ISSN (Online) : 0974-5645
ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 8(4), 301–306, February 2015* DOI : 10.17485/ijst/2015/v8i4/60376

Personalized Search Engine using Social Networking Activity

Nathaneal Ramesh* **and J. Andrews**

Department of Computer Science and Engineering, Sathyabama University, Chennai - 600119, Tamil Nadu, India; nathaneal31@gmail.com, andrews_593@yahoo.com

Abstract

The main objective of this research work is to obtain a personalized search result required by the user, by creating user profile based on social networking activity. The user profile is actually constructed by pages liked by the individual user in Facebook on their respective user account. According to the constructed user profile the results are re-ranked and the personalised search results are obtained. Lingo- a novel algorithm is used for clustering the data. The search results are retrieved to user using Carrot2 API search engine. In the past, personalized search engines acquired data from surfing history implicitly or explicitly by machine learning whereas this work acquires data implicitly through user likes and also explicitly through user defined categories . The Facebook likes are given by each user only on their personal interest. Thus, these data play a vital in providing accurate search results to each user and provide exact search results as per the user interest.

Keywords: Information Retrieval, Personalization, Search Engine, Social Network and Web Search

1. Introduction

Web search engine is designed to search and retrieve information from the World Wide Web (WWW). The search results are presented in a list and they are often called to be hits. The information from the search results of any given query may be of web pages, images, videos and other different types of files.

A query is the text that the user submits to the search engine. Most commercial search engines return the same results roughly for the same given query, regardless of the individual user's real interest. Since queries that are submitted to search engines tend to be very short and ambiguous, they are not likely able to express the user's precise needs. For example, if the query to be searched by a user is apple, the search results will have documents explaining both apple fruit and apple computers. But the user may be a computer person who is really interested to know about apple computers alone. Therefore the search results should be in such a way that documents explaining apple computers should be displayed at the top.

Thus, Personalized Search engine returns the most appropriate search results related to users interest. For the query "puma", a zoologist would be interested in the puma (Cougar) animal, species living in mountainous regions and a normal person would be interested in knowing about puma clothes and sportswear for its new arrival and purchasing. User profiling is a fundamental component of any personalization applications. Personalized search engines ranks the search results based on the user interests. Previously these personalized search engines are constructed either implicitly or explicitly.

Implicit learning through surfing activities of user, that is by observing each user's behaviour such as the time spent reading an online document or by tracking down the pages visited by the user[1] or explicitly by making the system to learn through training, asking for feedback through preferences or ratings. Explicit construction of user profiles has several drawbacks. The user might provide inconsistent or incorrect information, and the profile which is built will be static whereas the user's interests may change over time, and the construction of

*Author for correspondence

the user profile places a burden on the user which they do not have willingness to accept. But this requires the user intervention. This can be overcome by construction of user profiles automatically and implicitly while the users browse the web. But even implicit profile construction is also not very accurate in all cases as it is of system's judgement. Improving the performance of Search engine for obtaining better search results is done using fuzzy logic[2].

But personalized search engines can be also improved in much better way by using the user's social networking activities. Networking sites like Facebook, twitter can be used to track user daily activities and therefore deduce his interests. Over the last decade, the World Wide Web and Web search engines have dramatically transformed the way people share their information. Recently, a new way of sharing and locating information, known as social networking, has become highly popular. While numerous studies[3,4,5] have focussed on the hyperlinked structure of the Web and have exploited it for searching content, few studies[6,7] have examined the information exchange in online social networks. This system uses few parameters of Facebook to construct user profile and in this work the personalized search engine is constructed by implicit as well as explicit learning.

A web search engine often returns thousands of pages in response to a big query, making it very difficult for users to browse or to identify the relevant information which the user needs. Clustering methods can be used automatically to group the retrieved documents into a list of meaningful built-in categories[8,9], as it is achieved through Enterprise Search engines such as Northern Light and Vivisimo, and the consumer search engines such as PolyMeta and Helioid, or through an open source software such as Carrot2[10].

2. Problem Statement

The task of the system is to derive the user interests based on his/her activity in social networking sites such as facebook, twitter etc and re rank the search results based on the user interest profile.

