Location Aware Keyword Query Suggestion Based on Document Proximity

Ashwini Kulkarni[1] ,Prof.(Dr).N.R.Wankhade[2]

[1] Student, Computer Department,late G.N.Sapkal coe, Maharashtra,India
[2]Assoc Professor, Department,late G.N.Sapkal coe, Maharashtra,India

ABSTRACT

The Keyword suggestions are the most basic feature of the search engine. Naive users don't know how to express their queries correctly most of the time queries are short and unambiguous; keyword suggestion in web search assists users to access relevant information. Information will be more relevant if location of the users and user preferences are considered. The location aware keyword (LKPS) query suggestion method helps retrieve documents which relates to information provides by user, location where the user is located and the user profile defines the user interest. The user preference specific suggestion helps to improve the suggestions. The search results' relevance is known to be correlated with the spatial proximity of the user and user preferences. The system proposes location-aware and user preferences specific keyword query suggestion framework. The framework introduces graph techniques and generate KD graph and keyword specific graph partitions. The partition algorithm is proposed to retrieve query suggestions. This algorithm follows Random walk with restart technique on candidate nodes in graph partitions and hence improves system efficiency. The results are conducted on AOL dataset of various sizes and evaluate the system performance.

Keywords- *Query suggestion, spatial databases, location-based services, user preferences, graph*

1. INTRODUCTION

Keyword suggestion is the most important and the basic feature. The Major issue in today's web search is queries entered by users are very short and ambiguous. Users try multiple queries and still don't get the required results because keywords suggested by the search engine not very much good descriptor of what actually user needs. Many search engines have made query suggestions techniques for overcoming these problems. Spatial keyword search is very important and will precisely provide keyword suggestions. Location-aware Keyword query Suggestion framework is useful. This proposed method is based on a query clustering process in which groups of semantically similar queries are identified.

The location aware keyword(LKPS) query suggestion method provide the suggested queries retrieve documents which is related to user information, located near to users location and user preferences based on his profile. In this paper we will study all the techniques which are used for keyword suggestions to help user to retrieve the correct information.

Performing keyword suggestion instantly is important for the applicability of LKPS in practice. However, RWR search has a high computational cost on large graphs. Previous work on scaling up RWR search require pre-computation and/or graph segmentation , and require that the transition probabilities between nodes (i.e., the edge weights) are known beforehand. However, the edge weights of our KD-graph are unknown in advance, hindering the application of all these approaches. To the best of our knowledge, no existing technique can accelerate RWR when edge weights are unknown apriori (or they are dynamic). To address this issue, we present a novel partition-based algorithm (PA) that greatly reduces the cost of RWR search on such a dynamic bipartite graph. In a nutshell, our proposal divides the keyword queries and the documents into partitions and adopts a lazy mechanism that accelerates RWR search. PA and the lazy mechanism are generic techniques for RWR search, orthogonal to LKS, therefore they can be applied to speed up RWR search in other large graphs.
LKPS FRAMEWORK:

Consider a user profile P, supplied query q with initial input kq;
kq can be a single word or a phrase. Assuming that the query issuer is at location l, three intuitive criteria for selecting good suggestions are:

(i) the suggested keyword queries (words or phrases) should satisfy the user's information needs based on kq and
(ii) the suggested queries can retrieve relevant documents spatially close to l.
(iii) the suggested queries should be suited for user's profile P
The proposed LKPS framework captures these three criteria and retrieves relevant documents spatially close to l.

