RDQS: A Relevant and Diverse Query Suggestion Generation Framework

Hai-Tao Zheng and Yi-Chi Zhang

Tsinghua-Southampton Web Science Laboratory,
Graduate School at Shenzhen, Tsinghua University, Shenzhen, China
zheng.haitao@sz.tsinghua.edu.cn, zhangyichi12@mails.tsinghua.edu.cn

Abstract. Traditional query suggestion methods mainly leverage click-through information to find related queries as recommendations, without considering the semantic relateness between queries. In addition, few studies use click-through distribution in diversifying query suggestions. To address these issues, we propose a novel and effective framework to generate relevant and diversified query suggestions. We combine query semantics and click-through information together to generate query suggestion candidates which are highly relevant to original query , we use click-through distribution to diversify the candidates. We evaluate our method on a large-scale search log dataset of a commercial engine, experimental results indicate that our framework has significantly improved the relevance and diversity of suggested queries by comparing to four baseline methods.

Keywords: query suggestion, diversity, click-through information, query semantics.

1 Introduction

With the development of Internet, search engines have become one of the most important way to get information. User inputs queries into the search engine's search blank, then the information retrieval system returns the corresponding results according to the query user issues. However, users are seldom satisfied with the results at the first time. On one hand, users' input is always short, it can hardly reflect their search intent clearly. On the other hand, the queries themselves are ambiguous, search engines may have difficulty in giving right answers to users at the first time. Therefore, how to help users to formulate suitable queries has been recognized as a challenging problem to search engines. To solve this problem a valuable technique, query suggestion, has been employed by most commercial search engines, such as *Google*[1], *Bing*[2] and *Yahoo*![3]

Query suggestion plays an important role in improving the usability of search engines and it has been well studied in academia as well as industry. Traditional

[1] http://www.google.com/
[2] http://www.bing.com/
[3] http://www.yahoo.com/

query suggestion approaches mainly focus on recommending queries relevant to the original query. Due to the redundancy in the results of query suggestion, more and more researches have turned their attention into query suggestion diversifying. However, as an important feature to distinguish different query suggestions, click-through distribution has never been used in query suggestion diversifying work.

In addition, previous query suggestion methods measure similarity relied on either click-through information or query semantics, few of them combine the two factors together to generate query recommendations. In fact, click-through information and query semantics are both important factors in measuring the similarity between queries. Only considering one of them would reduce the relevance of recommendatory results.

In this paper, we propose a Relevant and Diverse Query Suggestion (RDQS) generation framework to produce relevant as well as diverse query suggestions. Specifically, our approach generates suggestion candidates using click-through information, re-ranks them by taking query semantics into account, then diversifies them based on click-through distribution between queries and URLs.

We evaluate our framework for query suggestion using click-through data of a commercial search engine. We measure the performance from different aspects. Empirical experimental results show that our framework can effectively generate highly relevant and diverse query suggestions.

The main contributions of our approach can be summarized as follows:

1. We combine query semantics and click-through information to generate query suggestions, thus the suggestions are not only related in click-through relations but also similar in semantics. query suggestions.
2. Since click-through distribution is an excellent feature to distinguish different kinds of queries, we are the first to apply click-through distribution in diversifying query recommendations.
3. We conduct comprehensive experiments to evaluate RDQS by comparing four baseline methods. Experiments results show that RDQS has significant performance in terms of relevance and diversity.

The rest of this paper is organized as follows: Section 2 presents the related work of query suggestion; Section 3 introduces our RDQS framework; In section 4 we conduct experiment to evaluate our proposal; We conclude this paper with future work in Section 5.

2 Related Work

Query suggestion techniques are used to help users to express their information needs. Search logs have been widely used in recent studies. Craswell[1], Wang[2], Mei[4] and Cao[8] leverage query logs first to construct a Query-URL bipartite graph, then generate suggestions using random walk model based on the click-through relations between queries and URLs. Boldi et.al.[5] utilize query logs to construct a query-flow graph, then apply a short random walk on the graph to generate query suggestions without the click-through information. Song[6] and Ozertem[7] mine a variety of characteristics of queries and use machine learning methods to find out related queries.

