ISSN:0975-9646

User- and Query- Dependent Ranking for Searching Technical Papers

Manish R. Jansari[1], Prof. P. M. Chawan[2]
[1,2] *CE & IT Department*
Veermata Jijabai Technological Institute,
Mumbai- 19, India

***Abstract-* In the era of Internet where millions of people browsing over millions of websites; searching for web databases such as Product database, Locomotive database etc. have become a routine task. Ranking and returning the most relevant result of a query based on the type of user is important in this aspect. Previous approaches have used values of database frequencies, user profiles and queries. Thus the ranking was done in a user- and /or query-independent manner. A new approach known as User and Query Dependent Ranking for giving ranking to query results can be useful in this context. This ranking framework is based on two fundamental aspects to the problem of ranking query results. They are query similarity and user similarity. These similarities are exploited to make efficient ranking of query results.**

***Keywords*—Automated ranking, Recommendation Systems, User similarity, Query similarity, Ranking query results, Data Mining.**

I. Introduction

The World Wide Web has more and more online Web databases which can be searched through their Web query interfaces. All the Web databases make up the deep Web (hidden Web or invisible Web). Often the retrieved information (query results) is enwrapped in Web pages in the form of data records. These special Web pages are generated dynamically and are hard to index by traditional crawler-based search engines, such as Google and Yahoo. These kind of special Web pages are often called as deep Web pages. [1]

Data mining is the process of extracting novel, interesting and useful patterns from usually large-scale data. Data mining is used in a wide range of industries - including retail, finance, health care, manufacturing transportation etc.

The World Wide Web and its associated distributed information services, such as Yahoo!, Google, AltaVista, provide rich, worldwide, on-line information services,[2] where data objects are linked together to facilitate interactive access. Users seeking information of interest traverse from one object via links to another. Such systems provide ample opportunities and challenges for data mining. For example, understanding user access patterns will not only help improve system design but also leads to better marketing decisions (e.g., by placing advertisements in frequently visited documents, or by providing better customer/user classification and behavior analysis). Capturing user access patterns in such distributed information environments is called Web usage mining (or Weblog mining) [2].

The web databases are from various domains such as product, locomotive, education, health care and so on. These web databases are searched by online users through a search mechanism provided. The queries can have criteria that correspond to the attributes of the database schema. When results returned are huge in number, user time gets wasted in brewing for required information.

To overcome this problem the present web databases simplify the results by sorting them in a particular attribute. This may not be suitable to the requirements of many users who prefer ordering on multiple attributes. Many existing web databases follow simple sorting for ranking while the extension of SQL allows providing attribute weights [3]. For most web databases this approach is not user friendly and time consuming for users as most of the time needs to be spent in browsing the query results. For this reason an automated ranking of query results is studied and some techniques are proposed in [4]. These approaches are either user query dependent or user dependent way of ranking query results. Another approach which is used to build extensive user profiles and in that case users are supposed to order the records, this approach is proposed for user-dependent ranking and that do not differentiate the difference between different queries and provide a single ranking order for any query. Even recommender systems made use of either user similarity or query similarity. Some of them are collaborative in nature and some of them are content based filters.

II. Related Work

Although there was no notion of ranking in traditional databases, it has existed in the context of information retrieval for quite some time. With the advent of the Web, ranking gained prominence due to the volume of information being searched/ browsed. Currently, ranking has become ubiquitous and is used in document retrieval systems, recommender systems, Web search/browsing, and traditional databases as well [6].

2.1 Ranking in Recommendation Systems:

Most existing recommender systems can be classified into two categories: collaborative filtering and content-based filtering. Hybrid recommender systems combine the advantages of the two for improved recommendation performance. Traditional recommender systems are rating-based. However, predicting ratings is an intermediate step towards their ultimate goal of generating rankings or recommendation lists.

i) Collaborative filtering

Collaborative filtering methods are based on collecting and analyzing a large amount of information on users' behaviors, activities or preferences and predicting what users will like based on their similarity to other users. A key advantage of the collaborative filtering approach is that it does not rely on machine analyzable content and therefore it is capable of accurately recommending complex items such as movies without requiring an "understanding" of the item itself. Many algorithms have been used in measuring user similarity or item similarity in recommender systems. For example, the k-nearest neighbor (k-NN) approach and the Pearson Correlation.

Collaborative Filtering is based on the assumption that people who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past.

When building a model from a user's profile, a distinction is often made between explicit and implicit forms of data collection.