2. REVIEW OF LITERATURE

The location aware keyword(LKS) query suggestion method provide the queries suggestion and retrieve documents which are related to user information and located near to users location.LKS framework construct and use keyword document bipartite graph(KD graph) that connect to keyword queries with their relevant document. LKS adjust weight on edges in KD graph to capture the semantics relevance between keyword queries and spatial distance between document location and user location. For distance calculation the Personalized PageRank(PPR) algorithm is used, it uses Random walk with restart(RWR) on KD graph, starting from user supplied query to find the set of keywords and spatial proximity to the user location. But RWR search has high computational cost on large graph to address this issue; a new portion based algorithm is used to reduce the cost of RWR search. Authors in [1] propose a novel context-aware query suggestion approach which is in two steps. In the offline model-learning step, to address data sparseness, click-through bipartite is clustered in order to summarize queries into concepts. In this approach queries are suggested to the user in location aware manner. User profile also affects the search selection. The user profile preferences are not considered in the LKS system.

Authors in [2] propose a novel query suggestion algorithm based on ranking queries with the hitting time on a large scale bipartite graph. This method captures the semantic consistency between the suggested query and the query given by user. Experiments show time is effective to generate semantically consistent query suggestions. The proposed algorithm and its variations can successfully execute huge queries, accommodating query suggestion.

Author [3] introduced novel, domain-independent and privacy preserving methods for enhancing MF models by expanding the user-item matrix and by imputation of the user-item matrix, using browsing logs and search query logs. The system introduced two approaches to enhance user modeling using data. Authors show that CF systems can be enhanced using Internet browsing data and search engine query logs, both represent a rich profile of individuals' interests. They demonstrate the value of their approach on two real datasets each comprising of the activities of tens of thousands of individuals. The first dataset details the download of Windows Phone 8 mobile applications and the second - item views in an online retail store. Both datasets are enhanced using anonym zed Internet browsing logs.

Author [4] proposed a new query suggestion paradigm, Query Suggestion with Diversification and Personalization that effectively integrate diversification and personalization into one unified framework. In the QS-DP, the suggested queries are successfully diversified to cover different facets of the input query and the ranking of the suggested queries are personalized to ensure that the top ones that align with a user's personal preferences. The system proposed a new representation for query log. The proposed multi-bipartite-graph representation comprehensively captures different kinds of relations between search queries in query log. Based on the multi-bipartite-graph representation, they design two strategies to identify the most relevant suggestion candidate.

Author [5] proposed a method that computes likeness among queries based on "Query- Clicked Sequence" model. This model counts weight of clicked document term by density of documents containing this term on clicked sequence, and filters content of unrelated documents during similarity computation. Based on the characteristics of different concentration on relevant and irrelevant documents occurring on clicked document sequence, this paper proposed a query similarity computing method based on irrelevant feedback analysis, and recommended queries based on this method. This method constructs a relevant term collection for each clicked sequence of one query, from relevant document and computes similarity among queries by relevant term collection offline with recommendation of online queries based on the computation result. Query recommendation based on their method can effectively decrease the negative effect on query similarity computation, and increase accuracy of query similarity computation, therefore increase accuracy of query recommendation, especially for informational queries.

Author [6] developed the QueRIE system for personalized query recommendations. QueRIE monitors the user's querying behavior and finds matching patterns in the system's query log, identifying same kind of users. These queries are used to recommend queries which user may find helpful. The system explore the use of latent factor models when, instead of ratings, the input consists of database-query log data. And explored how latent factor models, and in particular matrix factorization using ALS, affect the quality of the recommendations and computational efficiency of their framework. Such techniques have become very popular in traditional rating-based recommender systems, and in this work authors verified that they capture latent similarities between users and "items" even when the input is not explicit.

Author [7] proposed time aware structured query suggestion which clustered query suggestion along timeline so the user can narrow down his search from a temporal point of view. When the suggested query is clicked the method presents web pages from query-URL bipartite graph. After ranking those according to click count within a particular time period this method helping user to access relevant web pages. It free the users from burden of entering a specific time constraint with query, this method can be used in the context of real user search tasks.