Since only considering relevance of query suggestion is far from satisfying users, more and more studies have focused on diversifying query suggestions. Zhu et.al. recommend diverse and relevant queries based on the intrinsic query manifold[15], and they find among the existing researches, Maximal Marginal Relevance (MMR) [11] is the most well-known method used for result set diversification. Ma et.al. [10] combine MMR with random walk to diversify query suggestions. Zhu et.al. [12] apply an absorbing random walk on the graph. In order to achieve diversity, it turns the selected object into an absorbing state and then selects the next object based on the expected number of visits to each node before absorption.

However, the approaches mentioned above cannot guarantee to produce relevant as well as diverse recommendations. These methods fail to combine query semantics and click-through information together to generate highly relevant recommendations. In addition, they all ignore click-through distribution which is an important feature in diversifying process. In this paper, we will show how to combine query semantics and click-through information to guarantee recommendations which are closely related to original query, and how to take advantage of click-through distribution in diversifying process.

3 Methodology

RDQS framework is primarily composed of three steps. The first step is to generate suggestion candidates, and we adopt random walk algorithm to generate n query suggestion candidates in this paper. After that, we re-rank the candidates based on a relevancy criterion using snippets to achieve query suggestions that are more relevant to original query. In the last step, we diversify these suggestions using click-through distribution and generate the final results. The whole process is sketched in Figure 1.

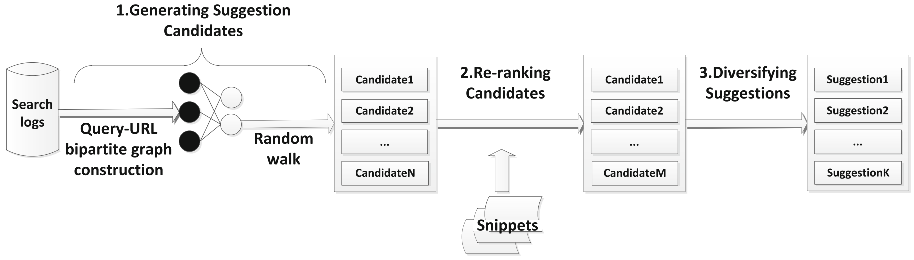


Fig. 1. The framework of RDQS

3.1 Generating Suggestion Candidates

First we use random walk to generate query suggestion candidates. We choose random walk because it's an efficient way to generate candidates as many as possible. Step 1 of RDQS focuses on generating candidates from search logs based on the forward random walk. A Query-URL bipartite graph is constructed as $G = (V, E)$, in which vertexes are composed by two parts $V = V_1 \bigcup V_2$. V_1 represents the set of all unique queries and V_2 represents the set of all unique URLs. There exits an edge from node $x \in V_1$ to node $y \in V_2$ if y is a clicked URL of query q in the search logs. The weight $w(q, u)$ is the total number of times that u is clicked when query q is issued. Note that since the edge is an undirected edge, the weight $w(q, u)$ is the same as $w(u, q)$. The transition probabilities from node i to node j in this paper is defined as follows:

$$P_{t+1|t}(j \mid i) = \frac{\omega(i,j)}{\sum_k \omega(i,k)} \tag{1}$$

where k ranges over all nodes. $P_{t+1|t}(j \mid i)$ denotes the transition probability from node i at step t to node j at time step $t + 1$.

We represent the transitions as a sparse matrix A and perform the random walk using A. We calculate the probability of transition from node i to node j in t steps as $P_{t|0}(j \mid i) = [A^t]_{ij}$. The random walk sums all the probabilities of all paths of length t between the two nodes. It gives a measure of the volume of paths between these two nodes. If there are many paths the transition probability will be high. The larger the transition probability $P_{t|0}(j \mid i)$ is, the more the node j is similar to node i. We select the top n largest $P_{t|0}(j \mid i)$ as candidates. In this paper, we set the $n = 200$ empirically, and 200 is big enough to cover all kinds of related queries.

3.2 Re-ranking Candidates

There are some noises in the generated query candidates. We find that some candidates rank high but not related to the original query, while some highly related candidates have low ranking. In order to guarantee that the results are ranked base on their semantic relevance, we choose to use query semantics to re-rank the candidates.

Query semantics is the content that are closely related to the query in semantic. As we all know, it's hard to determine the degree of semantic relevance between two queries according to their literal meaning because queries are short, generally consisted of less than ten words. However, we can extend the query semantics with related snippets.

Snippets, the few lines of text that appear under each search result, are designed to give users a sense for what's on the page and why it's relevant to their query. We utilize snippets to enrich query semantics, then we can leverage traditional documents similarity comparison methods to calculate queries similarity. In this paper we crawl top ten snippets from Google for each query.

