This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

A New Query Recommendation Method Supporting Exploratory Search Based on Search Goal Shift Graphs

Chao Ma, Bin Zhang

Abstract—Exploratory search is an increasingly important activity for Web searchers. However, the current search system can not provide sufficient support for exploratory search. Therefore, we made in-depth analysis for exploratory search processes, and found that there are a lot of search goal shift phenomena in exploratory search. Based on this fact, we have designed a new query recommendation method to support exploratory search. Firstly, according to the behavioral characteristics of searchers in the search goal shift processes, all the queries submitted in the search goal shift processes are extracted from search engine logs using machine learning. And then we have used the queries to build a search goal shift graph; finally, the random walk algorithm is used to obtain the query recommendations in the search goal shift graph. In addition, we demonstrated the effectiveness of the method for exploratory search by comparing experiments with the other methods.

Index Terms—Query recommendation, Exploratory search, Search goal shift graphs

1 Introduction

Exploratory search is an increasingly important activity yet challenging for Web searchers. In exploratory search, the searcher is unfamiliar with their problem domain, unsures about the ways to achieve their goal, or lacks a well-defined goal [1]. To support exploratory search, the search system is required not only to provide accurate search results, but also to help searchers explore related and novel aspects. Therefore exploratory search system needs an effective query recommendation method to resolve this problem.

However, the current query recommendation methods mainly focus on optimizing users' current query which is far away from satisfying users' information needs of the whole search session. To support exploratory search, we observed and analyzed the search logs of exploratory search process performed by different users, and we found that there are a lot of search goal shift phenomena in exploratory search. As the following example: A Chinese university student attends a birthday party organized by a French student, and he wants to choose a suitable birthday gift, which is a typical exploratory search task. Because the Chinese student only got some very vague goals, such as

Object: a gift not a normal thing
Applicable occasions: birthday party
Basic features: French favorite items
Budget: 200 RMB or so

Based on these conditions, the student used the key words "French people like flowers" for the query; explored "flowers" which is the most popular gift. He felt using flowers as a birthday gift doesn't feature after clicking many links of search results. And the search results mentioned that French people are very fond of drinking wine. So he changed his idea and felt that "wine" may be more appropriate as a gift for the birthday party. So the user used "French wine" as a key word and query "red wine" as new search goal to explore. Using the search results about the "French wine brand" and "French wine prices", the student figured out French wine prices are expensive far beyond his budget. Obviously "red wine" is not a suitable search goal either. At the same time, he thought, "arts and crafts" may be more appropriate. Then he used "handicrafts", "Chinese arts and crafts" as key words to query on the "arts and crafts" which is a new search goal, and eventually found hopeful gift to the end of the search task.

From the example, it's clear that the user's search goal shifts from the "flowers" to "red wine" and then from the "wine" to "arts and crafts". And the search goal shifts precisely reflect the user's exploratory behaviors and needs. Therefore we based on the "search goal shift" designed a new recommendation method to support exploratory search. Firstly, according to the user's behavioral characteristics in the search goal shift process, we extracted all queries during search goal shift processes from search logs; then we used the queries to construct a search goal shift graph; finally, we recommended other goals related to the current goals using the search goal shift graph.

In addition, we have designed a query recommendation test method, by which we can compare our recommendation method with the other methods. And the experimental results showed that the recommendation method we designed can significantly shorten the search

- *Chao Ma Author is with the College of Information Science and Engineering, Northeastern University, Shenyang, 110004 China. E-mail: macmacmac@ yeah.net.*
- *Corresponding author: Bin Zhang is with the College of Information Science and Engineering, Northeastern University, Shenyang, 110004 China. E-mail: zhangbin@mail.neu.edu.cn.*

xxxx-xxxx/0x/$xx.00 © 200x IEEE Published by the IEEE Computer Society

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

time to help users improve the search efficiency.

2 Related Works

2.1 Query Recommendation

Most of the query recommendation techniques are using similarity measures between queries by query terms, clicked documents, or sequences of queries in sessions. Baeza-Yates et al. [2] extracted query-clicked URL/doc bipartite graphs using search logs to find query recommendations. Craswell and Szummer [3] also used the query-click graph to find related documents and queries. Mei et al. [4] presented a "Hitting Time" algorithm to find related queries using the query-click graph. Cao et al. [5] tried to understand user's context which includes multiple information including age, gender, username, IP, tools etc. and also previous queries in a query session in order to suggest new queries. Boldi et al. [6] proposed a query-flow graph which represents the latent querying behavior contained in a query log.

2.2 Exploratory Search

In the past 30 years, many scholars have made in-depth study of the search process of exploratory search behavior. In 1989, Dr. Bates M J proposed Berrypicking model [7] that the user's search direction and the desired result will constantly change with the search process changing. In 1991, Kuhlthau C C proposed that information retrieval process includes starting, selection, exploration, collection and ending six stages [8]. In 1995, Byström K and Järvelin K used the methods of logs and questionnaires to analyze the relationship with search complexity of the task, type of information, information channels and resources [9]. In 2006, Marchionini G proposed exploratory search [10].

Recently, exploratory search research focuses on the characteristics of the exploratory search process and the different types of support needed to help people make exploratory searches [1]. Someone tries to provide a query preview control by allowing users to take nodes and record the results [11] so that they can view the distribution of newly-retrieved and retrieved documents before running the query [12]. Some research efforts focus on traditional search techniques such as query suggestions, aspects and information classification. For example, Hassan Awadallah et al. [13] constructed a method of automatically identifying and recommending tasks that allow searchers to explore and complete complex search tasks, Sun et al. [14] proposed a topic-oriented query for exploratory search method, Ksikes et al. [15] designed an exploratory faceted search system, Zhang et al. [16] grouped the relationships between entities into a virtual-generated hierarchical clustering to an effective leader to explore and discover. Other attempts have been made to design and research visual search interfaces and interactive user modeling to support exploratory search tasks. For example, Bron et al. [17] proposed an auxiliary exploratory search interface to support media research; Bespinyowong et al [18] designed exploratory data exploratory ranking interface; Peltonen et al [19] used a negative feedback search intent radar interface to help users conduct exploratory search.