Examples of explicit data collection include the following:

- Asking a user to rate an item on a sliding scale.
- Asking a user to search.
- Asking a user to rank a collection of items from favorite to least favorite.
- Presenting two items to a user and asking him/her to choose the better one of them.
- Asking a user to create a list of items that he/she likes.

Examples of implicit data collection include the following:

- Observing the items that a user views in an online store.
- Analyzing item/user viewing times
- Keeping a record of the items that a user purchases online.
- Obtaining a list of items that a user has listened to or watched on his/her computer.
- Analyzing the user's social network and discovering similar likes and dislikes

The recommender system compares the collected data to similar and dissimilar data collected from others and calculates a list of recommended items for the user. Several commercial and non-commercial examples are listed in the article on collaborative filtering systems.

One of the most famous examples of collaborative filtering is item-to-item collaborative filtering (people who buy x also buy y), an algorithm popularized by various online shopping websites (like Amazon.com's) recommender system.

ii) Content-based filtering

Content-based filtering methods are based on a description of the item and a profile of the user's preference. In a content-based recommender system, keywords are used to describe the items; beside, a user profile is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past (or is examining in the present). In particular, various candidate items are compared with items previously rated by the user and the best-matching items are recommended. This approach has its roots in information retrieval and information filtering research.

To abstract the features of the items in the system, an item presentation algorithm is applied. A widely used algorithm is the tf–idf representation (also called vector space representation).

To create user profile, the system mostly focuses on two types of information:

1. A model of the user's preference.
2. A history of the user's interaction with the recommender system.

Basically, these methods use an item profile (i.e. a set of discrete attributes and features) characterizing the item within the system. The system creates a content-based profile of users based on a weighted vector of item features. The weights denote the importance of each feature to the user and can be computed from individually rated content vectors using a variety of techniques. Simple approaches use the average values of the rated item vector while other sophisticated methods use machine learning techniques such as Bayesian Classifiers, cluster analysis, decision trees, and artificial neural networks in order to estimate the probability that the user is going to like the item.

Direct feedback from a user, usually in the form of a like or dislike button, can be used to assign higher or lower weights on the importance of certain attributes.

A key issue with content-based filtering is whether the system is able to learn user preferences from user's actions regarding one content source and use them across other content types. When the system is limited to recommending content of the same type as the user is already using, the value from the recommendation system is significantly less than when other content types from other services can be recommended. For example, recommending news articles based on browsing of news is useful, but it's much more useful when music, videos, products, discussions etc. from different services can be recommended based on news browsing.

iii) Hybrid recommender systems

Recent research has demonstrated that a hybrid approach, combining collaborative filtering and content-based filtering could be more effective in some cases. Hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model. Several studies empirically compare the performance of the hybrid with the pure collaborative and content-based methods and demonstrate that the hybrid methods can provide more accurate recommendations than pure approaches. These methods can also be used to overcome some of the common problems in recommender systems such as cold start and the sparsity problem.

Netflix is a good example of hybrid systems. They make recommendations by comparing the watching and searching habits of similar users (i.e. collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (content-based filtering).

A variety of techniques have been proposed as the basis for recommender systems: collaborative, content-based, knowledge-based, and demographic techniques. Each of these techniques has known shortcomings, such as the well known cold-start problem for collaborative and content-based systems (what to do with new users with few ratings) and the knowledge engineering bottleneck in knowledge-based approaches. A hybrid recommender system is one that combines multiple techniques together to achieve some synergy between them.

- Collaborative: The system generates recommendations using only information about rating profiles for different users. Collaborative systems locate peer users with a rating history similar to the current user and generate recommendations using this neighborhood.
- Content-based: The system generates recommendations from two sources: the features associated with products and the ratings that a user has given them. Content-based recommenders treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on product features.
- Demographic: A demographic recommender provides recommendations based on a demographic profile of the user. Recommended products can be produced for different demographic niches, by combining the ratings of users in those niches.
- Knowledge-based: A knowledge- based recommender suggests products based on inferences about a user’s needs and preferences. This knowledge will sometimes contain explicit functional knowledge about how certain product features meet user needs.

The term hybrid recommender system is used here to describe any recommender system that combines multiple recommendation techniques together to produce its output. There is no reason why several different techniques of the same type could not be hybridized, for example, two different content-based recommenders could work together, and a number of projects have investigated this type of hybrid: NewsDude, which uses both naive Bayes and kNN classifiers in its news recommendations is just one example.

