Journal of Computer and Communications, 2023, 11, 89-101
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Ranking of Web Pages in a Personalized Search

Mahmoud Abou Ghaly[1,2]

[1]Mathematics Department, Faculty of Science, Ain Shams University, Cairo, Egypt
[2]Information Technology Department, Faculty of Computer Science and Information Technology, Al-Baha University, Al-Baha, KSA
Email: maboughaly@bu.edu.sa

How to cite this paper: Ghaly, M.A. (2023) Ranking of Web Pages in a Personalized Search. *Journal of Computer and Communications*, **11**, 89-101.
https://doi.org/10.4236/jcc.2023.112007

Received: January 31, 2023
Accepted: February 25, 2023
Published: February 28, 2023

Abstract

The basic idea behind a personalized web search is to deliver search results that are tailored to meet user needs, which is one of the growing concepts in web technologies. The personalized web search presented in this paper is based on exploiting the implicit feedbacks of user satisfaction during her web browsing history to construct a user profile storing the web pages the user is highly interested in. A weight is assigned to each page stored in the user's profile; this weight reflects the user's interest in this page. We name this weight the relative rank of the page, since it depends on the user issuing the query. Therefore, the ranking algorithm provided in this paper is based on the principle that; the rank assigned to a page is the addition of two rank values R_rank and A_rank. A_rank is an absolute rank, since it is fixed for all users issuing the same query, it only depends on the link structures of the web and on the keywords of the query. Thus, it could be calculated by the PageRank algorithm suggested by Brin and Page in 1998 and used by the google search engine. While, R_rank is the relative rank, it is calculated by the methods given in this paper which depends mainly on recording implicit measures of user satisfaction during her previous browsing history.

Keywords

Implicit Feedback, Personalized Search, Web Page Ranking, User Profile

1. Introduction

The World Wide Web (WWW) holds a huge amount of information, it may hold hundreds of billions of web pages. Search engines have been extensively used by the people to search for specific information on the web. To offer the most relevant results, the search engines discover, understand, and organize the internet's content. These are arranged into three primary functions of the search

engine: crawling, indexing, and ranking. Ranking of the search results is an essential part of the search process, since the search engine may return thousands or even hundreds of thousands of pages upon a user query. And the user can't follow all such huge numbers of results.

Many algorithms for web page ranking have been proposed based on different assumptions. The algorithms proposed include PageRank [1] [2], Weighted PageRank [3] [4], BrowseRank [5], HITS [6] [7] and others.

PageRank focuses on ranking a page based on its absolute importance. This importance is mainly affected by the importance of the pages that site this page (named backlinks), therefore the rank assigned to a page using the PageRank algorithm is fixed for all users with respect to the same query. We name it an absolute value of the rank.

Whereas the BrowseRank focuses on ranking a page based on the value it provides to the user. It exploits the users browsing behaviour to determine page importance; but even if it gives more accurate results than PageRank algorithm. It is still like the PageRank, where the ordering of the pages is generic to all users, not personalized to individual users. This is different from the approach we present in this paper where the search results are tailored to meet different user's need. We will exploit the implicit feedbacks of user satisfaction during her browsing history to give a personalized search.

In our approach, search results are specified exactly by the search engine and they will be the same for all users issuing the same query, except they are different in the ordering of such results. Ordering of these results is different from user to user according to the relative rank of the pages stored in the user profile. Users' profiles were constructed earlier from previous users browsing history.

Therefore, the rank we suggest for a page is the addition of its absolute rank and its relative rank. The absolute rank is a measure of the page reachability whereas the relative rank is a measure of the page utility. The reachability of a page denotes the possibility that a user or a random surfer reaches the page (which is a generic value fixed for all users); whereas utility of the page represents the value the page gives to a user (which is different from user to user according to user interests).

Calculating page importance based on page reachability is the approach followed by the PageRank Algorithm. It exploits the link graph of the web to build a model to calculate page significance based on that graph. Actually, the ordering of the search results also affected by the query keywords (some pages may be more related to the query keywords than other pages). In our approach we consider this dependence within the absolute rank since it is also calculated by the search engine, and it does not depend on the particular user issuing the query. Also, the approach presented in this paper will take into consideration user's positive preferences as well as negative preferences. We will assign a negative relative rank to negative preferences, therefore pages with negative preferences relatively get to the back in the search results page.