All these previous methods focus on refining user requirements, showing several facets helping users refine their requirement and find their desired information. But they cannot satisfy such user needs as finding some novel search goals when users are losing the interests of current search goal.

In contrast to these methods our method focuses on recommending some new search goals that help users explore related and novel aspects when they are losing the interests of current search goal.

3 Motivations

In order to make a more accurate analysis for the behavioral characteristics of exploratory search, we firstly designed four search tasks with the characteristics of the exploratory search according to the literature [20], and organized 30 volunteers to make actual searches for these exploratory search tasks. Specific tasks as follows:

1. Supposing you are a reporter and asked to write an article about information security, the article needs to include a basic introduction theme, as well as significant information security incidents which happened in the past. And describe how the security of information affects people's daily lives from different views. Finally focus on the ways of preventing information leaks and improving information security.
2. Assuming you have a family of four and a monthly household income of 20,000 RMB, and you are ready to buy a house in Beijing. Now you need to write a purchase plan to help families make decisions. The plan includes each property's basic information outlined in items such as: location, size and price. Then it should also describe periphery of education, health care and property management information. Finally it should compare advantages and disadvantages of each project analyzed.
3. Supposing you have a 40-year-old, type 2 diabetic male relative who wants to lose weight. But he can only spend three hours a day exercising due to his busy work. He is not familiar with network technology, so he wants you to help him develop a diet and exercise plan. This plan should not only include the effects, but also mention that it may bring risks and the way of how to monitor the risks.
4. The examples mentioned in the preamble, let's say you are a Chinese university student and preparing for a birthday party organized by the French students. How to choose a suitable birthday gift? The task requires you to select the strategy and selection reason or exclusion reason for each item.

For each search task we required volunteers to complete in 1 to 2 hours. And in each task we obtained the results at the same time also collected each volunteer's search logs to analyze. We adopted the following concepts and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

definitions for analysis processes:

Related concepts:

1. **Query topic:** The query topic is a category used to describe the structure of the query information.
2. **Hierarchy of the query topics:** We divided the hierarchy of query topics for all the users by the two-tier topic model.

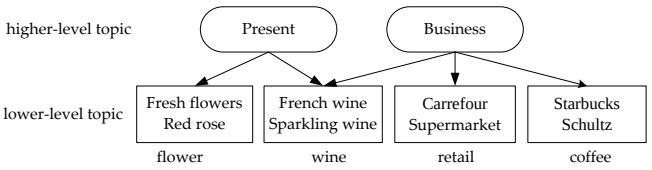

Fig. 1. Hierarchy of query topic

As Figure 1 shown, we define all the topics contained in the upper layer as higher-level topics, and define all the topics we include in the second layer as lower-level topics in this model. The lower-level topic represents common co-occurence patterns (correlations) between words in sentences. The higher-level represents common co-occurence patterns (correlations) between lower-level topics in documents. For example "starbucks" and "schultz (Starbucks creators)" often appear in the same sentence, so they are often clustered into the same lower-level topic (coffee). "Carrefour" and "supermarket" are also clustered into the same lower-level topic (retail) because they often appear in the same sentence. The statements that contain "starbucks" and "schultz" and statements that contain "Carrefour" and "supermarket" often appear in the business documents, so lower-level topics "coffee" and "retail" are often clustered into the higher-level topic "business".

3. **The relationship between the query topics:** According to the two-tier topic model, we can get the relationship between the query topics shown in Table 1.

Table 1
The relationship between query topics

$Sim_{L-topic}(q,q')$	$Sim_{H-topic}(q,q')$	**relationship**
>threshold η	>threshold ϕ	$T_q = T_{q'}$ topic consistency
>threshold η	<threshold ϕ	$T_q \leftrightarrow T_{q'}$ topic coherence
<threshold η	<threshold ϕ	$T_q \neq T_{q'}$ topic change

Related definitions:

1. **Search goal:** An atomic information need is reflected through a query query or more queries. That is, two consecutive queries in the same session, and if they are similar to each other based on a given threshold, they belong to the same search goal. The form is described as:

$$Same_{goal}(q,q') = \begin{cases} 1 & Sim(q,q') > threshold\ \theta \\ 0 & Sim(q,q') < threshold\ \theta \end{cases}$$

2. **Search goal shift:** In the exploratory search process, when the user isn't interested in the current search goal, and submits a query q′ for the next step search. If the query q and the query q′ do not belong to the same search goal but they are topically-coherent, the search process between the two quiries is defined as the search goal shift. The query q and the query q′ are called the search goal shift query pair. Its formal description is $q \rightarrow q'$.

Fig. 2 shows a typical search goal shift process. The search goal shifts from the "flower" to "wine", which can be represented by a search goal shift query pair "Red rose" $\rightarrow$"French wine".

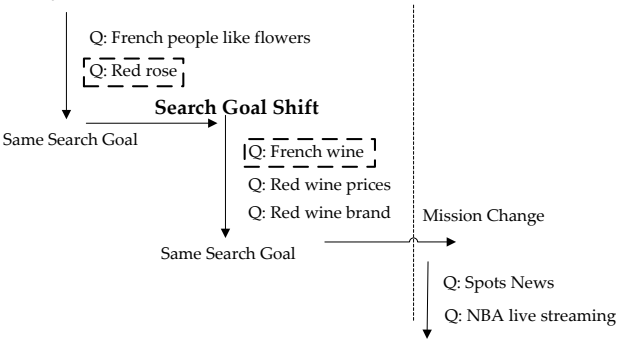

Fig. 2. "Search goal shift" process

After the experiments we got the results of the analysis in Fig. 3 and Fig. 4. From Fig. 3, we can see all queries submitted during the search goal shift have accounted for more than 50%, and at the same time from Fig. 4 we can see that the search goal shift is almost throughout the search process.