Seven hybridization techniques:

- Weighted: The score of different recommendation components are combined numerically.
- Switching: The system chooses among recommendation components and applies the selected one.
- Mixed: Recommendations from different recommenders are presented together.
- Feature Combination: Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.
- Feature Augmentation: One recommendation technique is used to compute a feature or set of features, which is then part of the input to the next technique.
- Cascade: Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.
- Meta-level: One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

Another important distinction apart from recommendation systems is the notion of *similarity*. In content filtering, the similarity between items is established either using a domain expert, or user profiles, or by using a feature recognition algorithm over the different features of an item (e.g., author and publisher of a book, director and actor in a movie, etc.). In contrast, since our framework requires establishing similarity between actual SQL queries (instead of simple keyword queries), the direct application of these techniques does not seem to be appropriate. To the best of knowledge, a model for establishing similarity between database queries (expressed in SQL) has not received attention.

In addition, a user profile is unlikely to reveal the kind of queries a user might be interested in. Further, since we assume that the same user may have different preferences for different queries, capturing this information via profiles will not be a suitable alternative.

The notion of user similarity used in our framework is identical to the one adopted in collaborative filtering; however, the technique used for determining this similarity is different [6].

In collaborative filtering, users are compared based on the ratings given to individual items (i.e., if two users have given a positive/negative rating for the same items, then the two users are similar).

In the context of database ranking, [6] propose a rigorous definition of user similarity based on the similarity between their respective ranking functions, and hence ranked orders. Furthermore, their work extends user-personalization using context information based on user and query similarity instead of static profiles and data analysis.

2.2 Ranking in Databases:

Although ranking query results for relational and Web databases has received significant attention over the past years, simultaneous support for automated *users* and *query-dependent* ranking has not been addressed in this context. For instance, address the problem of *query dependent* ranking by adapting the vector model from information retrieval, whereas do the same by adapting the probabilistic model. However, for a given query, these techniques provide the same ordering of tuples across all users.

Employing user personalization by considering the context and profiles of users for *user-dependent* ranking in databases has been proposed in [5]. A drawback in this

work is that it does not consider that the same user may have varied ranking preferences for different queries.
The closest form of *query*- and *user-dependent* ranking in relational databases involves manual specification of the ranking function/preferences as part of SQL queries. However, this technique is unsuitable for Web users who are not proficient with query languages and ranking functions. In contrast, our framework provides an automated *query*- as well as *user-dependent* ranking solution without requiring users to possess knowledge about query languages, data models and ranking mechanisms[6].

II. Problem Definition And Architecture

3.1 Problem Definition:

While searching for technical papers by different types of Users with large set of queries, the returned results should be more relevant i.e. the results returned should be ranked in a user- and query- dependent manner.
Users from different specializations (e.g. Computer Engineering, Civil Engineering, Mechanical Engineering etc...) search for technical papers.
Our goal is to provide most relevant results to these different categories of users by categorizing the Users and the Queries as well.