For two queries q_i and q_j, we tokenize their related words into term vector $\overrightarrow{Q_i} = (t_{i1}, t_{i2}, ..., t_{im})$ and $\overrightarrow{Q_j} = (t_{j1}, t_{j2}, ..., t_{jn})$. t_{kl} represents the frequency of term t_l in query q_k, the cosine similarity between two vectors $\overrightarrow{Q_i}$ and $\overrightarrow{Q_j}$ $\cos(\overrightarrow{Q_i}, \overrightarrow{Q_j})$ can measure the similarity of the two queries. The similarity of two queries is calculated as follow:

$$sim(\overrightarrow{Q_i}, \overrightarrow{Q_j}) = \frac{\sum_{k=1}^{n} t_{ik} \times t_{jk}}{\sqrt{\sum_{k=1}^{n} t_{ik}^2} \times \sqrt{\sum_{k=1}^{n} t_{jk}^2}} \tag{2}$$

We compare all the candidates to original query, and re-rank the candidates according to its similarity to original query from high to low. We choose top k instead of all re-ranked results as the input of step 3, because more candidates would bring unrelated suggestions.

3.3 Diversifying Suggestions

Click-through information reflects users' click behaviour in search engines. We define query q_i and URL u_j exist a click-through relationship $R(q_i, u_j)$ when u_j being clicked after q_i being issued. Click-through distribution reflects click-through information over queries. Queries may be redundant in semantic if they have similar click-through distribution. On the contrary, a query will be diverse to an other query if the two queries' click-through distributions are different. For example, for a Query-URL bipartite shown in Figure 2, we can achieve three suggestions – 'nikeid nike', 'nike id' and 'nike shoes' – for original query 'nike'. However 'nike id' seems like a redundant suggestion with 'nikeid nike' since they have similar click-through distribution. In fact, 'nike shoes' is actually a better

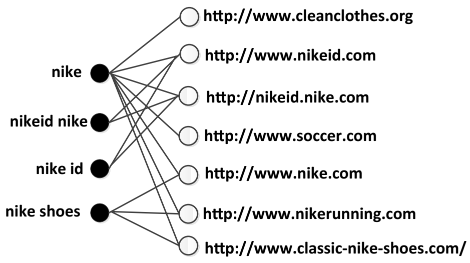


Fig. 2. An example of click-through distribution

suggestion than 'nike id' if we already add 'nikeid nike' in final suggestion results.

Basically similar queries lead to similar click-through distribution, so we can diversify the query suggestion candidates according to the diverse between the click-through distribution. The main idea of the proposed diversified method is that the less click-through distribution the two queries share, the more different they are. Let $\boldsymbol{C}$ be the set of all query candidates, and $\boldsymbol{S}$ be the set of all final query suggestions. At first, $\boldsymbol{S}$ is empty and $\boldsymbol{C}$ is consisted of all the candidates generated from step 2. We first pick the candidates in $\boldsymbol{C}$ that is most similar to original query, that is the one has the highest score in step 2, as the first suggested query. After the selection of the first suggested query, we then employ the click-through distribution to calculate the diversity score of the rest queries in $\boldsymbol{C}$, pick up the candidate which has the highest score as final query suggestion and add it into $\boldsymbol{S}$. We pick out a candidate in $\boldsymbol{C}$ which shares least click-through information with original query and suggestions in $\boldsymbol{S}$. The diversifying algorithm is summarized in Algorithm 1.

Algorithm 1. Diversifying Query Suggestions

```
Input: candidates set $C$ and suggested queries set $S$
Output: A ranked list of all suggested queries $L$, and its size is K
Steps:
1: Initialize candidates set $C$ and suggested queries set $S$: add original query and top
   suggestion candidate $c_0$ generated in re-ranking process into $S$, add all the results
   generated in re-ranking process except $c_0$ into $C$.
2: Repeat the following statements until K queries are suggested in $L$:
   (1) For all queries $q_i$ in $C(q_i \in C)$, calculate diversity score:
           $d(q_i) = \sum_k div(q_i, q_k)$
       $q_k$ is an element in $S$, that is $q_k \in S$
   (2) Pick the next suggested query as $\arg\max_{q_i \in C} d(q_i)$ , and add it into set $S$
```

In this paper, we propose two different ways to calculate diversity between two queries $div(q_i, q_k)$. They are described as follows:

$RDQS_1$: For a query q, we can draw a vector $\overrightarrow{q} = (\omega_1, \omega_2, \omega_3, ..., \omega_n)$ from search logs, the elements in this vector are URLs that clicked through q and w_i in $\overrightarrow{q}$ is the weight of URL_i.

$$\omega_i = \frac{count_i}{\sum\limits_{k=1}^{n} count_k} \tag{3}$$

$count_i$ represents the count of URL_i being clicked through q. Then we can calculate diversity of two queries as follows:

$$\begin{aligned} div(q_i, q_j) &= 1 - \cos(\overrightarrow{q_i}, \overrightarrow{q_j}) \\ &= 1 - \frac{\sum_{k=1}^{n} \omega_{ik} \times \omega_{jk}}{\sqrt{\sum_{k=1}^{n} \omega_{ik}^2} \times \sqrt{\sum_{k=1}^{n} \omega_{jk}^2}} \end{aligned} \tag{4}$$

$RDQS_2$: For a query q, there exists a union $U(q) = (u_1, u_2, u_3, ..., u_n)$, which u_i in $U(q)$ is the URL that clicked through query q. Then we can define the diversity of two queries q_i and q_j as follows:

$$div(q_i, q_j) = 1 - \frac{U(q_i) \cap U(q_j)}{U(q_i) \cup U(q_j)} \tag{5}$$

The time complexity of Algorithm 1 is $O(|C| * k)$, actually the number of queries in C and k are not bigger, and the time complexity of Algorithm 1 is acceptable.

4 Experiment

4.1 Experimental Setup

In this section, we conduct empirical experiments to show the effectiveness of RDQS framework. We select AOL Search Data as our data set, which is a collection of real search log data based on real users. In summary, the data set consists of 20M web queries collected from 650K users over three months. For each query, the following details are available: a user ID, the query itself, timestamp, the clicked URL and the rank of that URL.

Since this dataset is raw data recorded by the search engine, and contains many noises. We conduct a similar method employed in [2] to clean the raw data. we clean the data by keeping those frequent well-formatted English queries (queries which only contain character 'a', 'b', ..., 'z' and space, and appear more than 5 times). After cleaning, we obtain a total of 9,752,848 records, 604,982 unique queries and 785,012 unique URLs.

We construct a query-URL bipartite graph on our data set, and randomly sample a set of 120 queries from our data set as the testing queries. For each testing query, we obtain six query suggestions. In order to evaluate the quality of the results, three experts are requested to rate the query suggestion results with '0' or '1', in which '0' represents 'irrelevant' and '1' means 'relevant'.

In order to evaluate the effectiveness of our framework, we use four methods as baselines, they are *Forward Random Walk(FRW)* [1], *Backward Random Walk(BRW)* [1], *Hierarchy Agglomerative Clustering(HAC)* [9], *and Diversifying Query Suggestions(DQS)* [10]. In order to evaluate the effect of re-ranking process, we add the results which are generated by re-ranking candidates using snippets(RCS) to compare.

4.2 Evaluation Measurements

In this paper, we adopt three metrics(*Relevance*, *Auto-Diversity* and *Human-Diversity*) to evaluate the precision and diversity in query suggestions.

Precision. In order to evaluate the quality of the results, three experts are requested to rate the query suggestion results with "0" and "1", where "0" means "irrelevant" and "1" means "relevant". We use the precision measurement in our experiment, i.e., precision at position n is defined as:

$$p@n = \frac{rn}{n} \tag{6}$$

where rn is the number of relevant queries in the first n results.

Auto-diversity. Auto-diversity metric is a metric using a commercial search engine(i.e., Google), we adopt the same method used in [14]. Specifically, given two queries q_1 and q_2, we compute the proportion of different URLs among their top k ($k = 10$ in this paper) search results, the definition is as follows:

$$d(q_1, q_2) = \begin{cases} 1 - \frac{|o(q_1,q_2)|}{k} & \text{if } q_1 \neq q_2, \\ 0 & \text{otherwise.} \end{cases}$$

where $|o(q_1, q_2|$ is the number of URLs among the top k search results of the query q_1 and q_2. Then for a query q_1, the diversity of its suggestion is defined as:

$$div(q_1) = \sqrt{\frac{\sum_{q_1 \epsilon U} \sum_{q_2 \epsilon U} d(q_1, q_2)}{|U| \left(|U| - 1\right)}} \tag{7}$$

Human-experts-diversity. There exists a problem in auto-diversity evaluation: if a query suggestion has nothing to do with the original query but they share some URLs, it would harm the users' experience, however it still contribute the diversity score in auto-diversity evaluation.