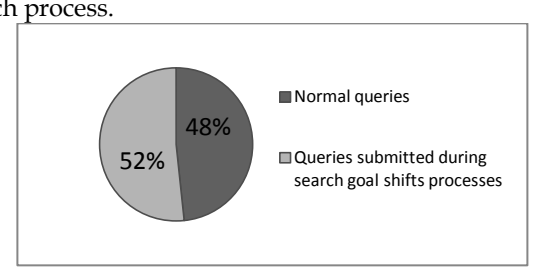

Fig. 3. "search goal shift" distribution

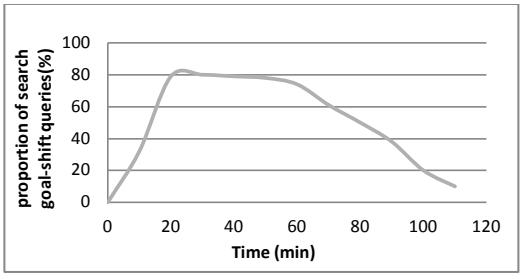

Fig. 4. "search goal shift" scale changes by time

The above results show that the search goal shift phenomenon is one of the important behavioral characteristics of exploratory search process. So it has very high research value of the query recommendation method based on the search goal shift.

4 Query Recommendation Framework

Based on the basic framework of the search goal shift graph, exploratory query recommendation method mainly consists of two parts, offline and online, as shown in Fig. 5.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

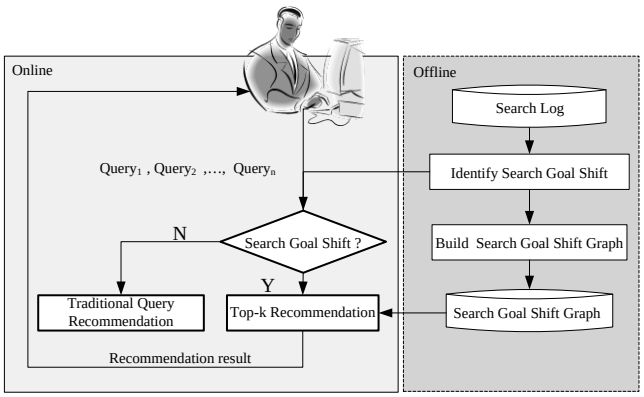

Fig. 5. Query recommendation framework

4.1 Offline Part

Offline part mainly includes two major steps, the search goal shift identification and the search goal shift graph building. In the offline part, we manually annotate the search goal shift in some users' search session, then use machine learning to convert inefficient manual identification process into efficient AI calculation. Finally we use all queries submitted during search goal shifts to construct a search goal shift graph.

4.2 Online Part

Online part also contains two steps, user's search behavior judgment and top-k recommend. In the online part, we use the identification model which is trained from the offline part to judge whether users' search behaviors belong to "search goal shift", then we use a random walk algorithm to find the top-k most relevant search queries from the search goal shift graph as a result of recommendation.

5 Identifying Search Goal Shift

Because our final goal is to provide query recommendations for users, the process of "identifying search goal shift" is to identify all search goal shift query pairs from the search engine logs and use them to compose of the candidate set. To this end, we took Sogou company's search engine logs in June 2008 as experimental data, used the approach designed in [21] to extract the massive exploratory search sessions from the logs, and randomly selected 5000 of them as the sample set. And then we instructed external human judges to examine each query pair in these sessions. If there is a search goal shift between two queries in a query pair, the query pair is marked as a search goal shift query pair, otherwise marked as a normal query pair. Finally we used various feature values extracted from the annotation results to train an identification model. The specific approach is as follows:

5.1 Features of the Search Goal Shift

In this section, we described the extraction process of the features used for identification.

5.1.1 Query similarity

According to the previous definitions, if the search goal shift occurs between two queries q, q', the two queries belong to different search goals. Thus we might expect the similarity between two queries in a query pair can be used to identfiy whether the query pair is a search goal shift query pair or a normal query pair. In order to examine this, we need to measure the similariy between q and q'. In our work, we proposed two metrics and two measures for assessing the relatedness of two queries both in terms of their text content and their semantic.

1) **Content Similarity**. Two queries that share some common terms are likely related. Such as the following two queries:

q: French wine brand

q': France wine prices

To measure the content similarity between queries in a query pair, we began by performing standard text normalization where we removed the stop word, unified the case of the text, and replaced all runs of whitespace characters with a single space. Thus every query is represented as a bag of non-stop word terms, and then we adopted Jaccard index [22] based on terms to calculate content similarity between two queries.

The Jaccard index, also known as Intersection over Union and the Jaccard similarity coefficient, is a statistic used for comparing the similarity and diversity of sample sets [22]. The Jaccard similarity coefficient between two queries q, q' is computed as follows:

$$Jaccard(q,q') = \frac{\left|T(q) \cap T(q')\right|}{\left|T(q)\right| + \left|T(q')\right| - \left|T(q) \cap T(q')\right|} \quad (1)$$

where $\left|T(q)\right|$ is the number of terms in query q, and $\left|T(q) \cap T(q')\right|$ is the number of matched terms in q and q'.

In setting the term match rule, we found that sometime the two terms may be very similar, but not identical, due to mispelling, or different prefixes/suffixes, as in the example above, "french" and "france". So we use the Levenstein distance [23] to measure the degree of match between the two terms.