- The system recognizes the interests of the user and returns the appropriate results to the user.
- Enables the user to constantly feedback his interests without his own intentions, thus reducing the need to train the system.

3. Proposed System

In this work a personalized search engine is proposed which implicitly constructs user profile using User's facebook activities. User interest are derived from FB activities though explicit, it is done by user for his social networking needs and not for any personalization need. Therefore it is an implicit based learning method. Search result is derived from any popular search engine such as Google, yahoo, Bing, AltaVista etc. These search results are re ranked based on user interests.

Here, Lingo - a novel algorithm for clustering search results is presented, which emphasizes cluster description quality. Lingo algorithm first tries to make sure whether a human-perceivable cluster label can be created and then assigns documents to the category. Very importantly the frequently used phrases are extracted from the input documents, considering that these frequently used phrases will be the most useful informative source of human-readable topic descriptions. Then lingo performs reduction of the original term-document matrix using singular value decomposition (SVD). The SVD is performed and discovery of any existing latent structure of diverse topics in the search result is founded out. Finally group descriptions are matched with the extracted topics and assigns relevant documents to them.

SVD breaks a $t \times d$ matrix into three matrices namely U, Σ and V, Therefore $A = U\Sigma V^T$. Where, U is a $t \times t$ orthogonal matrix whose column vectors are called as the left singular vectors of A, and V is a $d \times d$ orthogonal matrix whose column vectors are called as the right singular vectors of A, and Σ is a $t \times d$ diagonal matrix that has the singular values of A ordered in a decreasing manner along its diagonal. The rank rA of, A matrix is equal to number of its non-zero singular values. The first rA columns of U form the orthogonal basis for the column space of A - an essential fact used by Lingo[11].

The (Figure 1.) shows the overall architecture of the system depicting the complete representation of modules. User logs in to the search engine using their Facebook account. Then User enters a search query and the results for it are obtained from an existing search engine like Google. Then the search results are pre-processed to extract the key terms that are related to the user query. Preprocessing is done with the Lingo clustering algorithm. The resultant cluster labels are said to be the key terms for the given query and are called as concepts. After the cluster formation, the search results are retrieved to user

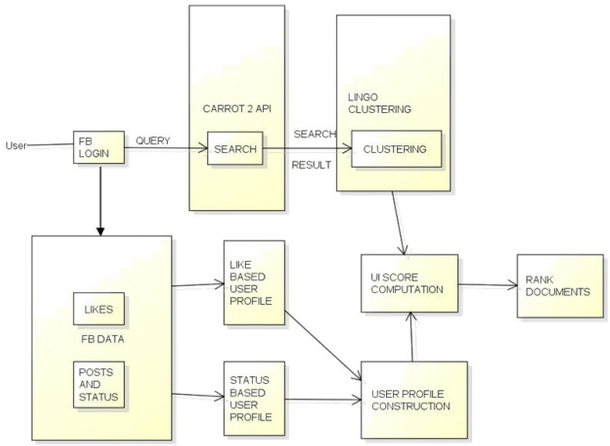

Figure 1. Functional architecture.

using Carrot2 search engine. The Carrot2 connects the local server and pulls the relevant data from Google. The steps involved in lingo clustering algorithm, are depicted in flow chart and are explained in detail (Figure 2).

3.1 Pre-Processing

Stemming and the stop word removal is a very common operation in the process of Information Retrieval. It is to be noted that these process always doesn't provide positive results. In certain applications stemming process doesn't show any improvement at all to the overall quality. Let it be so, but still study on recent experiments show that pre-processing technique is of great importance in Lingo clustering approach. It is because the input snippets are generated automatically from the original documents and they are generally very small. Though SVD can deal with noisy data, yet without pre-processing, the most of the discovered abstract concepts will be related to meaningless frequent terms. In pre-processing three steps are carried out: (i) HTML tags, entities and other characters are removed using text filtering, except for sentence boundaries. (ii) Each snippet's language is identified and then appropriate stemming and the (iii) stop word removal completes the pre-processing process.