To solve this problem, we propose a new diversity evaluation metric: Human-experts-diversity. Manual evaluation is essential since human experts have a better understanding of the latent semantic meaning than any machines. We ask experts to label the relationship between two queries, there are three kinds of relationships: unrelated, duplicate, distinct. Experts label the query candidates based on the queries semantics and their relations to the original query.

After the accessors labeled all the suggestions, we calculate the diversity of one query suggestion method M as all the number of distinct queries dtn divided by total number of relevant queries rn which consisted of distinct ones dtn and duplicate ones dpn:

$$\begin{aligned} div(M) &= \frac{\sum dtn}{\sum rn} \\ &= \frac{\sum dtn}{\sum dtn + \sum dpn} \end{aligned} \tag{8}$$

4.3 Results and Discussion

Case Study. Some examples of suggestions generated by six methods for original test queries are given in Table 1. From the results shown in Table 1, we can see that some approaches, such as FRW, BRW, HAC, recommend closely related queries while introducing many redundant suggestions. For examples, for the test query 'jet blue', FRW recommends redundant suggestions 'jetblue', 'jetblue airways', 'blue jet', 'jet blue airways', 'jetblue airlines', 'jet blue airline'. BRW recommends redundant suggestions 'jetblue', 'jet blue airlines', 'jetblue

Table 1. Query suggestion comparisons among all methods

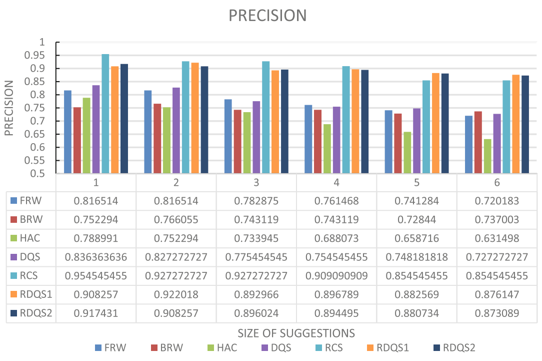
Query	jetblue	playboy	nike
FRW	jetblue	sex	nike shoes
	jetblue airways	play boy	nikeid
	blue jet	playboy magazine	nike golf
	jet blue airways	myspace	nike soccer
	jetblue airlines	playboy centerfolds	nike id
	jet blue airline	my space	niketown
BRW	jetblue	playboy store	nikeid
	jet blue airlines	www playboy com	nike soccer
	jetblue airlines	playboy magazine	nikeid nike
	airlines	nude myspace pictures	nike id
	jetblue airways	playboy plus	nike football
	jet blue airways	playboy centerfolds	nike sb
HAC	jetblue	www playboy com	nike soccer
	jetblue airlines	play boy	nikeid
	jetblue airways	jay hickman	nikeid nike
	jet blue airlines	wackiest ship in army	nike store
	jet blue airways	bing russell	jim rome is burning
	jet blue airline	diana hyland	jim rome
DQS	jetblue	sex	nike shoes
	jet blue airlines	nicole narain	nike tennis apparel
	jetblue airlines	my space	mens shox elevate
	airlines	playboy models	xx jordan
	jetblue airways	playboy pictures	nike fitness apparel
	jet blue airways	kara monaco	adidas
$RDQS_1$	airtran airlines	playboy playmates	nike air max iconic
	delta air	playboy plus	nike yoga
	jetblue	playboy centerfolds	nike fitness apparel
	jetblue airways	playboy store	nikeid nike
	jetblue airlines	playboy of the month	nike football
	cheap airline bargains	playboy pictures	nike shoes
$RDQS_2$	airtran airlines	playboy big boobs	nike air max iconicst
	delta air	playboy store	nike fitness apparel
	airline flights	playboy plus	nike golf
	cheap airline bargains	playboy pictures	nike dunks
	jetblueairlines	playboy lingerie	nikeid nike
	airtran	playboy centerfolds	nike football

airlines', 'jetblue airways' and 'jet blue airways'. HAC recommends redundant suggestions 'jetblue', 'jetblue airlines', 'jetblue airways', 'jet blue airlines', 'jet blue airways', 'jet blue airline'. For the test query 'playboy', HAC suggests redundant suggestions 'www playboy com' and 'play boy'. Since traditional methods only focus on relevancy, they do little work to diversify the results, they produce many redundant recommendations inevitably.