Levenshtein distance is a string metric for measuring the difference between two sequences. Informally, Levenshtein distance between two terms is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one term into the other [23]. The Levenshtein distance between two terms t, t' given by $Lev_{t,t'}(|t|,|t'|)$ where

$$Lev_{t,t'}(i,j) = \begin{cases} Max(i,j) & if\ \min(i,j) = 0 \\ \min\begin{cases} Lev_{t,t'}(i-1,j)+1 \\ Lev_{t,t'}(i,j-1) \\ Lev_{t,t'}(i-1,j-1)+1_{(t_i \neq t'_j)} \end{cases} & otherwise \end{cases} \quad (2)$$

For example, the Levenshtein distance between "french" and "france" is 2, since the following two edits change one into the other, and there is no way to do it with fewer than two edits:

1. fr**e**nch → franch (substitution of "a" for "e")
2. franch → france (substitution of "e" for "h")

The rule of terms matched is as follows:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

$$match(t,t') = \begin{cases} 0 & if\ \ lev(t,t') > 2 \\ 1 & otherwise \end{cases} \quad (3)$$

According to the rule, content similiarity between "French wine brand" and "France wine prices" is 0.5 as follows:

$$Sim_{context}(q,q') = \frac{2}{3+3-2} = 0.5$$

2) Semantic Similarity: In the calculation of query similarity, we found that two queries may also belong to the same search goal due to similar semantic relatedness even if the text content of them is completely different. For example, the following two queries:

q: flesh flower

q′: red rose

The query q and query q′ share the semantics of the "flower", "plant" and so on. So we need to measure the semantic relatedness between the two queries.

For the semantic relatedness between two terms, humans are very easy to judge by their experience and knowledge. But for the computer it is a very difficult thing. In order to realize the computer intelligence calculation of semantic similarity between terms, we must let the computer have relevant semantic knowledge. Usually, this semantic knowledge comes from large corpus (e.g., WordNet, Wikipedia, etc.).

At present, there are two main ways to use semantic knowledge to measure the semantic relatedness between terms. One is based on the path of the calculation method. The method builds a knowledge graph with all the semantic knowledge, and then calculates the length of the path between the different nodes to measure the semantic similarity between the terms [24]. The other is based on the word vector calculation method. This method is to represent each word in a collection as a point in a space (a vector in a vector space). Points that are close together in this space are semantically similar and points that are far apart are semantically distant [25]. Following the latter approach, we used a recently popular and fast tool called word2vec [26] to generate word embeddings (word vectors in a vector space with much lower dimension) from a big corpus.

Word2vec is a group of related models that are used to produce word embeddings. It takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space [26].

We trained word2vec on the Wikipedia documents, with 1897 million words, using the default parameters (CBOW model with window size as five). For each word, we finally obtained a 200 dimensional vector, as follows:

$$\vec{V}(t_i) = (w_1, w_2, ...w_{n=200})$$

For example, the previously mentioned "flower", "rose" can be expressed by the following word embedding:

$$\vec{V}(flower) = [0.792, -0.177, -0.107, 0.109, -0.542, \ldots]$$

$$\vec{V}(rose) = [0.817, 0.213, -0.289, 0.353, -0.912, \ldots]$$

The semantic similarity between two terms is calculated by the cosine of the angle between the vectors as follows.

$$Sim_{semantic}(t_i, t_j) = \frac{\vec{V}(t_i) \times \vec{V}(t_j)}{\left|\vec{V}(t_i)\right| \times \left|\vec{V}(t_j)\right|} \quad (4)$$

Finally, we used the following steps to compute the semantic similarity between queries based on the term-term semantic similarity:

Step 1: we constructed a semantic vector space with all the terms without duplication in queries q and q′. As in the example above, q: "flesh flower" and q′: "red rose", there are 4 terms without duplication in q and q′, the semantic vector space is constructed as follows:

$$T = \{"flesh", "flower", "red", "rose"\}$$

Step 2: To generate a vector for each query, such as $\vec{V}(q) = (w_1, w_2, ..., w_{n=4})$, we compared each term in T with that in the query. If the term t_i belongs to the query q, the weight value w_i is 1.

For example: q is "flesh flower" and t_2 is "flower", then w_2 is 1.

If the term t_i does not belong to a query q, we calculated the semantic similarity between the term t_i and each term in the query q with formula (4). And then we computed the weight value w_i as follows:

$$w_i = \begin{cases} Sim_{\max} & if\ \ Sim_{\max} > threshold \\ 0 & otherwise \end{cases} \quad (5)$$

Finally, we obtained the semantic-based vetor for each query, such as

$$\vec{V}(q) = ["flesh", "flower", "red", "rose"] = [1, 1, 0, 0.59]$$

$$\vec{V}(q') = ["flesh", "flower", "red", "rose"] = [0, 0.59, 1, 1]$$

Step 3: we computed the semantic similarity between queries as follows:

$$Sim_{semantic}(q, q') = \frac{\vec{V}(q) \times \vec{V}(q')}{\left|\vec{V}(q)\right| \times \left|\vec{V}(q')\right|} \quad (6)$$

3) Feature validity verification: In order to verify that query similarity can be used to distinguish between the search goal shift query pair and the normal query pair, we use the above two metrics to calculate the similarity between queries in each query pair in our dataset.

$$Sim(q, q') = \alpha \times Sim_{context}(q, q') + (1 - \alpha) \times Sim_{semantic}(q, q') \quad (7)$$

And then based on the query on the different positions in the session, we compared search goal shift query pairs with normal query pairs on the average query similarity. The comparison results are shown in Fig. 6.