Author [8] designed a location-aware keyword query suggestion framework. They propose a weighted keyword-document graph, which captures both the semantic significance between keyword queries and the spatial distance between the resulting documents and the user location. The graph is browsed in a random-walk-with-restart fashion, to select the keyword queries with the highest scores as suggestions. To make framework scalable, authors propose a partition-based approach that outperforms the baseline algorithm by up to an order of magnitude. And design the first ever Location-aware Keyword query Suggestion framework, for suggestions relevant to the user's information needs that also retrieve relevant documents close to the query issuer's location. Also extend the state-of-the-art Bookmark Colouring Algorithm (BCA) for RWR search to compute the location-aware suggestions.

Problem Formulation:

Keyword query suggestion approaches can be classified into three main categories: random walk based approaches, learning to rank approaches, and clustering based approaches. We also briefly review alternative methods that do not belong to any of these categories. To the best of our knowledge, no previous work considers user location along with user profile in query suggestion.

To design, develop and test keyword suggestion system based on following three factors:

- Keyword relevance
- Location proximity
- User profile preferences

To improve efficiency of query suggestion system.

3.SYSTEM ARCHITECTURE / SYSTEM OVERVIEW

Following figure shows the architecture of system. The architecture is mainly categories in two sections: Initialization and Testing. For initialization phase query logs is given as input to the system. For testing user query with location information is given to the system. The system generates top m query keyword suggestions and relevant documents.

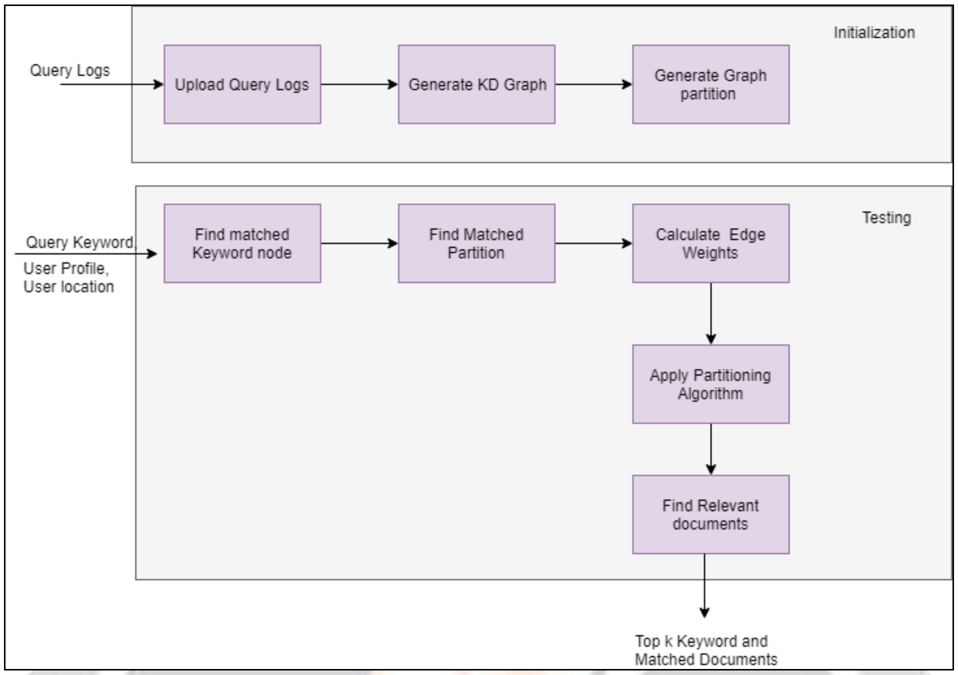

Fig:. System Architecture

3.1 Initialization Phase:
In initialization phase, Query logs are given as input. The log contains query keyword, search result document, query location, and user profile who have fired the query. Using this input, number of distinct keywords and documents are extracted. Let K={k1,k2..kn} be the number of keywords in dataset and D = {d1,d2,..dk} be the number of documents in dataset. Using number of documents set and keyword set KD graph is generated. A document Dj is clicked after the query keyword Ki then there is an edge between query keyword Ki and document Dj. The edge weight is represented based on the number of clicks.