2. Related Work

There are several attempts to define a typical search engine that return the same results regardless of who submitted the query, PageRank algorithm, and the BrowseRank are two of the most pioneering ones. Surgey Brin and Larry Page [1] [2] developed a ranking algorithm used by Google, named PageRank (PR). They exploit the link structure of the web to determine the importance of web pages. This algorithm states that if a page has some important incoming links to it then outgoing links to other pages also become important. Therefore, it takes backlinks (pages that cite this page) into account and propagates the ranking to the page outlinks (pages that are cited by this page). Thus, a page obtains a high rank if the sum of the ranks of its backlinks is high.

Another important typical ranking algorithm is the BrowseRank Algorithm, presented in [5]. It exploits the user browsing behaviour during a specified period of times to calculate the page importance by first constructing users browsing graph. If a user transferred to a page q after clicking a hyperlink in page p then there is an edge directed from page p to page q. Continuing in this manner for a certain number of users will construct the users browsing graph from the actual users browsing history. Also, the time spent by each user on a page is recorded which normally referred as the page dwell time. Therefore, according the BrowseRank algorithm; the page importance depends on the more visits of the page made by the users and the longer time periods spent by the users on the page.

Therefore, the BrowseRank approach for ranking web pages exploits users' behaviour to determine page importance; but even if it gives more accurate results than PageRank algorithm. It is still like the PageRank a typical ranking, since the ordering of the pages is fixed to all users issuing the same query, not personalized to individual users, *i.e.*, it gives the same results to all the users. This is different from the approach we present in this paper where the search results are tailored to meet different users need.

We will exploit the implicit feedbacks of user satisfaction during her browsing history to give a personalized search based on a pre calculated relative rank. The idea behind the relative rank of a page is that: the relative rank is a measure of the user interest or preference on this page which must be reflected on future browsing of the user. By adding this relative rank to the absolute rank shows a more accurate rank of the page.

There are many other attempts to define a personalized search engine as in [8] [9] most of them depends mainly on constructing a user profile from the user browsing history, where the user profile contains a glossary of the terms the user highly interested in; these set of terms are extracted from the web pages the user highly interested in.

As indicated in [8] the glossary of terms helps in ranking future user browsing by calculating the similarity between each page and the glossary of terms. Our approach is simpler than these approaches since we do not construct the glossary of terms and don't calculate the similarity between each page and the glos-

sary of terms. We just calculate the relative rank of the page and store this value with page URL in the user profile, so that it can be added to the absolute rank of the same page to in increase the rank value of the page during future user browsing.

H. Kumar *et al.* in [10] have defined a method for a personalized web search based on constructing an index of the anchor text retrieved from the web pages that the user has clicked during her past browsing. They proposed a weight computation method that assigns different values to anchor texts according to the browsing behaviours "clicking" or "downloading".

There are several attempts to construct a recommender system based on implicit measures of user satisfactions during her browsing history, e.g., in [11] [12]. Also, the approach we present in this paper for measuring user satisfactions could be a basis for constructing a recommender system for user searching the web, which could be arranged in future work.

Query logs can also be considered as one of the most valuable tools for search engine optimization. Every search engine maintains a log of what users search on it including user ID, query, clicked URLs, rank of URLs and time of access. Thus, this huge amount of information from query logs can provide a good insight about user browsing behaviour and user information needs. S. Kataria and P. Sapra in [13] emphasised a new method for mining search engine query logs for rank optimization and to get fast recommendation in terms of its related queries on a large scale.

Shen and *et al.* in [14] proposed a decision theoretic framework for modelling implicit profiling, they developed an intelligent client-side web search agent (UCAIR) that can perform eager implicit feedback, e.g., query expansion based on previous queries and immediate result reranking based on click through information.

K.W.-T. Leung and D.L. Lee discussed in [15] that, profiles which capture and utilize both user's positive and negative preferences perform the best. They showed negative preferences can increase the separation between similar and dissimilar queries. They extracted concepts from Web-snippets to create accurate and up-to-date user profiles.

As discussed in [16] some other systems provide personalized results, e.g., systems in which users register their interest or preference information, and systems that recommend information based on user feedbacks or views rating. In such systems users may provide feedbacks on relevant or irrelevant judgements, ratings on a scale from 1 (very bad) to 5 (very good), and so on. Which are not interesting for most users, since they prefer easier methods? Usually, users are reluctant to explicitly provide their preferences due to the extra manual effort involved.