3.2 Frequent Phrase Extraction

These are phrases that occur frequently in an ordered sequence in the given input documents. While writing about something, it is very common to use keywords that are related to the main subject in-order to maintain reader's attention. And for a good writing it usage of synonymy

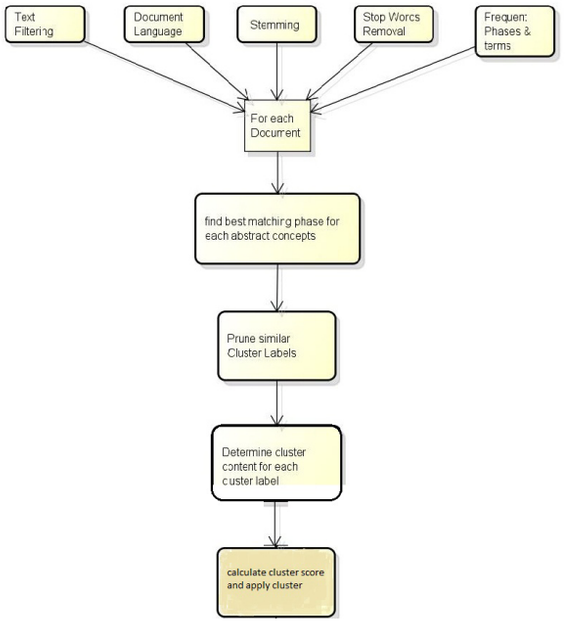

Figure 2. Flow of Lingo Clustering.

and pronouns is necessary so as to avoid annoying repetition. The former is partially overcome by lingo algorithm using the SVD-decomposed term document matrix to identify the abstract concepts. To be a candidate for a labelled cluster, the general guidelines for the frequent phrases or a single individual term is:

- Appearance of phrase or term in the input document at least certain number of times,
- Boundaries should not be cross sentence,
- Should be a entire phrase,
- The stop word should not be at the beginning or at the end.

3.3 Cluster Label Induction

After knowing the frequent phrases and individual frequent terms that exceeds the term frequency threshold, they are used for induction of cluster label. Three steps are followed for induction of cluster label. They are, (1) building of term-document matrix, (2) abstract concept discovery, (3) matching of phrase and label pruning.

The construction of term-document matrix is from single terms that exceed the predefined term frequency threshold. The Weight of each term is calculated using the standard term frequency, inverse document frequency (tf-idf) formula. In abstract concept discovery, the orthog-

onal basis of the term-document matrix is founded using SVD method. The vectors of this basis SVD's U matrix represent the abstract concepts that appear in the input documents. The 'K' value is estimated by selecting the Fresenius norms of the term-document matrix 'A' and its k-rank approximation 'Ask'. Let the threshold 'q' be a percentage value that determines to what extent the k-rank approximation should retain the original information in matrix A. Hence k is defined as the minimum value that satisfies the following condition,

$$||Ask||F /||Ask|| \geq q,$$

Where, ||X||F denotes the Fresenius norm of matrix X. Matching the phrase and the label pruning phase, is responsible for discovery of group descriptions depending on a serious observation, whether both abstract concepts and frequent phrases are expressed in the same vector space. How close a phrase or a single term is to an abstract concept is calculated by classic cosine distance. It is denoted by P, a matrix of size $t \times (p + t)$ where, t and p are the number of frequent terms and number of frequent phrases used respectively. Vector 'mi' can be calculated using,

$$mi = UiT P.$$

The phrase corresponding to the maximum component of the 'mi' vector must be selected as the human-readable description of the i^{th} abstract concept. And also, the value of cosine becomes the score of the cluster label candidate. Similar processing is carried out for a single abstract concept that can be extended to the entire 'Uk' matrix. A single matrix multiplication M = UkTP.

Last step is to prune the overlapping label descriptions in label induction. Consider 'V' vector of cluster label candidates and their scores. Another term-document matrix Z is created, where the documents are the cluster label candidates. ZT, after column length normalization Z is calculated, which yields the outcome matrix that are similar between the cluster labels. Finally select columns which cross the threshold of label similarity and discard everything, but keep one cluster label candidate for each row with the maximum score.