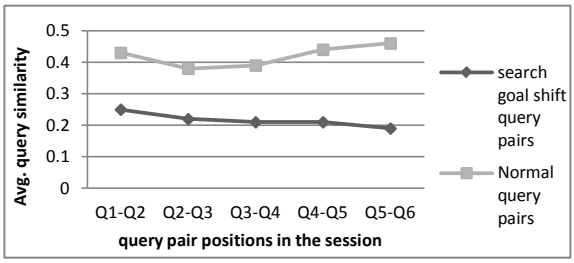

Fig. 6. Comparison result of query similarity

Fig. 6 shows that the average query similarity in the search goal shift query pairs is lower than the average query similarity for the normal query pair. Since the scale of the sample data is much smaller than the scale of the real dataset, we

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

performed a two-tailed T-test in order to make sure the same result in the real dataset. The test results show that all differences reported in Fig. 6 are statically significant at the 0.05 level. This confirms that the query similarity can be used to distinguish between the search goal shift query pair and the normal query pair.

5.1.2 Topic similiarity

According to the previous definitions, if the search goal shift occurs between two queries q, q', the topical relationship between q and q' is just topically-coherent rather than topically-consistent. Thus we guessed that the topic similarity between queries in the search goal shift query pair is lower than the topical similarity between queries in the normal query pair. To verify this, we need to measure the topic similarity between q and q'.

To compute the topical similarity between two queries, we began by calculating the lower-level topic similarity and the higher-level topic similarity.

1) Lower-level topic similarity: According to the definition of lower-level topic, the lower-level topic represents common co-occurence patterns (correlations) between words in sentences. Thus we have access to all documents clicked by the user during the search session in our dataset. And then we computed the lower-level topic similatiry between two queries based on the term-term co-occurrence information in the sentences. To obtain the term-term co-occurrence information, we used the Normalized Pointwise Mutual Information. Pointwise mutual information (PMI) is a measure of how much the actual probability of a particular co-occurrence of events p(x, y) differs from what we would expect it to be on the basis of the probabilities of the individual events and the assumption of independence p(x)p(y) [27]. PMI for two terms is computed as follows:

$$PMI(t_i,t_j)=\ln\frac{P(t_i,t_j)}{P(t_i)\times P(t_j)} \tag{8}$$

where $P(t)=s_t/S$, $P(t_i,t_j)=s_{t_i,t_j}/S$, S is the number of all sentences in the dataset, s_t is the number of sentences containing the word t in the dataset, s_{t_i,t_j} is the number of sentences containing both the word t_i and word t_j in the dataset.

In addition, since PMI can take arbitrary positive or negative values, we normalized it into NPMI as follows:

$$NPMI(t_i,t_j)=\frac{\ln\frac{P(t_i,t_j)}{P(t_i)\times P(t_j)}}{-\ln p(t_i,t_j)} \tag{9}$$

Finally, the lower-level topic similarity between queries is computed as follows:

$$Sim_{L-topic}(q,q')=\frac{\sum_{t_i\in q}\sum_{t_j\in q'}\ln\frac{P(t_i,t_j)}{P(t_i)\times P(t_j)}}{-\sum_{t_i\in q}\sum_{t_j\in q'}\ln P(t_i,t_j)} \tag{10}$$

2) Higher-level topic similarity: According to the definition of the higher-level topic, the higher-level represents common co-occurence patterns (correlations) between lower-level topics in documents. So the higher-level topic similarity calculation method is same as the lower-level topic similarity calculation method except for the computing for $P(t)$ and $P(t_i,t_j)$.

In the higher-level topic similarity calculation method, $P(t)$ and $P(t_i,t_j)$ are computed as follows:

$P(t)=d_t/D$, $P(t_i,t_j)=d_{t_i,t_j}/D$, D is the number of all documents in the dataset, d_t is the number of documents containing word t in the dataset, d_{t_i,t_j} is the number of documents in which the word t_i and the word t_j are in the same sentence.

3) Feature Valid confirmation: In order to verify that the query topic similarity can be used to distinguish between the search goal shift query pair and the normal query pair, we use the above two metrics to calculate the topic similarity between queries in each query pair in our dataset.

$$Sim(q,q')=\beta\times Sim_{L-topic}(q,q')+(1-\beta)\times Sim_{H-topic}(q,q') \tag{11}$$

And then based on the query on the different positions in the session, we compared search goal shift query pairs with normal query pairs on the average topic similarity. The comparison results are shown in Fig. 7.

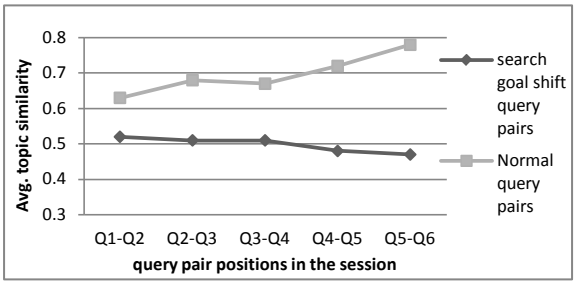

Fig. 7. Comparison result of topic similarity

Fig. 7 shows that the average topic similarity in the search goal shift query pairs is lower than the average query similarity for the normal query pair. All differences reported in Fig. 7 are statically significant at the 0.05 level according to a two-tailed t-test. This confirms that the query topic similarity can be used to distinguish between the search goal shift query pair and the normal query pair.

5.1.3 Click ranking

According to the definition of the search goal shift, the user lost the interests of the current search goal, which is the root cause of the search goal shift. And the rank of clicked URLS is an important indicator of whether users are satisfied with the current search goal [28]. Therefore we suspected that users tended to click on lowly ranked results during the search goal shift processes. To verify this, we extracted all clicks performed by the user during the search session in our dataset and filtered out all invalid clicks (click through less than 10 seconds), as well as clicks that lead to another search result page (e.g., related search). And then we computed the rank of clicked URLS for the first query of each query pair in our dataset as follows:

$$Rank_{url}(q,q')=\frac{\sum_{i=1}^{n}\frac{1}{Rw_i\times\log_2(1+pos_i)}}{N_{click}} \tag{12}$$

where pos_i is the posion of the i-th clicked URL in the result. N_{click} is the number of clicks. Rw_i is a weight coeffi-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

cient. Since browsing time reflecs whether the user is satisfied with the clicked document [29], we used browsing time to compute Rw_i as follows:

$$Rw_i = \frac{brs\_t_i}{brs\_t_{agv}} \tag{13}$$

Where brs_t_i is the time spent by the user browsing the ith clicked document and brs_t_{agv} is average browsing time in our dateset. The browsing time can be estimated from click logs by computing the time between the click and the next seen click or query on the search engine.