3. Implicit Measures of User Satisfactions during Web Browsing

There are several implicit indicators for user interest during her browsing of a

web page; They include direct ones e.g., bookmarking, printing, saving the page, copying some page text, downloading files, watching videos, and displaying photos, …etc., and the derived ones such as dwell time. As discussed in [17] [18] the dwell time is a very important indicators for a user interest in a page. It is the total time spend by the user on a page it starts as soon as the user clicked the page URL in the search results page and end as soon as the user returns back to the search results page.

The system proposed in this paper collects such implicit measures for user satisfaction to specify a weight value signifying the user interest of the page. We refer to this weight value, relative rank of the page since it is different from user to user which is opposed to the absolute rank of the page which is fixed for all users since it depends only on the link structure of the web besides the keywords of the issued query.

The suggested ranking system assigns a rank to the page equal to the relative rank plus the absolute rank as:

$$page\_Rank = R\_rank + A\_rank;$$

where R_rank is the relative rank value; it is assigned to the page by the intended user. While A_rank is the absolute rank value; it is assigned to the page by the search engine. Some bit related to the work presented in this paper, Joachims in [19] collected implicit measures in place of explicit measures, introducing a technique to learn a ranking function, but based only on clickthrough data. Also, Kelly and Teevan in [20] while developing valuable insights into implicit relevance measures, results were not applied to improve the ranking of web search results in realistic settings.

4. The System Architectures

In our approach instead of constructing a glossary of the terms that interests the user after deduction from web pages that interests the user, and storing them in the user profile, as suggested in [8] [9]. The user profile consists of a set of clusters; each cluster stores a set of keywords deduced from the issued query; and a set of URL's for pages browsed by the user in response to the issued query. Thus, the clusters record only the pages displayed in the search results and clicked by the user.

We use the java language to formalize the basic definitions in the system, which are the definition of the object user_profile, Cluster, and the object browsed_pages. They are summarized in Table 1.

The definitions listed in Table 1 show that:

- The user_profile object can hold any number of the Clusters objects.
- The Cluster object can hold any number of browsed_pages
- Each browsed_page object holds the web page URL, the relative rank assigned to this page, and the date of browsing this page, so it may be displayed to the user beside the page URL in the final search results page.
- Besides the set of clusters, the user_profile object holds the following data:

Table 1. Basic definitions.

```
class browsed_pages
{
String url;
double R_rank;              // is the relative rank assigned to the page with URL url
Date browsing_date;         // is the date of browsing the page
browsed_pages next_page;    // a reference to another page in the cluster
}
```

```
class Cluster
{
String keyWords[100];              //set of keywords deduced from the search query
Cluster (String words) {keyWords = words}          // class constructor
browsed_pages pages;               // a reference to the set of browsed pages
Cluster next_cluster;              // a reference to another cluster in the user profile
}
```

```
class profile
{
String user_name;
long ID;
Cluster cluster_ref;          // a reference to a cluster object
}
```

- A username, so it can be shown on top of the search results page, we can also add a photo of the user,
- A user ID which is a unique integer number to distinguish the user among all other users, it is automatically generated by the system.

We may visualize user profile as in **Figure 1**:

5. The System Procedures

The system procedures start when the user issues a new query. Assume the keywords of such query are the set **kwords** then, the system procedures proceeds in steps as follows: where step1 and step2 could be performed in parallel, and either step 3 or step 4 is executed.

Step 1: The search engine proceeds in the usual way and determines the **search_results_page** which contains the set of pages relevant to the issued query, ranked by the absolute rank.

Step 2: The search engine triggers the method **keyword_similarity** from the user profile to determine the cluster that best matches the keywords **kwords** of the issued query which is the cluster that represents the same kind of information.

Step 3: If no cluster returned by **keyword_similarity** meaning that no cluster satisfies the similarity conditions with the issued query and the issued query is considered a new query for the user. Therefore, the pages of the **search_results_page** are displayed to the user ranked by the absolute rank. We do the following to record the user behaviour during her browsing of the pages of such query.