From the generated KD graph, the graph is partitioned based on disjoint keywords called as G^{KP} graph. A query node ki is connected to document partition P_D if k_i connects in G to at least one document in P_D.

3. 2. Testing Phase:
In testing phase user fire a query to the system. The query contains query keywords and location information in terms of <latitude, longitude>. To enhance system result user profile is also considered. The user profile details are fetched form database who has fired the query.

Initially matched nodes are extracted from the dataset. Based on the graph partitions, partitions are selected based on matched keywords. The nodes of matched partitions are again filtered based on the users profile. The keywords present in partitions are loaded form query logs. Using query log information nodes those are matched with users profile are selected and considered for further processing. This graph partitioning strategy and node filtration based on user profile reduces the size of candidate results and hence it improves the efficiency of system.

Weights of all edges starting from the generated candidate keyword nodes are calculated using following formula:
$W(e) = \beta * w(e) + (1- \beta) * (1- dist(\lambda q - dj.\lambda)$

Where β is constant for weight distribution among keyword relevance and distance

W(e) : initial weight of e in the KD-graph

dist(λq –dj.λ) : is the normalized distance between user query location and document location

The distance is calculated using Euclidean distance formula:

$$\sqrt{(\lambda q1 - dj.\lambda 1)^2 + (\lambda q2 - dj.\lambda 2)^2}$$

Where $(\lambda q1, \lambda q2)$ is the co-ordinates of user location and $(dj.\lambda 1, dj.\lambda 2)$ is the co − ordinate of document location.

The normalized distance is calculated by dividing the maximum number of clicks in the query log for any query documents pair.
Using candidate nodes and graph partition the data is again filtered and top m query keyword suggestions are generated using partition Algorithm. The partition algorithm is based on the bookmark coloring algorithm. It follows the Random walk with restart(RWR) technique.

3.3 Partition Algorithm:

- Input : G(D,K,E): Document keyword graph,
- G^{KP} : Keyword Partitioned Graph
- q=(Kq,λ q) Query with KQ keyword and λ location
- m: Count of Keyword Suggestion,
- Ɛ: Active ink threshold

- Output: C: top-m candidate suggestions

Algorithm work:

Processing:

```
1. initialize Priority Queue Q and result set C as ø
2. select partition P Э kq
3. for kw in selected P
4. if kw matches the user profile
5. Add kw in candidate list
6. add P.aink = 1
7. assign AINK =1
8. while $ Q <> ø and Q.top.aink >= Ɛ do
9. Deheap the top entry Pt from Q
10. tm= top-m entry from C
11. tm' = top-(m+1) entry from C
12. if tm.rink > tm'rink + AINK then
13. break
14. Spread the active ink to candidate nodes in Pt
15. for each node v in partition Pt do
16. distratio=1
17. if v is a keyword query node in candidate list then
18. distratio=1-α
19. v.rink= v.rink+ v.aink * α
20.  AINK= AINK -v.aink * α
21. if there exist a copy t of v in C then
22. Remove t from C
23. v.rink= v.rink + t.rink;
24. Add v to C;
25. Get partition set P connected from v in G^KP;
26. for each partition Pi in P do
27. ink= v.aink X distratio X ~{w(v,Pi);
28. if ink+v.acc.Pi>= Ɛ then
29. Pi.aink= ink+v.acc.Pi;
```

```
30. if there exist a copy P'i of Pi in Q then
31. Remove P'i from Q;
32. Pi.aink= Pi.aink+P'i.aink;
33. Else
34. Accumulate ink at node v for Pi(v.acc.Pi);
35. return the top-m entries(excluding kq) in C;
```

4. SYSTEM ANALYSIS AND RESULT

4.1 Implementation:

The system is implemented in java using jdk1.8. A web based system is generated using apache tomcat 6.0.To save query logs mysql 5.3 database is used. A google map api is used for location specific data display.