3.4 Cluster Content Discovery

The assignment of input documents to the cluster labels are done through the Vector Space Model. The assignment process denotes document retrieval based on the VSM model. Define a matrix in which each of the cluster labels is represented as a column vector. Let the matrix be Q.

$$Let\ C = QTA$$

Where, A is the original term-document matrix for the input documents. The element cij shows the strength of membership of the j^{th} document to the i^{th} cluster of the C matrix. If the element cij exceeds the Threshold of snippet assignment then that document is added to a cluster, which is another control parameter of the algorithm. Documents unassigned to any of the clusters are called "Others". Therefore "others" are artificial clusters.

3.5 Final Cluster Formation

At final stage the clusters are sorted in a specific order for display, based on their score, which is calculated using the following formula,

$$Cscore = label\ score \times ||C||$$

Where, ||C|| - total number of documents assigned to the C cluster.

The scoring function looks simple, but prefers well-described and prefers larger groups over smaller groups, mostly the noisy ones. Lingo algorithm does not follow any cluster merging strategy or hierarchy induction.

Simultaneously when the user logs in, their Facebook activities are retrieved using graph API. The activity data of the user is then segregated into likes and status posts. Then the likes are segregated into specific categories by sending its page name to Wikipedia. These likes are then separated into recent likes and history of likes (other than recent). The statuses and posts are retrieved and are again categorized based on the tags used in the posts and statuses. Then again they are separated into recent posts and history of posts (other than recent). On creating an overall user profile by merging the like based and the status based profile. Then re-ranking is done based on the user profile, and the personalized results are returned to the user.

4. Experimental Setup

There are three systems in total for building personalised search engine. (Figure 3.) depicts how the search results are retrieved and clustered from a common search engine such as Google, Yahoo, Bing, and AltaVista etc. The clustered results are stored into a MySql database.

(Figure 4.) depicts how a user profile is constructed using one's Facebook activities. Based on their activities user profile which represents interest score for each category is constructed .These user interest score is stored in a separate MySql database. Categories are chosen based on the categories available in Facebook.

(Figure 5.) combines clustered database and user interest score database. It gives user interest score for each URL returned by the clustered database. The URL's are re ranked based on the interest score and displayed to the user.

5. Results and Discussion

In this section, a thorough discussion about the result and performance measures of the proposed system is discussed. Consider a scenario where two user profiles are taken into discussion (Figure 6). First user is more interested in journals whereas the second user is more interested in movies and journals. These data's are obtained from the likes given by each user in their facebook account. Now, when these users give query as 'Education' in Bing search, it returns results about Education levels or Education System in which both users are not interested in it. But the personalized search engine gives search results with respect to user's interest. The bar chart depicts how much user interest is provided in the first result page for etool (Bing) engine and for the proposed personalized search engine. Bing engine gives the same result to both user 1 and user 2. While personalized engine gives higher results about movies to user 2 and it gives higher results about reading journals or newspapers for user 1.

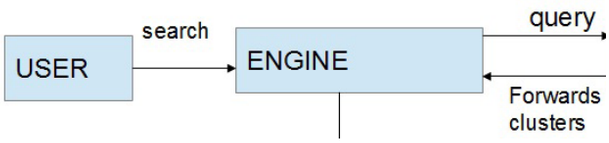

Figure 3. Clustering architecture.

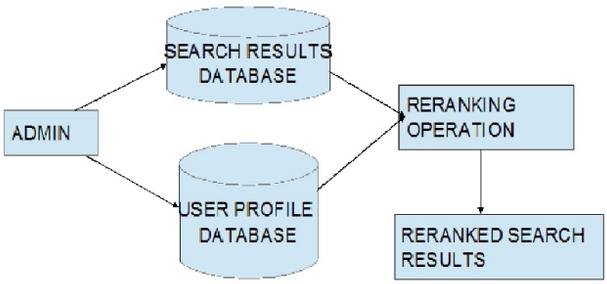

Figure 5. Search result.