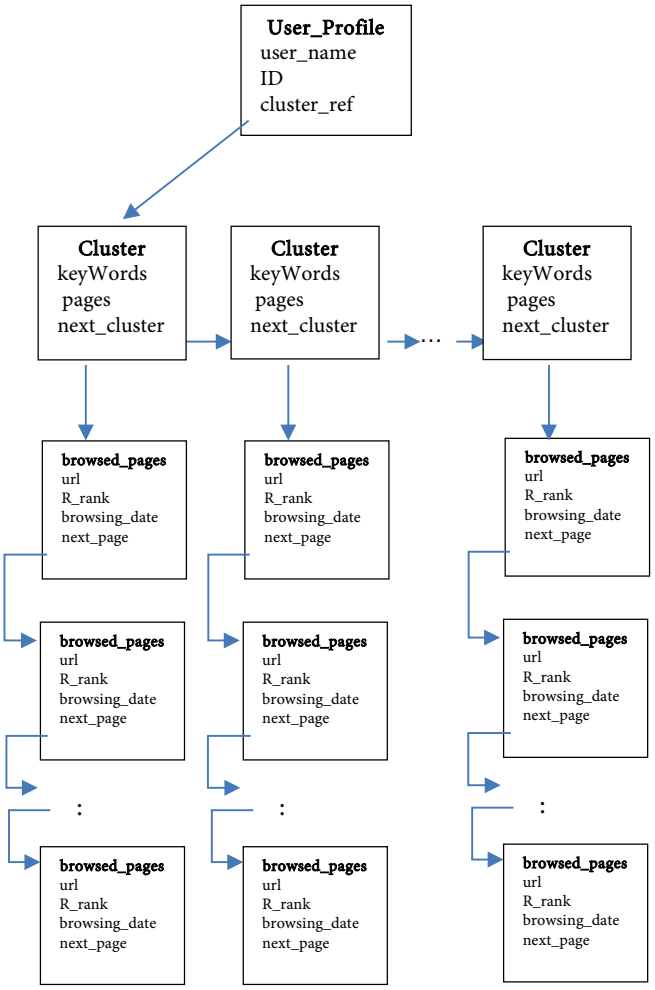

Figure 1. Sketch of user profile.

- An empty cluster object is created as:

Cluster new_cluster = new Cluster (canonical_form (kwords));

- The new_cluster is added to the set of clusters in the user profile.

- Then the method **record_behaviour** is called with inputs the new_cluster object to record the behaviour of the user during her browsing of the search_results_page, where the browsed_pages are added to the new_cluster.

Step4: Otherwise, assume the cluster best_match_cluster is the cluster returned by the method **keyword_similarity**, then the relative rank of each page stored in this cluster is added to the absolute rank of the same page of the search_results_page returned by search engine, which reflects the user interest of such pages. Then the search_results_page are displayed to the user ordered according to this new ranking. Also, the user's behaviour during her browsing of such search results must be recorded. Such recording of behaviour should be added to the best_match_cluster. This is done by calling the method **record_behaviour** with the inputs best_match_cluster. Before this, the set of keyWords of the best_match_cluster may need some update to record some new keywords of the given query as:

best_match_cluster.keyWords = best_match_cluster.keyWords U canonical_form (kwords).

The method **canonical_form** () makes a given set of keywords in a canonical form by removing words such as: we, you, they, and, the, therefore, thus, between, together, …etc.

5.1. The Method Keyword_Similarity

The method **keyword_similarity** calculates the similarity percentage between the keywords of each cluster and the keywords kwords of the issued query and returns the cluster with the highest similarity value provided that its similarity value ≥ α for a threshold α, best chosen ≥ 1/2. If no cluster has similarity value ≥ α then no cluster is returned. Formally, the similarity value between two keyword sets X and Y is calculated using the following formula.

$$\text{keyword\_similarity}(X,Y) = \left(\left|X \cap Y\right| / Y\right)$$

where $|Y|$ is the number of elements of the set Y, and $|X \cap Y|$ is the number of elements common to X and Y.

Therefore, X and Y are similar if **keyword_similarity** (X, Y) ≥ α for a threshold $1/2 \le \alpha \le 1$ (in this case; more than half of the keywords of Y are in X). Otherwise, they are not similar.