Datasets:
AOL dataset is downloaded[9] and used. The dataset contains 35,58,412 records. Each record in a dataset contain:
Query, QueryTime, ItemRank, ClickURL, Click Location-latitude, Click Location-longitude
To add profile preferences in a dataset, new last column is generated. The values are assigned to the last column randomly among 2 profiles: student profile and professional profile

Performance Measures:
1: variation in Beta: The beta value defines the importance of keyword relevance and location proximity. As the value of beta increases the query relevance is considered more than the location proximity.
2: Response Time: The response time is captured based on the following two parameters:
1. Number of search keyword: Time required for processing number of search keywords in query.
2. Data Size: Time required for processing the query on dataset. The dataset size is increases gradually and processing time is captured.

4.2: Results

1:Result Extraction by varying constant value:
The following figure 2 and 3 shows query result for existing and proposed system respectively. The query "lottery" is fired on dataset of size 2M. The result is taken for varying the value of β. The three results are captured for β value 0.1,0.5 and 0.9 respectively. As the beta value increases the location proximity impact is get reduced and the results are shown only based on the query keyword relevance. To manage the good tradeoff between the keyword relevance and location proximity the β factor value need to set near to 0.5 .

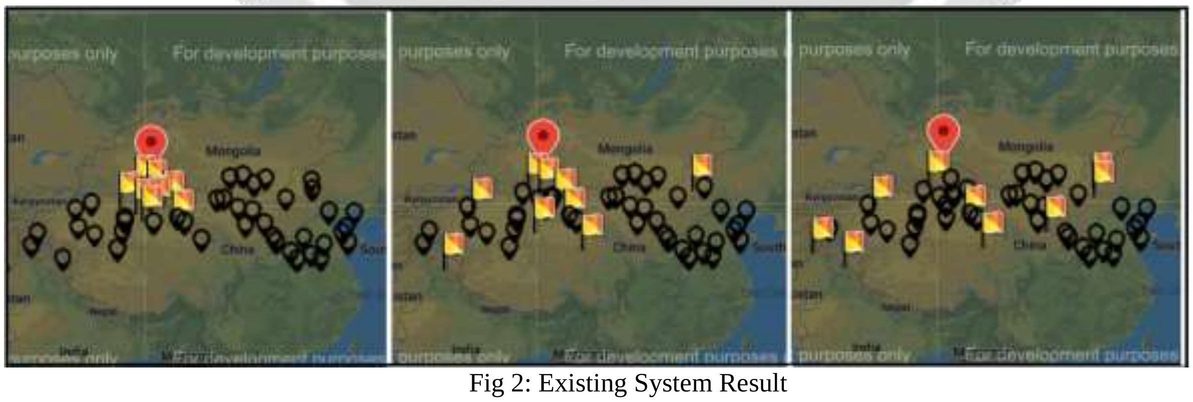

Fig 2: Existing System Result

The proposed system includes the user profile preference selection as well as the query relevance and location proximity. The suggestions are changed based on the users profile value. The profile specific suggestions are more relevant than the existing system suggestions.

