve the efficiency ofpersonalization mechanisms. The pages are re-ranked by analyzing individuals behaviorand are projected to them in dual window.

Some of the strategy for personalization ofweb search is described as follows:

1. A users search history can be collected withoutdirect involvement.

2. The users profile can be constructed automatically from the userssearch history and is dynamically updated.

3. The categories that are likely to be ofinterest to the user are deduced based on his search behavior.

Feature Extraction

The first step in the project is to extract the feature of the user-visiting page. The featureof a page, P is defined as a set of top n frequently occurring terms. In order to extractthe features, the source content of each page is extracted and it is de-tagged. From the de-tagged page, the stop words are removed and the terms in the page are extracted. From the set of terms, the top n frequently occurring terms is extracted. These n termsform as index words of the page.

Algorithms

Feature Extraction

User visiting page Pi
Procedure:
step1:

The de- tagged and stop words eliminated page Pi can be represented asIW= IW1, IW2, IW3, IWn andF= F1, F2, F3 Fn where IW is the index word set and F is the Frequency set corresponding to IWand n is the number of index words in the page.

step2:

Select top k frequency wordsFtopk= F1, F2 Ft which corresponds to IWtopk= IW1, IW2 IWt Where k<=t<=n.

step3:

Compute the mean for the above set(Ftopk) =[F1+F2++Ft]/t

step4:

The keywords in IWtop that have frequency above (Ftopk) form the feature ofthe page.

step5:

Now represent the feature of the page asFeature (Pi) = F1, F2 FmWhere 1<=m<=k.

step6:
End

User Association Analysis

The user association is analyzed to find the similarity of search among different users. From the set of visited-pages, the actions performed by the user are monitored. Fromthe action it is concluded whether the page is useful to the user. The order in which the page is visited is also tracked and a directed graph is constructed. The usage-time for each page forms the weight of the page.

Algorithm :

Similarity measure
Given: User behavior graph Procedure:
step1:
Indexingfor i:=1 to N dofor every vertex j of the web graph doBehavior [i] [j] []: =reversed path of length l starting from j.
end
end

step2:
User Sim(i,j) ,Sim:=0
for i:=1 to N do
for j:=1 to N do
let k be the smallest offset with
Behavior[i][u][k]=Behavior[j][u][k]
if such k exists then
Sim=Sim+ck
end
end
return Sim/N
step 3:
End

The above algorithm is used to identify the similarity behavior between two users. Whenever the search behavior is common, then it is certain that the users might comefrom same source point. Thus, higher the length l in the above algorithm, greater is the
similarity.

Search Query Categorization

Greater the number of times a user uses a particular SQ, greater is the interest of the useron the particular topic associated with the search query terms. If the previous searchresult is not relevant to the users information need, then the user might modify the SQ to _t into their context of search. Even though the search keyword gets changed, the information need on a topic doesn't get changed in that session. Hence the alternatekeyword supplied by the user may also be intended to search exactly for same topic. So it is necessary to identify the alternate meanings of users search query, which leads to categorization of the search query. The query categorization is necessary to reduce thelimitations in key word based search.

Algorithm :

SQ Categorization

Step1: Collect all the search queries given by the user in a due course of time.

Step2: Find the alternate meaning of the search query using a word dictionary.

Step3: Find whether the result of exists in the search query set provided by the user.

Step 4: If such commonalities exist, update the TF matrix and ST matrix.

Visited Page Categorization

Higher the similarity ranks between two users, greater the commonalities of search between them. Two pages can be said to similar even they spoke of exactly the same topicwith different keywords. Hence it is necessary to identify the alternate meanings of the Index words, which leads to categorization of the visited page. The categorization of the page is used to expand the similarity rank calculation, which aids to identify common search behavior.

Algorithm:

Page Classification

Step1: Collect all the index terms of all the visited pages by the user in a due course oftime.

Step2: Find the alternate meaning of the index terms using a word dictionary.

Step3: Find whether the result from step2 exists in the index terms set collected from user history.

Step 4: If such commonalities exist, update the SFmatrix.

Use of Simulation software

- Jdk , JCreator , Net beans etc.
- Oracle , MySQL, WampServer etc
- SkyServer etc.
- TC for C programming

VII. CONCLUSION

Scientists need help in exploring databases Query recommendations can be an effective tool in guiding exploration.

Collaborative filtering provides a natural method to generate recommendations Experiments show promising results on real world datasets

ACKNOWLEDGMENT

With all respect and gratitude, we would like to thank all people who have helped us directly or indirectly for the completion of this project work .We express our heartily gratitude towards **Prof. S. G. Chordiya** for guiding us to understand the work conceptually and also for his constant encouragement to complete this Project work on ***Query Recommendations for Interactive Database Exploration*** We also express our thanks to **Prof C.N.Patki** Head of department of Computer Engineering for providing necessary information and required resources.With deep sense of gratitude we thank to our Principal **Prof.Dr.Dayanand P.Joshi** and Management of the A.V.E.W. Trust's for providing all necessary facilities and their constant encouragement and support. Last but not the least we thanks to all the Teaching and Nonteaching staff members of Computer Engineering Department for providing necessary information and required resources.

We are ending this acknowledgement with deep indebtedness to our friends who have helped us.

REFERENCES

[1] International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 6, June 2014

[2] J. Akbarnejad, G. Chatzopoulou, M. Eirinaki, S. Koshy, S. Mittal, D. On, N. Polyzotis, and J. S. V. Varman. SQL QueRIE Recommendations (demo paper). In Proc.of the 36th International Conference on Very Large Data Bases (VLDB 2010), 2010.

[3] A. Giacometti, P. Marcel, and E. Negre, Recommending Multidimensional Queries,in Proc. Of the 11th Intl. Conf. on Data Warehousing and Knowledge Discovery(DaWaK09), 2009.

[4] G. Chatzopoulou, M. Eirinaki, and N. Polyzotis, Collaborative filtering for interactive database exploration, in Proc. Of the 21st intl. Conf. on Scientific and StatisticalDatabase Management(SSDBM 09), 2009

[5] H. JERBI, F. RAVAT, O. TESTE, G. ZURFLUH, Preference basedRecommendations for OLAP Analysis , International Conference on DataWarehousing and Knowledge Discovery (DaWaK 2009), Linz (Austria), 2009, pp.467- 478.

[6] A. Giacometti, P. Marcel, and E. Negre. Recommending Multidimensional Queries.In DaWaK09.

[7] [CE09] G. Chatzopoulou, M. Eirinaki, and N. Polyzotis, Query Recommendations for Interactive Database Exploration, in SSDBM09.

[8] Amazon.com Recommendations, Item-to-Item Collaborative Filtering, Greg Linden,Brent Smith and Jeremy York.

[9] Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents, Manos Papagelis, Dimitris Plexousakis.35

[10] G. Chatzopoulou, M. Eirinaki and N. Polyzotis. Query recommendations for interactive database exploration. In proceedings of SSDBM, pages 3-18, New Orleans,LA, USA, June 2009.

AUTHORS

First Author –Rajendra Desale ,BE(Computer Technology), Shatabdi Institute Of Engineering, Agaskhind, rajdesale2151@gmail.com

Second Author – Vaibhav Patil , BE(Computer Technology), Shatabdi Institute Of Engineering, Agaskhind ,vaibhav.ps444@gmail.com

Third Author – Tushar Aware , BE(Computer Technology), Shatabdi Institute Of Engineering, Agaskhind , tusharaware888@gmail.com