5.2. The Method Record_Behaviour

The method **record_behaviour** records in each given cluster the browsed behaviour of the user corresponding to her issued query. Following is a sketch of the body of such method;

```
record_behaviour (Cluster A_cluster)
{
while Is_open (search_results_page) loop
        // the search results page is loaded on the memory
if (search_results_page.new_clicked_URL) // the user clicked the
        //new_clicked_URL from the search_results_page
then
A_cluster.browsed_pages.add (new_clicked_URL)
Is_Bookmarked = Is_Saved = Is_Printed = 1;          // initial values
Number_of_copies = Number_of_clicks = 0;            // initial values
Dwell_time = Current_Time;
A_cluster.browsed_pages.new_clicked_URL.R_rank =
calculate_relevance (new_clicked_URL);
else // the user is thinking
end if;
end loop;
}       // end of record_behaviour
```

Once a page is clicked its URL is added to the set of browsed pages by the method add; and method calculate_relevance is triggered to calculate the relative

rank of such pages. The relative rank is stored in the R_rank field of the page.

5.3. The Method Calculate_Relevance

The method **calculate_relevance** monitors and records the implicit behaviours of the user during her acquisition of the page. Table 2 shows a list of the activities recorded by the**calculate_relevance** method and the suggested variables used to record such activities. Also, Table 2 shows the weight we assign for each activity.

We will adopt the following claims for recording the user interests during her browsing activities.

- The user shows the same interest value when he bookmarks, saves, or prints a page; assume this value is the constant R. The value of the constant R must be determined by the search engine to be in conformance with absolute rank.
- The interest value of the activity: copying of some text = R/3.
- The interest value of the activity: clicking of some page entities for downloading or viewing = R/2.
- The interest value of the activity: Dwell time (1/2…3 minutes) = R.
- The interest value of the activity: Dwell time > 3 minutes = 2*R.
- When the Dwell time activity < 1/2 minute then it does not show any interest of the user.

Following is a sketch of the body of the method **calculate_relevance()**.

```
double calculate_relevance (URL url)
{ double weight;
while not_Is_open(search_results_page)    // away from the results
                                          // page means the user browsing
                                             some page loop
if (text_is_copied) {Number_of_copies ++; text_is_copied = FALSE;} // what
is copied should be a text
if (URL_is_clicked) {Number_of_clicks ++; URL_is_clicked = FALSE;}
                  // what is clicked should be a URL
end loop; // the user clicked the back button until reached the
search_results_page again
if (url.print()) Is_Printed = 1;     //the user printed the page
if (url.bookmark()) Is_Bookmarked=1;      //the user bookmarked the page
if (url.save()) Is_Saved = 1;        //the user saved the page
Dwell_time = Current_Time - Dwell_time;         // in minutes
if (Dwell_time < 1) dw = 0;
else if (½=<Dwell_time =<3) dw = 1;
else if (Dwell_time >3) dw = 2;
weight = (Is_Printed + Is_Bookmarked + Is_Saved + Number_of_copies/3 +
Number_of_clicks/2+dw)*R;
If (weight == 0) weight = - € // for a threshold € specified by the search engine
return (weight);
}
```

Table 2. Measures of user activities.

Activities	Suggested_Variables	Weight
Bookmarking the page	Is_Bookmarked	R
Saving the page	Is_Saved	R
Printing the page	Is_Printed	R
How many times the user copied some page texts.	Number_of_copies	Number_of_copies /3
How many times the user clicked some page entities for download or view, e.g. photo, video, files, URL's…etc.	Number_of_clicks	Number_of_clicks /2
The Dwell time that is the total time spent by the user comprehending the page.	Dwell_time	(1/2…3) m → R >3 m → 2R

Since recording of the user interests during her browsing of a page starts once the user clicked the URL of the page in the **search_results_page** and stopped once the user returns to the **search_results_page**. Therefore, this system for recording the user interests is accumulative; meaning that in recording the user activities in a certain page we also record the activities in all pages the user will transfer to by clicking hyperlinks in that page. E.g., we will count the Number_of_copies of a text, the Number_of_clicks of an entity, and the Dwell_time from the current clicked page and all the pages accessed from it.

Our approach will also take into consideration the pages the user not interested in (pages with zero weight), since the method **calculate_relevance** assigns a negative relative rank to such pages. Which implies their order will relatively get to the back in the search results displayed to the user.

6. An Optimization to the User Profile Construction

Offline on the server side, the search engine can perform an optimization to the user profile; by merging clusters with similar browsed pages (common clicked URL's). The routine **Cluster_ Similarity_Analyzer** will merge similar clusters where similarity between clusters is based on the following criteria. Two clusters are similar if their browsed pages are similar (common clicked URL's). There is no need to also check for the similarity of keywords since similarity of keywords will yield similarity of the browsed pages. The similarity value of two clusters is the similarity value of their respective sets of browsed pages. The similarity value between two sets of browsed pages is based on the following criteria.