3.2 Proposed System Architecture:

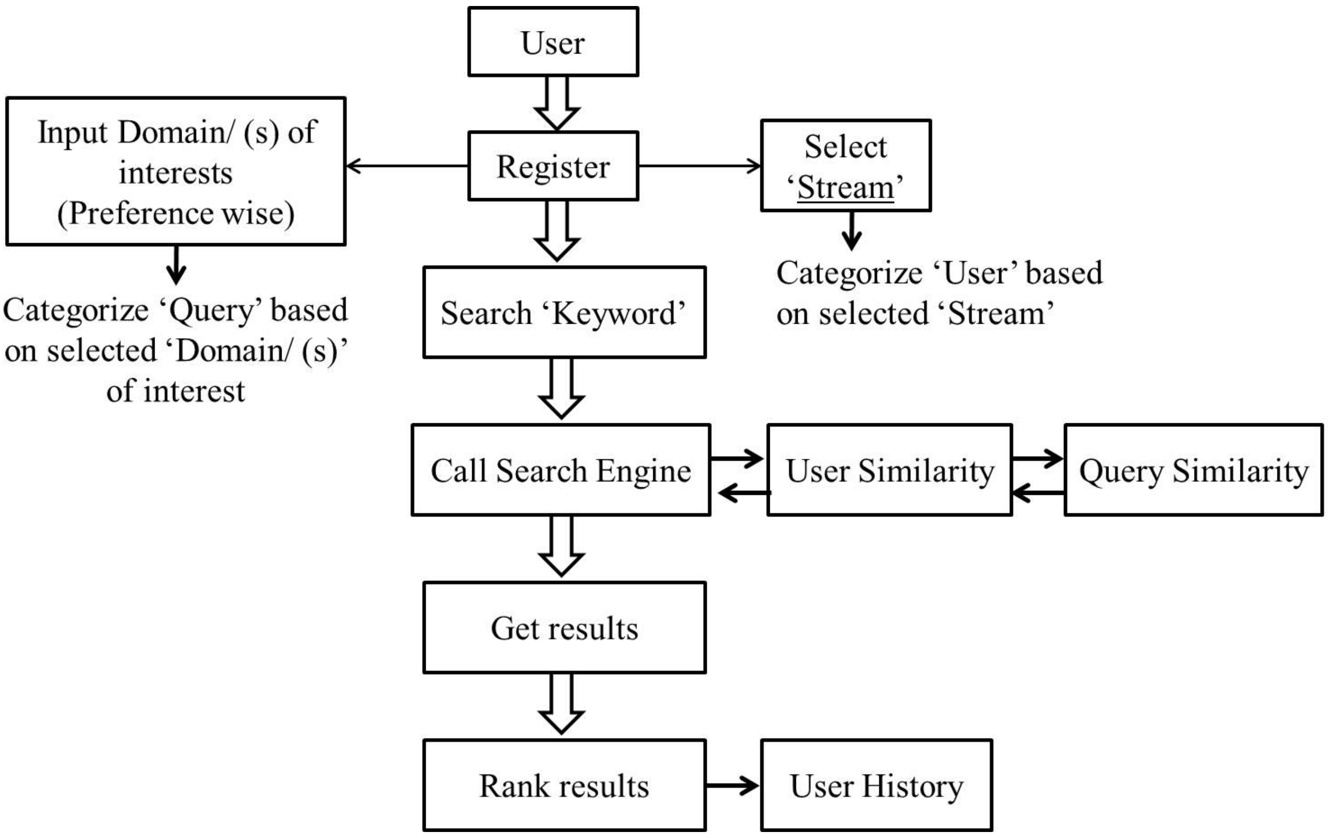
User
Input Domain/ (s) of interests (Preference wise)
Register
Select 'Stream'
Categorize 'Query' based on selected 'Domain/ (s)' of interest
Search 'Keyword'
Categorize 'User' based on selected 'Stream'
Call Search Engine
User Similarity
Query Similarity
Get results
Rank results
User History

Fig.1. Proposed System Architecture

3.3 Steps to be followed in implementation:

Step 1: Studying the current suggested techniques.
Step 2: Preparing datasets.
Step 3: Determining the relevant attributes to split the dataset.
Step 4: Developing the algorithm.
Step 5: Verify ranked results in real time.

3.4 Objectives:

1. Searching for Technical papers based on user given keywords by providing most relevant results based on user's category.

VI. Conclusion And Future Work

In this paper various ranking techniques are described such as ranking in Collaborative, content-based and hybrid recommendation systems.
Proposed architecture is also presented. We proposed a system for searching Technical Papers on the web also we proposed a user and query dependent solution for ranking query results.
In future, to provide most relevant results, user history can be considered for ranking the results.

ACKNOWLEDGMENT

I would like to thank all those people whose support and co-operation has been a valuable asset during the course of this Paper. I would also like to thank our Guide Prof. P. M. Chawan for guiding me for completing this paper.

REFERENCES

[1] Wei Liu, Xiaofeng Meng, Member, IEEE, and Weiyi Meng, Member, IEEE, "ViDE: A Vision-Based Approach for Deep Web Data Extraction," in IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 22, NO. 3, MARCH 2010.

[2] P.Ayyadurai, S.Jayanthi, "Automated Ranking for Web Databases using K-Means Algorithm and UQDR Approach," International Conference on Research Trends in Computer Technologies (ICRTCT - 2013), Proceedings published in International Journal of Computer Applications® (IJCA) (0975 – 8887).

[3] A. Marian, N. Bruno, and L. Gravano, "Evaluating Top-k Queries over Web-Accessible Databases," ACM Trans. Database.

[4] Weifeng Su, Jiying Wang, Qiong Huang, Fred Lochovsky, "Query Result Ranking over E-commerce Web Databases," CIKM'06, November 5 – 11, 2006.

[5] G. Koutrika and Y. E. Ioannidis. Constrained optimalities in query personalization. In *SIGMOD Conference*, pages 73–84, 2005.

[6] Aditya Telang, Chengkai Li, Sharma Chakravarthy, ―One Size Does Not Fit All: Towards User- and Query-Dependent Ranking For Web Databases, IEEE Transactions on Knowledge and Data Engineering, 2012.