Meanwhile, we can easily find that DQS recommends queries with better diversity. However, there still exist some redundancy in the approach. For example, for the test query 'jet blue', 'jetblue', 'jet blue airlines', 'jetblue airlines', 'jetblue airways' and 'jet blue airways' suggested by DQS are the same meaning.

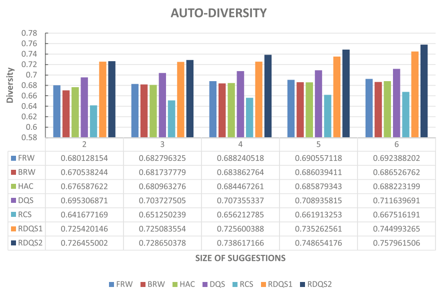
Moreover, results recommended by these baseline methods may not so closely related to the original query. For example, suggestion of 'myspace' with respect to 'playboy', and suggestion of 'jim rome' with respect to 'nike'.

Different from other methods, which only leverage search logs to generate suggestions, our method exploits semantic relationships between original query and suggestions based on snippets. We re-rank the suggestion candidates based on their relevancy using snippets. Therefore, the suggestions are guaranteed to be more semantically related to the original queries. We also use click-through information to diversify the results. Among all these approaches, we observe that $RDQS_1$ and $RDQS_2$ obtain best performance, more relevant as well as diverse queries can be found in their query suggestion results.

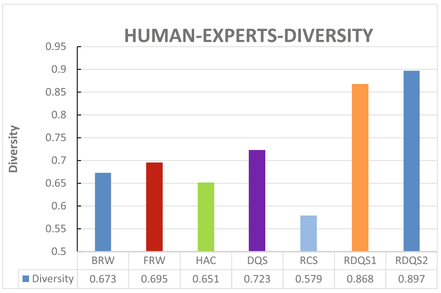


	1	2	3	4	5	6
FRW	0.816514	0.816514	0.782875	0.761468	0.741284	0.720183
BRW	0.752294	0.766055	0.743119	0.743119	0.72844	0.737003
HAC	0.788991	0.752294	0.733945	0.688073	0.658716	0.631498
DQS	0.836363636	0.827272727	0.775454545	0.754545455	0.748181818	0.727272727
RCS	0.954545455	0.927272727	0.927272727	0.909090909	0.854545455	0.854545455
RDQS1	0.908257	0.922018	0.892966	0.896789	0.882569	0.876147
RDQS2	0.917431	0.908257	0.896024	0.894495	0.880734	0.873089

Fig. 3. Precision of query suggestion over Re-ranking, $RDQS_1$ and $RDQS_2$ and other methods



	2	3	4	5	6
FRW	0.680128154	0.682796325	0.688240518	0.690557118	0.692388202
BRW	0.670538244	0.681737779	0.683862764	0.686039411	0.686526762
HAC	0.676587622	0.680963276	0.684467261	0.685879343	0.688223199
DQS	0.695306871	0.703727505	0.707355337	0.708935815	0.711639691
RCS	0.641677169	0.651250239	0.656212785	0.661913253	0.667516191
RDQS1	0.725420146	0.725083554	0.725600388	0.735262561	0.744993265
RDQS2	0.726455002	0.728650378	0.738617166	0.748654176	0.757961506



	BRW	FRW	HAC	DQS	RCS	RDQS1	RDQS2
Diversity	0.673	0.695	0.651	0.723	0.579	0.868	0.897

Fig. 4. Auto-diversity of query suggestion over Re-ranking, $RDQS_1$ and $RDQS_2$ and other methods

Fig. 5. Human-experts-diversity of query suggestion over Re-ranking, $RDQS_1$ and $RDQS_2$ and other methods

Experimental Results. Figure 3 shows the comparison results of precision. Figure 4 shows the comparison results of auto-diversity. Figure 5 shows the comparison results of human-experts-diversity. We have serval findings from the results:

Firstly, in Figure 3 we can see that RCS achieves the best score on precision. In fact, RCS is 13.14% higher than FRW. This is because re-ranking leverages semantics to re-rank the results generated in FRW, thus the re-ranking results are highly relevant to original queries. $RDQS_1$, $RDQS_2$ and RCS take both semantic and click-through relations into consideration, while other baselines only care about click-through relations, so we can observe that $RDQS_1$, $RDQS_2$ and RCS are about the same on precision, much better than other baselines. In addition, RCS is a litter better than $RDQS_1$ and $RDQS_2$ on precision. This is because $RDQS_1$ and $RDQS_2$ diversify the results generated by RCS, and it brings some irrelevant suggestions.