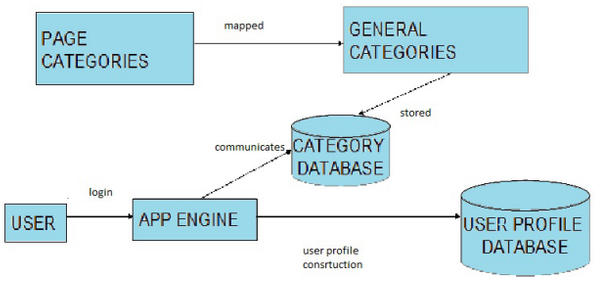

Figure 4. User profile construction.

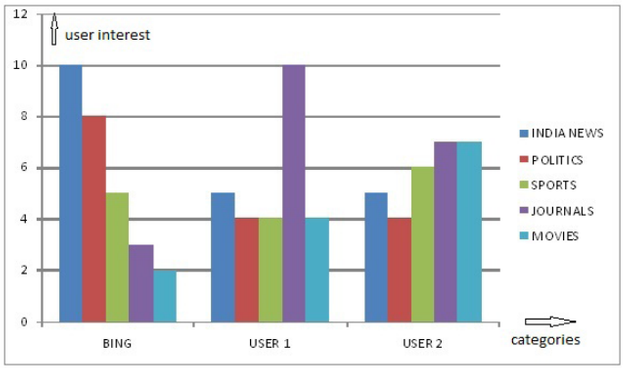

Figure 6. Bar Chart depicting interest relevance for Bing and Personalized engine.

Performance measure depicts that proposed personalized search engine returns different search results to each users. The search results depend on the user interest profile constructed from facebook activity. Thus the personalized engine is both dynamic and interest based.

6. Conclusion and Future Work

The search results obtained when the user enters the search query are re-ranked based on the social networking activities of the user. The user profile is constructed by using the social networking activity of the user by creating a like based user profile. This profile is then used to re-rank and thus the personalized search results are obtained according to user's preference.

In future this work can be extended by extracting status of user and creating status based profiles and the processing speed of the pages can be enhanced by using more specific algorithms because time taken to retrieve

and calculate score for search results is not friendly. More categories can be added to give more specific results on a wider range of fields.

7. References

1. Geetha Rani S, Sorana Mageswari M. A link-click-concept based Ranking Algorithm for Ranking Search Results. Indian Journal of Science and Technology. 2014 Oct; 7(10):1712–9.
2. Rezaei HR, Dehkordi MN, Moghadam RA. Improving performance of search engines based on fuzzy classification. Indian Journal of Science and Technology. 2012 Nov; 5(11):3607–11.
3. Sieg A, Mobasher B, Burke R. Learning ontology based user profiles: a semantic approach to personalized web search. IEEE Intelligent Informatics Bulletin. 2009 Nov; 8(1):7–18.
4. Radlinski F, Joachims T. Evaluating the robustness of learning from Implicit Feedback. ICML Workshop on Learning in Web Search; 2005. p. 42–50.
5. Xu Z, Luo X, Zhang S, Wei X, Mei L, Hu C. Mining temporal explicit and implicit semantic relations between entities using web search engines. Future Generat Comput Syst. 2014; 37:468–77.
6. Carmel D, Zwerdling N, Guy I, Ofek-Koifman S, Har'el N, Ronen I, Uziel E, Yogev S. Personalized Social Search Based on the User's Social Network. Proceedings of the 18th ACM Conference on Information and Knowledge Management; 2009. p. 1227–36.
7. Prates C, Fritzen E, Siqueira S, Helena M, de Andrade LCV. Contextual web searches in Facebook using learning materials and discussion messages. Journal Computers in Human Behavior. 2013; 29(2):386–94.
8. Lu Q, Conrad JG, Al-Kofahi K, Keenan W. Legal document clustering with built-in topic segmentation. Proceedings of the 20th ACM International Conference on Information and Knowledge Management; 2011 Oct. p. 383–92.
9. Teitler BE, Sankaranarayanan J, Samet H, Adelfio MD. Online document clustering using GPUs. New Trends in Databases and Information Systems. Springer International Publishing; 2014. p. 245-254.
10. Available from: http://en.wikipedia.org/wiki/Carrot2
11. Waterworth A. New South Wales, Sydney: University of Sydney. Available from: http://clusteringalgorithms.blogspot.in/2007/07/lingo-algorithm.html