6.1. Criteria of Similarity between Browsed Pages (Common Clicked URL's)

The similarity value sim(P_1, P_2) between two sets of **browsed_pages** P_1 and P_2 is computed by the formula:

$$\text{sim}(P_1, P_2) = |P_1 \cap P_2| / (|P_1| + |P_2|)$$

where $|P_i|$ is the number of elements of the set P_i, i = 1 or 2, $P_1 \cap P_2$ are the

common elements between P_1 and P_2, and two pages are equal if their URL's are equal.

We said that two sets of browsed pages are similar iff their similarity value $\geq \beta$ for some threshold $1/2 \leq \beta \leq 1$.

void Cluster_Similarity_Analyzer ()
{
repeat
(C1, C2) ← Find two candidates clusters for merging.
C1 ← merge_clusters (C1, C2).
until there is no similar clusters.
}

Two clusters are candidates for merging if they have the maximum similarity value among all similar clusters. The routine **Cluster_Similarity_Analyzer** will invoke the routine **merge_clusters** to merge two similar clusters into the bigger cluster.

6.2. Merging Clusters

Assume C1 and C2 are two similar clusters; and C1 have a bigger size (*i.e.* have more browsed pages), then the result of the merging will be stored in C1, As

C1 ← merge_clusters (C1, C2).
cluster merge_clusters (C1, C2)
{
C1.keywords ← C1.keywords ∪ C2.keywords
if p ∈ C1.browsed_pages ^ p ∈ C2.browsed_pages then
C1.browsed_pages.p.R_rank ← C1.browsed_pages.p.R_rank + C2.browsed_pages.p.R_rank
if p ∈ C2. browsed_pages ^ p ∉ C1. browsed_pages then add p and its relative rank to C1.browsed_pages.
return C1
}

Note that: the method merge_clusters will generate clusters for all similar kinds of queries, since all must have a common clicked URL's.

7. Conclusions

In this paper we have presented an approach for a personalized web search; where personalization is achieved in future browsing of the user by assigning a higher rank to the pages the user browsed previously and showed an interest or preference on these pages. Our system is so simple compared to other systems for a personalized web search, since it just constructs a user profile storing such pages with a weight assigned to each page. This weight reflects the user's interest or preference on that page, since it is derived from implicit feedback of user satisfaction during her web browsing history.

The work presented in this paper has a strong relationship with many other research areas e.g.:

- In web mining; especially in web usage mining (WUM), which is used to discover user navigation patterns and useful information from the web data present in server logs.
- In Recommender system, it could be the first step in building a recommender system for a user navigating the web based on her browsing history.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Page, L. and Brin, S. (1998) The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Proceedings of the* 7*th International World Wide Web Conference*, Brisbane, 14-18 April 1998, 107-117. https://doi.org/10.1016/S0169-7552(98)00110-X

[2] Page, L., Brin, S., Motwani, R. and Winograd, T. (1999) The PageRank Citation Ranking: Bringing Order to the Web. Technical Report, Stanford Digital Libraries SIDL-WP-1999-0120, 161-172.

[3] Tyagi, N. and Sharma, S. (2012) Weighted Page Rank Algorithm Based on Number of Visits of Links of Web Page. *International Journal of Soft Computing and Engineering* (*IJSCE*), **2**, 441-446.

[4] Tuteja, S. (2013) Enhancement in Weighted PageRank Algorithm Using VOL. *IOSR Journal of Computer Engineering* (*IOSR-JCE*), **14**, 135-141. https://doi.org/10.9790/0661-145135141

[5] Liu, Y., Gao, B., Liu, T.-Y., Zhang, Y., Ma, Z., He, S. and Li, H. (2008) BrowseRank: Letting Web Users Vote for Page Importance. *SIGIR*'08: *Proceedings of the* 31*st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, July 2008, 451-458. https://doi.org/10.1145/1390334.1390412

[6] Kleinberg, J.M. (1999) Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, **46**, 604-632. https://doi.org/10.1145/324133.324140