Finally, based on the query on the different positions in the session, we compared search goal shift query pairs with normal query pairs on the average rank of clicked URLS. The comparison results are shown in Fig. 8.

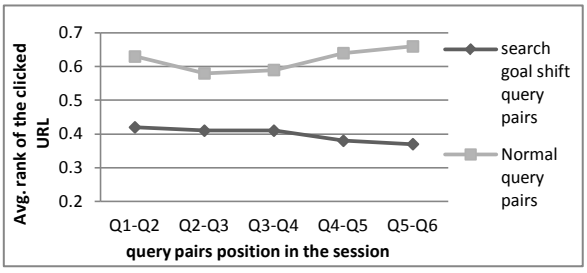

Fig. 8. Comparison result of rank of the clicks

Fig. 8 shows the average rank of clicked URLS for search goal shift query pairs is less than the average rank of clicked URLS for normal query pairs. All differences reported in Fig. 8 are statically significant at the 0.05 level according to a two-tailed t-test. This confirms that the rank of clicked URLS can be used to distinguish between the search goal shift query pair and the normal query pair.

5.2 Identification Algorithm

In this section, we used the logistic regression algorithm to transform the identification of the search goal shift into the binary classification problem.

The core idea of logistic regression algorithm is using sigmoid function to convert classification problem into a probability estimate. If a feature value is inputted into the sigmoid function and the calculated value is greater than a certain threshold, the query pair is search goal shift query pair. Otherwise less than or equal to the threshold, the query pair is a normal query pair.

T (true) indicates that the goal shift query pair, F (false) indicates the normal query pair. The model target is to find the function f,

$$f : D \to \{T, F\} \tag{14}$$

where $D=\{x_1, w_1, x_i, w_i, \ldots, x_n, w_n\}$, x is the input feature values vector. w is the coefficients vector. The probability of search goal shift query pair is:

$$P(y = T \mid x; w) = \frac{1}{1 + e^{-w^T x}} \tag{15}$$

$$P(y = F \mid x, w) = 1 - P(y = T \mid x; w) = 1 - \frac{1}{1 + e^{-w^T x}} \tag{16}$$

The function f is:

$$f(P(D)) = \begin{cases} T, P(D) > threshold \\ F, P(D) \le threshold \end{cases} \tag{17}$$

According to the derivation above, the process of getting model (function f) is a finding process for best-fit parameters, which is the coefficient of each feature value. For parameter vector w, we use maximum likelihood estimation algorithm to obtain.

$$\text{Let } h_w(x) = g(w^t x) = \frac{1}{1 + e^{-w^T x}};\ g(z) = \frac{1}{1 + e^{-z}} \text{ then;}$$

$$P(D) = P(y \mid x; w) = (h_w(x))^y (1 - h_w(x))^{1-y} \tag{18}$$

According to (18), we can get the maximum likelihood function L (w | x, y) and the optimization function l (w).

$$\begin{aligned} l(w) &= \log(L(w \mid x, y)) \\ &= \sum_{i=i}^{m} (y^i \log h_w(x^i) - (1 - y^i) \log(1 - h_w(x^i)) \end{aligned} \tag{19}$$

Finally, we can use gradient descent optimization method to obtain update way of w:

$$w := w + \alpha \cdot x^T (g(w \cdot x) - y) \tag{20}$$

Specific recognition algorithm as follows:

Algorithm 1 ExIdentifyGSPair
Input: rate, x, y
// rate is learning rate
// X is feature value vector
// Y the artificial identification results
Output: w
// parameter vector
1. w = initialize_value;
2. For every $x_i \in X$, $y_i \in Y$ Do
3. $p = \exp(w \cdot x_i) \big/ (1 + \exp(w \cdot x_i))$
4. If p > threshold Then
5. Predict true;
6. Else
7. Predict false;
8. End If
9. If $y_i == 1$ Then
// It indicates that the current query pair is
// the goal shift query pair
10. $w = w + (1 - p) \cdot x_i \cdot rate$
11. Else
12. $w = w - p \cdot x_i \cdot rate$
13. End If
14. End For

6 The Search Goal Shift Graph

The search goal shift graph is a directed graph Ggs = (V, E, w) where:

1. The set of nodes is V=Q, Q is a distinct set of search goals;
2. $E \subseteq V \times V$ is the set of directed edges;
3. $w : E \to (0,1]$ is a weighting function that assigns to every pair of search goals $\{g, g'\} \in E$ a weight $W(g, g')$.

6.1 Building the Search Goal Shift Graph

1) Adding node: As shown in Fig. 9, we used the search

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

goal shift identification method designed in section 5 to extract all search goal shift query pairs from search engine logs. And then we used the query similarity function designed in section 5.1.1 to group all queries pertaining to the same search goal together, such as query "Fresh follower" and query "Red rose". They were assigned to the same group due to belonging to the same search goal ("Flower" node).

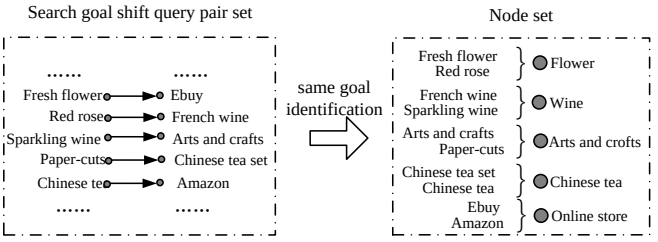

Fig. 9. Construction of node set

2) Adding edge: Given two search goals (query groups) g, g′, we tentatively connect them with an edge if there is at lest one search goal shift query pair between g and g′. As shown in Fig. 10, there is an edge between the "flower" node and the "wine" node because there is a search goal shift query pair {"Red rose" →"French wine"} between them.

