Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 3, March 2015, pg.749 – 752

SURVEY ARTICLE

A Survey on Recommendation Techniques for Database Systems

Ms. Kiran S. Sutar[1], Mr. Hemant A. Tirmare[2]

[1]Department of Technology Shivaji University, Kolhapur, India

[2]Department of Technology Shivaji University, Kolhapur, India

[1] kiransutar333@gmail.com; [2] hat_tech@unishivaji.ac.in

Abstract— Database exploration is the process of discovering interesting information from the database. The only way to interact with database systems is by using querying languages, for example SQL. Size of database systems is huge and it always grows tremendously over time. And also most of the users do not have SQL skills to explore the database. Even though users are experts in SQL, they may not know which part of the database provides valuable information. It is possible that they may neglect the queries which will retrieve information of their interest. This degrades task of database exploration. To address such a vital problem recommender systems are emerged, which are capable of automatically recommending useful queries to the users. In this paper we discussed some approaches taken by recommender systems. And then presented various frameworks designed by different authors to give recommendations.

Keywords— Data mining, Information Retrieval, Query recommendation, Database Exploration, Relational Databases

I. INTRODUCTION

Most of the scientific communities use relational databases for the storage of large volumes of data. Examples of such systems include SkyServer which provides access to huge amount of astronomical data using web-enabled interface. Users can retrieve information from these databases using SQL queries. Even though such a system supports complex SQL queries over large databases users faces grate difficulties in retrieving useful information. Generally users do not have clear knowledge of database schema. Even though users are expert in SQL it is possible that they may neglect the queries which will retrieve interesting information, or they may not know which tables or fields of the database contain useful information. The problem is more serious for the users who do not have SQL skills. This clearly precludes the database exploration, and subsequently degrades the benefits of using database systems.

Interactive database exploration is a process of information mining. The objective of this exploration process is to discover interesting information or verify a particular hypothesis. User formulates queries on the basis of this goal. It is obvious that the queries posted by a user during one session to the database are typically interrelated. Next query in the sequence presented by the user is based on the result of previous queries.

Typically user doesn't know the schema of the database; as a result he tries number of queries to find particular information even though that information will be answered in single query. It would be very efficient if the database system

could recommend the appropriate query immediately after user fires his first query. This is possible if a similar session already exists in the systems query log, and can be used to generate recommendations for the current user.

To address this vital problem, we can use techniques imposed by web recommender systems. Consider the following example of Book recommendations in user based web collaborative filtering: If Alice and Bob both like book X and Alice likes book Y then Bob is likely to be interested in reading book Y.

II. TYPES OF RECOMMENDATION APPROACHES

Today large number of web applications involves the task of automatically finding items which may be potentially interested for user. Such a task is called query recommendation. There are different approaches taken by recommendation systems.

A. Content-based

In this system each item is described by its features. For instance consider an example of online book store. In this case each book is described by features like category, author, publication etc. These features are used to calculate similarity between different items. Past history of user is used to recommend new items. The items which are similar to the previously liked items by user are recommended to them. For example, if a user likes networking related books of author Behrouz Forouzan then the system suggest him other books of the same author which are also related to networking category. These systems generally construct profile for each user which contains features of items previously liked or rated by that user. Profile captures interest of user, which is subsequently used in the recommendation process.

B. Collaborative filtering

These systems find users which are similar to the current user, and recommends items liked by that similar users. Past history of users is used to find similar users. Such systems focus on user to user similarity. That's why in [1] the author used the term "people-to-people correlation" for collaborative filtering. [2] Highlights some recent extensions in the in the field of collaborative filtering technique. In [3] their QueRIE system uses collaborative filtering approach.

C. Demographic

Some systems use demographic information to generate suggestions. The idea behind this approach is that different suggestions should be generated for different demographic niches. For example, the website suggested for the user depends on their language and country. Or suggestions may vary depending on age of user. Even though such systems are quite popular in marketing literature, there has been somewhat little proper research into demographic systems [4].