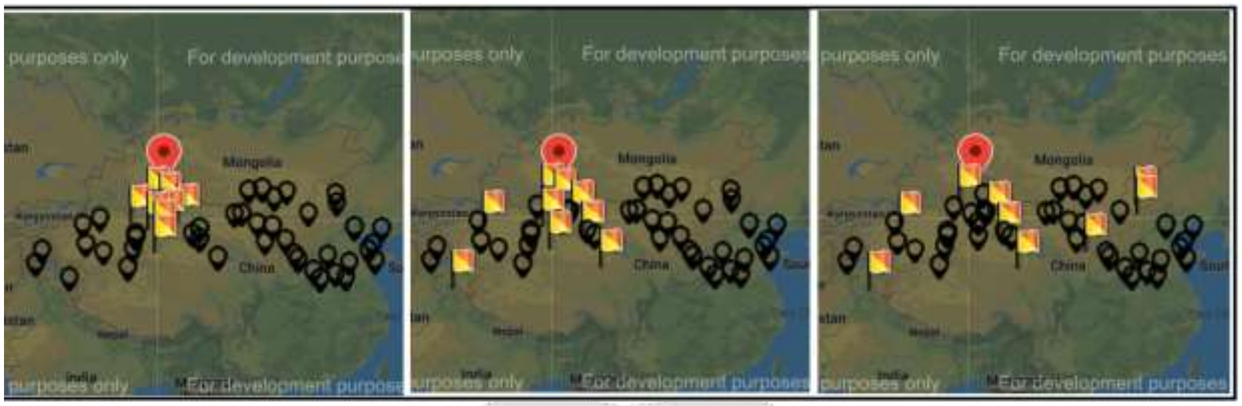

Fig 3: Proposed System Result

2: Response Time for variation in Query Size:

The time is capture for processing various queries on dataset of size 2M. The queries containing single word, 3 words are 5 words are processed and time is captured for result evaluation. As the number of keyword increases the number of matched partition may get increases. It depends on the query. If number of matched graph partitions increases after filtering the relevant keywords then the time required for processing get increases.

The system filters the nodes from graph partition based on user profile. The time required for processing profile preference queries is less as compared to processing of query execution using keyword relevance and location proximity.

Table 1: Response Time for variation in Query Size

Query Keyword	Response Time for Existing system (In seconds)	Response Time for Proposed system (In seconds)
1	1.72	1.47
3	1.84	1.64
5	1.91	1.76

The following graph shows the time required for processing quires of various sizes on 2M dataset. The comparative study between existing and proposed system is done. The proposed system requires less time as compared to the existing system.

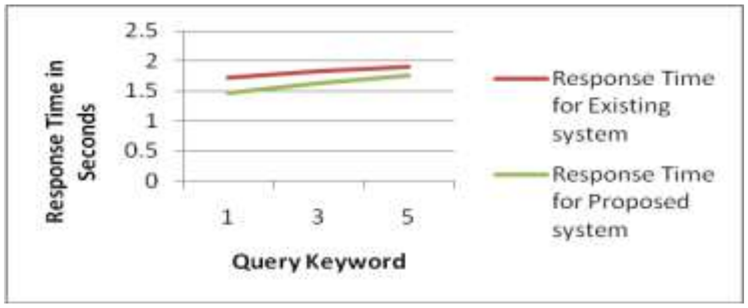

Fig 4: Response Time for variation in Query Size

3: Response Time for variation in Data Size:

The following table shows the time required for processing a single word query over datasets of various sizes. The size of data varies from 0.5 M to 2M. As the dataset size increases the processing nodes are also increases. Due to increase in processing nodes the time required for processing also increases.

The proposed system requires less time as compared to the existing system. The system filters the nodes form graph dataset based on the user profile and generate a small candidate result set. Using this candidate result set query suggestions are generated.

Table 2: Response Time for variation in Data Size

Dataset Size	Query Response for Existing System(In seconds)	Query Response for Proposed system (in seconds)
0.5M	0.6	0.551
1M	0.93	0.88
1.5M	1.24	1.02
2M	1.72	1.47

The following fig. shows the graphical analysis of existing and proposed system based on time. The propose system requires less time as compared to the existing system.