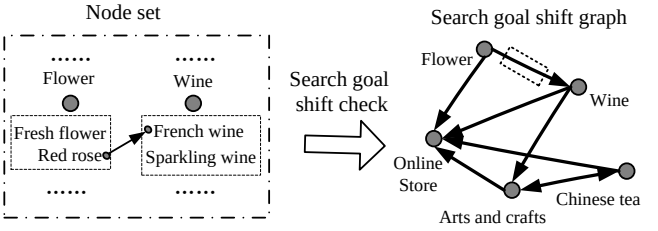

Fig. 10. Construction of edge set

3) Computing edge wight: For two goals (nodes) N_i, N_j linked by an edge, the edge weight w(N_i, N_j) represents the strength of their association.

To measure the association between pairs of search goals we used the Normalized Pointwise Mutual Information described in section 5.1.2.

$$Weight(N_i, N_j) = \frac{\sum_{q_i \in N_i} \sum_{q_j \in N_j} \ln \frac{p(q_i, q_j)}{P(q_i) \times P(q_j)}}{-\sum_{q_i \in N_i} \sum_{q_j \in N_j} \ln p(q_i, q_j)} \quad (21)$$

where $P(q_i) = C_{q_i} / Q$, $P(t_i, t_j) = C_{q_i, q_j} / Q$, Q is the number of all queries in the dataset, C_{q_i} is the number of query qi in the dataset, C_{q_i, q_j} is the number of search goal shift query pairs $\{q_i \rightarrow q_j\}$ in the dataset. Specific recognition algorithm as follows:

Algorithm 2 ExBuildSGSGraph
Input: P
// P is the goal shift query pair set
Output: V，E
// V is the set of nodes E is the set of edges

```
1.      A = getAllquerise(p);
// put all queries submitted during search goal shifts into a
// query set A
2.      For q ∈ A Do
3.          For q′ ∈ A Do
4.              v = new Node(q)
5.              V.add(v)          // add a new node
6.              If SameGoal(q, q′) Then
7.                  query_listv.add(q′)
8.              End If
9.          End For
10.     End For
11.     For v ∈ V Do
12.         For v′ ∈ V Do
13.             query_list_v = getQueryList(v)
14.             query_list_v′ = getQueryList(v′)
15.     If ExitSGS (P, query_list_v, query_list_v′) Then
16.             e_v,v′ = 1          // get a new edge
17.             w(v,v′)=NMI(v,v′)  // set weight
18.         End If
19.     End For
20.     End For
```

6.2 Query Recommendation Based on Search Goal Shift Graphs

Given the nature of the task graph, we operated in the space of search queries and recommended queries for a given query. In this process, we adopted the personalized Pagerank random walk algorithm [30] to identify candidate recommendations.

PageRank is a link analysis algorithm and it assigns a numerical weighting to each element of a hyperlinked set of documents. Given a graph $G = (V, E)$ where V is a set of nodes $V = \{v_1, v_2, ..., v_n\}$ and E is a set of edges. A surfer starts from a random node v_i of V and at each step he/she follows a hyperlink with probability $c \in (0,1)$ or gets bored and jumps to a random node with probability $1-c$. PageRank vector $\vec{p}$ is computed as follows:

$$\vec{p} = (1-c) \times \vec{p}A + c \times \vec{r} \quad (22)$$

where A is an adjacency matrix of the graph G with normalized rows. If the random jump probability $\vec{r}$ is uniform, such as $\vec{r} = \left[\frac{1}{N}\right] N \times 1$, the algorithm is named as the global Pagerank algorithm.

In contrast to the Global Pagerank algorithm, the existing Personalized Pagerank algorithms use the personalized jump probability vector $\vec{r}$ to enter user preferences such as increasing the probability of jumping to the nodes most closely associated with the current query. For our method, we achieve full personalization by increasing the probability of jumping to the nodes most closely associated with all queries submitted in the user's whole search process.This approach may help to alleviate the data sparsity problem --the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

current query may be rare, but among the previous queries there might be queries for which we have enough information in the search goal shift graph.

According to this idea, the random jump probability vector $\vec{r}$ is computed as follows:

$$\vec{r}_{ij} = \begin{cases} \frac{1}{|U_j|} & if\ i \in U_j \\ 0 & otherwise \end{cases} \quad (23)$$

If all queries submitted by the user belong to the node set S, U_j is the set of nodes connected by the directed edges starting from S. The recommendation algorithm as follows:

Algorithm 3 ExRankQuery

Input: A, c, $\vec{r}$

Output: $\vec{p}$

1. $\vec{p}$ = initialize_value
2. $\vec{x} = (1-c) \times \vec{p}A + c \times \vec{r}$
3. For 1==1 Do
4. If $|\vec{p} - \vec{x}| < \varepsilon$ Then // ε is infinitesimal
5. Return $\vec{x}$
6. Else
7. $\vec{p} = \vec{x}$
8. $\vec{x} = (1-c) \times \vec{p}A + c \times \vec{r}$
9. End If
10. End For

After calculating the rank value of all the nodes, we can rank them by descending order, and take the top k nodes as the recommendation result. If there are multiple queries in the recommendation node, we need to recommend the most frequently occurred query in the query group.

7 EXPERIMENT

7.1 Experimental Data

We choose search engine query logs of Sogou Company in June 2008 as base experimental data. The log data format is: access time \t user ID \t [search words] \t the URL's ranking in returned results \t sequence number of use clicked\t URL user clicked. In the experimental data pre-processing stage, we used the approach designed in [21] to extract 5000 exploratory search sessions from the search engine log, totally 50,616 queries.