D. Community-based

These systems make use of preferences of users friends. The system follows the saying "Tell me who your friends are, and I will tell you who you are" [5][6]. Evidence suggests that most of the people take suggestions from their friends than from similar but unknown users [7]. This type of systems also called social recommender systems [8]. This type of recommender systems extracts information about the social relations of the user and subsequently uses the preferences of user's friends. These systems totally relay on the ratings given by user's friends.

E. Hybrid recommender systems

Hybrid recommender system combines above mentioned techniques. For example, a hybrid system which uses techniques A and B tries to take benefits of A to reduce drawbacks of B. We can create several different hybrid systems by using two (or more) basic recommendation techniques.

III. EXISTING RECOMMENDER SYSTEMS

In the paper [9] the author presented an algorithm that first evaluates similarities between various items, and subsequently uses those similarities to determine set of items to be suggested. They have presented two methods for computing resemblance between various items. One method is cosine-based similarity which is derived from vector space model. In this method each item is considered as a vector in the space of users. The measure of similarity is nothing but cosine between these vectors. Since, cosine function is symmetric similarity function it considers frequently liked items similar to the other frequently liked items. Frequently liked items and infrequently liked items will not be considered as similar. Second method is conditional probability based similarity. In this method the similarity is calculated by using conditional probability of liking one of the item given that the others has already been liked. In general, the conditional probability of preferring j, when i have already been preferred is more than the number of users that preferred both items i and j divided by the total number of users that preferred i.

In [10] the author has presented a collaborative framework that recommends OLAP queries to the users. They have used approximate String Matching Technique [11] of Information Retrieval. They have represented multi dimensional queries (MDX) as set of references. The task of comparing two MDX queries includes comparison of two sets of references. For this comparison they have used classical Hausdorff distance [12]. Generally, two sets are considered to be close if every element of either set is close to some element of the other set.

Another different type of recommendation system called YMAL is explained in [13]. In YMAL when user submits his query the system returns the result of that query along with additional suggested results. In this, instead of suggesting other queries the system suggests results of other useful queries. Assume that database system is denoted by D set of users is denoted by U. When a user $u \in U$ submits a query q then database system returns the result R(q) in the form of tuples. This R(q) may contain combination of relations of D. In addition to R(q) their system will recommend to u the set of tuples that may be useful for them. They called this set of extra tuples as "YMAL results" denoted by YMAL (q, u). They have mentioned three different approaches for recommending YMAL results. First one is current-state approach, which utilize the content and schema of the result R(q) as well as database instance. Second is History-based approach, which utilizes past queries submitted by user to the database system. This approach makes use of systems query log. The third is External Sources approach, which focuses on external resources of database system; for example relevant web pages, ontology's, etc.

The current state approach is again classified into three categories depending on the scope of analysis. (i) Local analysis deals with only the properties of R(q), whereas (ii) global analysis deals with properties of database D. (iii)Hybrid analysis combines both local and global analysis.

There are three categories in History-based approach, (i) query-based approach which is similar to content –based recommendations. In that when a user submits a query q, YMAL(q,u) includes results of previous queries $q_i \in Q$ that are most similar to current query q. Similarity is calculated by using similarity function $sim_q(q_i, q_j)$ between queries.

$$sim_q(q_i, q_j) = |R(q_i) \cap R(q_j)| \quad (1)$$

(ii) User-based approach is similar to collaborative recommendations. In this approach YMAL(q,u) includes the results of queries submitted by other users $u_i \in U$ that are similar to current user u. To calculate the similarity between users the function $sim_u(u_i, u_j)$ is used.

$$sim_u(u_i, u_j) = |Q(q_i) \cap Q(q_j)| \quad (2)$$

(iii) Hybrid approach first finds the most similar users to u and subsequently recommends most similar queries of those similar users.