Secondly, both $RDQS_1$ and $RDQS_2$ leverage click-through distribution to diversify suggestion results, and achieve the best diversity performance. In Figure 4 and Figure 5 we can see both $RDQS_1$ and $RDQS_2$ achieve the highest diversity score, both $RDQS_1$ and $RDQS_2$ are at least 10% higher than other methods on diversity. It shows that click-through distribution is an effective feature to diversify query suggestion results. DQS is worse than $RDQS_1$ and $RDQS_2$ but better than other methods in diversity, because FRW, BRW and HAC only focus on finding related queries. $RDQS_1$, $RDQS_2$ and DQS take a step towards diversifying, thus achieve a higher diversity score. $RDQS_1$ and $RDQS_2$ obtain a much better performance than DQS shows that the overlaps of click-through distribution is more effective than hitting information in diversifying process.

Thirdly, RCS achieves the lowest performance in auto-diversity and human-experts-diversity. This is because RCS re-rank results based on the semantics of queries, the suggestions which are similar in semantics are usually representing the same meaning. The results in RCS exist redundancy.

Lastly, as can be seen from Figure 4 and Figure 5, $RDQS_2$ is a bit higher than $RDQS_1$ in diversifying score. That's because $RDQS_1$ cares about not only the overlaps of clicked URLs but also the counts of clicked, it may be overfitting in diversifying process.

5 Conclusion

In this paper, we introduce a uniform framework to generate relevant and diverse query suggestions. Our framework first generates suggestion candidates using forward random walk model which leverage the click-through information, then re-rank the candidates utilizing snippets related to queries. Finally it diversifies candidates based on click-through distribution. In this way, our framework can generate query suggestions by simultaneously considering diversity and relevance between queries in a unified way. The empirical results clearly show that our approach outperforms all the baseline methods in recommending highly diverse as well as closely related query suggestions.

Acknowledgement. This research is supported by the 863 project of China (2013AA013300), National Natural Science Foundation of China (Grant No. 61375054 and 61402045), Tsinghua University Initiative Scientific Research Program Grant No.20131089256, and Cross fund of Graduate School at Shenzhen, Tsinghua University (Grant No. JC20140001).

References

1. Craswell, N., Szummer, M.: Random walks on the click graph. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2007)
2. Wang, X., Zhai, C.: Learn from web search logs to organize search results. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2007)

3. Spink, A., Jansen, B.J.: A study of web search trends. Webology 1(2), 4 (2004)
4. Mei, Q., Zhou, D., Church, K.: Query suggestion using hitting time. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management. ACM (2008)
5. Boldi, P., et al.: Query suggestions using query-flow graphs. In: Proceedings of the 2009 workshop on Web Search Click Data. ACM (2009)
6. Song, Y., Zhou, D., He, L.-W.: Post-ranking query suggestion by diversifying search results. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2011)
7. Ozertem, U., et al.: Learning to suggest: a machine learning framework for ranking query suggestions. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2012)
8. Cao, H., et al.: Context-aware query suggestion by mining click-through and session data. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2008)
9. Beeferman, D., Berger, A.: Agglomerative clustering of a search engine query log. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2000)
10. Ma, H., Lyu, M.R., King, I.: Diversifying Query Suggestion Results. In: AAAI, vol. 10 (2010)
11. Carbonell, J., Goldstein, J.: The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (1998)
12. Radlinski, F., et al.: Redundancy, diversity and interdependent document relevance. ACM SIGIR Forum 43(2) (2009)
13. Baeza-Yates, R., Tiberi, A.: Extracting semantic relations from query logs. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2007)
14. Uhlmann, S., Lugmayr, A.: Personalization algorithms for portable personality. In: Proceedings of the 12th International Conference on Entertainment and Media in the Ubiquitous Era. ACM (2008)
15. Zhu, X., Guo, J., Cheng, X., et al.: A unified framework for recommending diverse and relevant queries. In: Proceedings of the 20th International Conference on World Wide Web, pp. 37–46. ACM (2011)