[7] Li, L., Shang, Y. and Zhang, W. (2002) Improvement of HITS-Based Algorithms on Web Documents. *WWW*'02: *Proceedings of the* 11*th International Conference on -World Wide Web*, Honolulu, 7-11 May 2002, 527-535. https://doi.org/10.1145/511446.511514

[8] Desu, H.S., Paladugu, P., Adibhatla, S.S., Sorda, S.S. and Sudeep, K.S. (2018) Personalized Web Search. *Proceeding of ICACIE* 2018 *Advanced Computing and Intelligent Engineering*, **2**, 201-211. https://doi.org/10.1007/978-981-15-1483-8_18

[9] Takano, K. and Li, K.F. (2010) An Adaptive E-Learning Recommender Based on User's Web-Browsing Behaviour. *Proceedings of the* 2010 *IEEE International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, Fukuoka, 4-6 November 2010, 123-131. https://doi.org/10.1109/3PGCIC.2010.24

[10] Kumar, H., Park, S., and Kang, S. (2008) A Personalized URL Re-Ranking Methodology Using User's Browsing Behaviour. *KES-AMSTA*'08: *Proceedings of the* 2*nd KES International Conference on Agent and Multi-Agent Systems: Technologies and Applications*, Incheon, 26-28 March 2008, 212-221.

[11] Makwana, K., Patel, J. and Shah, P. (2017) An Ontology Based Recommender Sys-

tem to Mitigate the Cold Start Problem in Personalized Web Search. *Proceedings of Information and Communication Technology for Intelligent Systems* (*ICTIS* 2017), **1**, 120-127. https://doi.org/10.1007/978-3-319-63673-3_15

[12] Gudla, S.K., Bose, J. and Sane, K.R. (2018) Enhanced Service Recommender and Ranking System Using Browsing Patterns of Users. *Proceedings of* 2019 16*th IEEE Annual Consumer Communications & Networking Conference* (*CCNC*), Las Vegas, 11-14 January 2019, 1-2. https://doi.org/10.1109/CCNC.2019.8651758

[13] Kataria, S. and Sapra, P. (2016) A Novel Approach for Rank Optimization Using Search Engine Transaction Logs. *Proceedings of the* 2016 3*rd International Conference on Computing for Sustainable Global Development* (*INDIACom*), New Delhi, 16-18 March 2016, 3387-3393.

[14] Shen, X., Tan, B. and Zhai, C. (2005) Implicit User Modeling for Personalized Search. *CIKM*'05: *Proceedings of the* 14*th ACM International Conference on Information and Knowledge Management*, Bremen, 31 October-5 November 2005, 824-831. https://doi.org/10.1145/1099554.1099747

[15] Leung, K.W.T. and Lee, D.L. (2010) Deriving Concept-Based User Profiles from Search Engine Logs. *IEEE Transactions on Knowledge and Data Engineering*, **22**, 969-982. https://doi.org/10.1109/TKDE.2009.144

[16] Sugiyama, K., Hatano, K. and Yoshikawa, M. (2004) Adaptive Web Search Based on User Profile Constructed without Any Effort from Users. *WWW*'04: *Proceedings of the* 13*th International Conference on World Wide Web*, New York, 17-20 May 2004, 675-684. https://doi.org/10.1145/988672.988764

[17] Kim, Y., Hassan, A. White, R.W. and Zitouni, I. (2014) Modelling Dwell Time to Predict Click-Level Satisfaction. *WSDM*'14: *Proceedings of the* 7*th ACM International Conference on Web Search and Data Mining*, New York, 24-28 February 2014, 193-202. https://doi.org/10.1145/2556195.2556220

[18] Liu, C., White, R.W. and Dumais, S. (2010) Understanding Web Browsing Behaviours through Weibull Analysis of Dwell Time. *SIGIR*'10: *Proceedings of the* 33*rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Geneva, 19-23 July 2010, 379-386. https://doi.org/10.1145/1835449.1835513

[19] Joachims, T. (2002) Optimizing Search Engines Using Clickthrough Data. *KDD*'02: *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining* (*SIGKDD*), Edmonton Alberta, 23-26 July 2002, 133-142. https://doi.org/10.1145/775047.775067

[20] Kelly, D. and Teevan, J. (2003) Implicit Feedback for Inferring User Preference: A Bibliography. *ACM SIGIR Forum*, **37**, 18-28. https://doi.org/10.1145/959258.959260