SnipSugest [14] provides context aware on the go support for formulation of SOL query. SnipSuggest automatically recommends extra column names for the SELECT clause; table names, join conditions for FROM clause and predicates for the WHERE clause, and column names for GROUP BY clause of the current query. It maintains log of queries to suggest snippets. It also provides support for sub-queries. They have introduced two terminologies: query snippets and DAG. Query snippet is small fragment of SQL query which belong to particular SQL clause. Queries are represented by set of features. These features are nothing but specific fragments in the query, i.e. table name in the FROM clause, column name in the SELECT clause. Every possible set of features are considered to be the vertex in the directed acyclic graph (DAG). Their system transforms partially written query into set of features, which subsequently maps onto a node in the DAG. Outgoing edge from that node represents addition of a feature. The query which the user wants to write is somewhere below the current vertex. Recommendation problem is now become the problem of ranking the addition of different features. They have introduced two ways to deal with this problem. One is to recommend the feature which is most popular. Alternatively, it can recommend k features which can cover maximum number of queries. Two metrics are defined to evaluate the quality of recommendations: accuracy and coverage. Two algorithms are developed SSAccuracy and SSCoverage to recommend snippets.

IV. CONCLUSIONS

In this paper we explained the need for recommender systems in the context of database systems. There are various approaches taken by recommender systems, in that the collaborative filtering is most popular approach. Different approaches are useful for different application of database systems. In this paper we have surveyed different recommender systems for database exploration. Different authors presented different architectures for providing recommendations. Each author's point of view is different for giving recommendations. Some directly suggests potentially useful SQL queries to the users, while other provides the facility of auto completion of SQL queries on the go. Instead of recommending queries some authors presents the results of other useful queries with the results of original query.

REFERENCES

[1] Schafer, J.B., Konstan, J.A., Riedl, J.: E-commerce recommendation applications. Data Mining and Knowledge Discovery 5(1/2), 115–153 (2001)

[2] Advances in Collaborative Filtering Yehuda Koren and Robert Bell F. Ricci et al. (eds.), Recommender Systems Handbook, DOI 10.1007/978-0-387-85820-3_5, © Springer Science+Business Media, LLC 2011

[3] Magdalini Eirinaki, Suju Abraham, Neoklis Polyzotis, and Naushin Shaikh, "QueRIE: Collaborative Database Exploration", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 26, NO. 7, pp. 1778-1790, JULY 2014

[4] Mahmood, T., Ricci, F.: Towards learning user-adaptive state models in a conversational recommender system. In: A. Hinneburg (ed.) LWA 2007: Lernen - Wissen - Adaption, Halle, September 2007, Workshop Proceedings, pp. 373–378. Martin-Luther-University Halle-Wittenberg (2007)

[5] Arazy, O., Kumar, N., Shapira, B.: Improving social recommender systems. IT Professional 11(4), 38–44 (2009)

[6] Ben-Shimon, D., Tsikinovsky, A., Rokach, L., Meisels, A., Shani, G., Naamani, L. "Recommender system from personal social networks". In: K. Wegrzyn-Wolska, P.S. Szczepaniak (eds.) AWIC, *Advances in Soft Computing*, vol. 43, pp. 47–55. Springer (2007)

[7] Sinha, R.R., Swearingen, K.: Comparing recommendations made by online systems and friends. In: DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries (2001)

[8] Golbeck, J.: Generating predictive movie recommendations from trust in social networks. In: Trust Management, 4th International Conference, iTrust 2006, Pisa, Italy, May 16-19, 2006, Proceedings, pp. 93–104 (2006)

[9] Mukund Deshpande, George Karypis, "Item-Based Top-*N* Recommendation Algorithms", ACM Transactions on Information Systems, Vol. TBD, No. TBD, TBD 20TBD, Pages 1–34.

[10] Giacometti, P. Marcel, and E. Negre, "Recommending multidimensional queries," in *Proc. 11th Int. Conf. DaWaK*, Linz, Austria, 2009.

[11] Navarro, G.: A guided tour to approximate string matching. ACM Comput. Surv. 33(1), 31–88 (2001)

[12] Hausdorff, F.: Grundz¨uge der Mengenlehre. von Veit (1914)

[13] K. Stefanidis, M. Drosou, and E. Pitoura, "'You May Also Like' results in relational databases," in *Proc. 3rd Int. Workshop PersDB*, 2009.

[14] N. Khoussainova, Y. Kwon, M. Balazinska, and D. Suciu, "SnipSuggest: Context-aware autocompletion for SQL," *PVLDB*, vol. 4, no. 1, pp. 22–33, 2011.