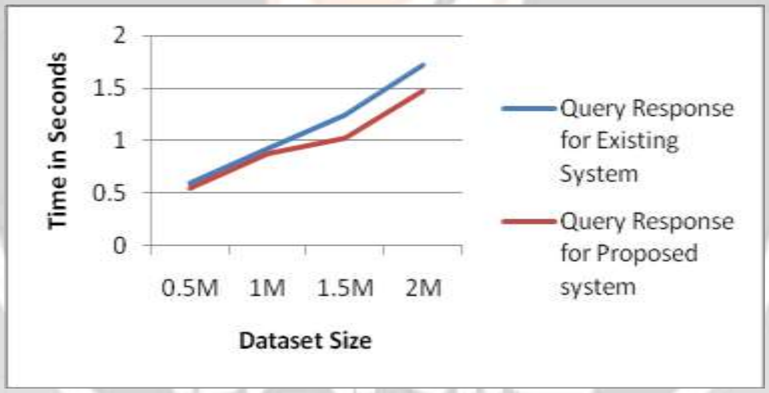

Fig: 5: Response Time for variation in Data Size

5. CONCLUSION

User satisfaction plays very important role in information retrieval. Query recommendation is best method for helping users to satisfy the users' information need by suggesting queries related to current users need. The system proposes LKP framework for keyword query suggestion method. The keyword suggestions are based on keyword relevance, location proximity and user profile preferences. The method follows graph techniques. The document keyword graph is generated from keywords and documents in dataset. The graph partitioning helps to improve system efficiency. To find the result system follows RWR technique and assign weights to graph edges dynamically. This method is useful for big datasets.

In our future work we want to prepare synonym suggestion for given keyword query using offline dictionary interface.

6. REFERENCES

[1] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li, Context-aware query suggestion by mining click-through and session data, in Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2008, pp. 875-883.

[2] Mei, Qiaozhu, Dengyong Zhou, and Kenneth Church. Query suggestion using hitting time. Proceedings of the 17th ACM conference on Information and knowledge management. ACM, 2008.

[3] Royi Ronen and et.al, Recommendations Meet Web Browsing: Enhancing Collaborative Filtering using Internet Browsing Logs, IEEE 32nd International Conference on Data Engineering (ICDE), pp 1230 – 1238, 2016

[4] Di Jiang, Kenneth Wai-Ting Leung, Lingxiao Yang, Wilfred Ng, Query suggestion with diversification and personalization, Knowledge-Based Systems. Volume 89, Pages 553–568, 2015.

[5] Bo Zhang, Bin Zhang, Shubo Zhang, Chao Ma, "Query Recommendation Based on Irrelevant Feedback Analysis", 8th International Conference on Bio Medical Engineering and Informatics, pp 644-648, 2015.

[6] Magdalini Eirinaki , QueRIE reloaded: Using matrix factorization to improve database query recommendations, IEEE international conference on Big Data, pp 1500 – 1508, 2015. .

[7] T. Miyanishi and T. Sakai, Time-aware structured query suggestion, in Proc. 36th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval,pp. 809–812, 2013 .

[8] R. Bhushan and R. Nath. 2012. Recommendation of optimized web pages to users using Web Log mining techniques. Advance Computing Conference (IACC), 2013 IEEE 3rd International. IEEE, 2012.

[9] Goyal, Poonam, and N. Mehala. 2011. Concept based query recommendation. Proceedings of the Ninth Australasian Data Mining Conference-Volume 121. Australian Computer Society. Inc.

[10] Y. Liu, Junwei Miao, Min Zhang, Shaoping Ma, and Liyun Ru. 2011. How do users describe their information need: Query recommendation based on snippet click model. Expert Systems with Applications 38, no. 11 (2011): 13847-13856.

[11] Shuyao Qi,Dingming Wu and Nikos Mamoulis, Location aware keyword Query suggestion based on document proximity, IEEE transaction paper on knowledge and data engineering, Vol 28, No 1, pp 82-97, 2016.

[12] D. Beeferman and A. Berger, Agglomerative clustering of a search engine query log, in Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2000, pp. 407–416.

[13] S. Basu Roy and K. Chakrabarti, Location-aware type ahead search on spatial databases: Semantics and efficiency, in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2011, pp. 361–372.

[14] H. Tong, C. Faloutsos, and J.-Y. Pan, Fast random walk with restart and its applications, in Proc. 6th Int. Conf. Data Mining, 2006, pp. 613–622.

[15] R. Zhong, J. Fan, G. Li, K.-L. Tan, and L. Zhou, Location-aware instant search, in Proc. 21st ACM Conf. Inf. Knowl. Manage., 2012, pp. 385–394.