7.2 Evaluation of the Identification Algorithm

Ground Truth for test: We used the method of artificial identification to select 10000 query pairs from the 5000 search sessions. The choice criterion is to extract two query pairs from each session, a search goal shift query pair and a normal query pair. And then we randomly selected a certain number of query pairs, and used them to compose of a dataset for test. According to the observed result in section 3 (Fig. 3), the proportion of search goal shift query pairs in the dataset is 52%, and these search goal shift query pairs were considered to be ground truth. For this proportion, the maximal number of query pairs in the dataset for test is limited to 9000 pairs.

Baseline method: We compared logistic regression algorithm adopted by us (LR) with the following two the-state-of-art identification algorithms

1) Support Vector Machine (polynomial kenel) [31]
2) Decision tree (C5.0) [32].

Evaluation Metrics:

1) **Precision:** It indicates the proportion of the actual positive examples in positive examples set.

$$precision = \frac{TP}{TP + FP} \times 100\% \quad (24)$$

2) **Error rate:** It indicates the proportion of misclassified.

$$error = \frac{FP + FN}{P + N} \times 100\% \quad (25)$$

3) **Recall:** It is a measure of coverage, metrics of how many positive cases in positive examples.

$$recall = \frac{TP}{TP + FN} \times 100\% \quad (26)$$

Where TP represents the number of correctly classified as positive examples. FP represents the number of incorrectly classified as positive examples. TN indicates the number is correctly classified as a negative example. FN represents the number of incorrectly classified as negative example.

Experimental results and analysis:

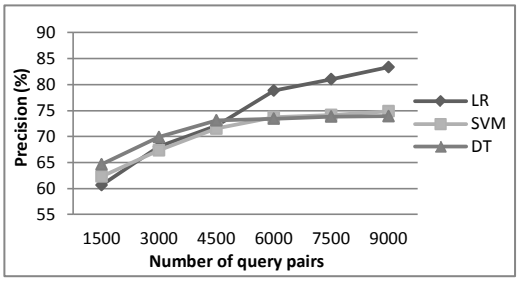

Fig. 11. Comparison result of precision. We split each set to 4/5 training and 1/5 testing

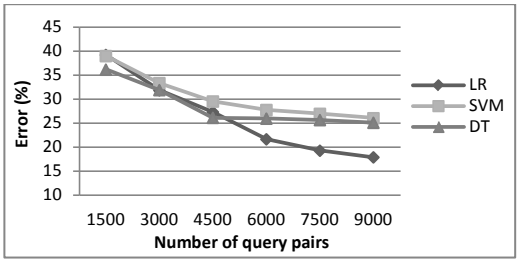

Fig. 12. Comparison result of error rate. We split each set to 4/5 training and 1/5 testing

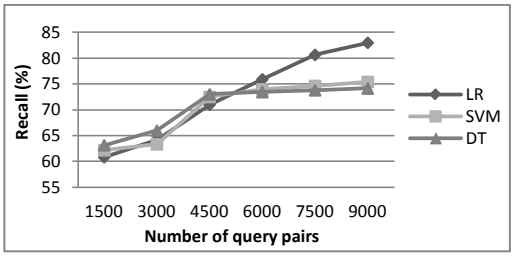

Fig. 13. Comparison result of recall. We split each set to 4/5 training and 1/5 testing

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

It can be seen from Fig. 11, Fig. 12 and Fig.13 that with the increase of training data, logistic regression algorithm can achieve higher accuracy and lower error. The highest accuracy reached 83.36%, the lowest error rate is only 17.89% and the highest recall reached 83.01%. In addition, according to the curves in Fig. 11, Fig. 12 and Fig.13 we can see that the difference between logistic regression algorithm and other two algorithms is small ini-tially, and becomes larger later. This is because not all the queries belong to the current search task in some exploratory search session. Sometimes users submitted some queries unrelated to the current search task. For example, in the "selecting the birthday gift" search task mentioned earlier, the following search process may appear.

Q1: "fresh flowers"
Q2: "Beijing weather"
Q3: "rose"
Q4: "France Red wine"
Q5: "red wine prices"
Q6: "red wine types"

It is obvious that the query "Beijing weather" is irrelevant to the "selecting the birthday gift" search task. Thus the query pair {Q1 "fresh flowers",Q2 "Beijing weather"} does not belong to the search goal shift query pairs. Then this query pair was marked as 0 (non-search goal shift query pair) in the training set. However the attribute values of this query pair are close to the attribute values of the search goal shift query pair rather than close to that of the normal query pair. Therefore, it belongs to noisy data for the classification algorithm and it leads to the different classification results of the three classification algorithms. The specific effects are as follows:

The weight function of the logistic regression algorithm is:

$$h(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \ldots$$

Thus this algorithm belongs to a linear model in a broad sense. In contrast to logistic regression algorithm, the weight function of SVM (polynomial kernel) algorithm is:

$$h(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1 x_2 + \theta_4 x_1^3 + \theta_5 x_2 + \theta_6 x_2^2 \ldots$$

, so this algorithm belongs to a nonlinear model in a broad sense.

When the training data is linearly inseparable, the SVM algorithm (polynomial kernel) is superior to the logistic regression classification algorithm. However when the linearly separable data are transformed into linearly inseparable data due to the noisy data, for the SVM (polynomial kernel) algorithm, there will be a lot of over-fitting for noisy data (over-considering the effects of some data). The over-fitting greatly increases the effects of noisy data on the classification results, and then reduces the accuracy of classification, the higher the degree of polynomial, the more over-fitting. For logistic regression algorithm, the emergence of a large number of noisy data will cause the under-fitting. (It doest not consider the effects of some data). In contrast to the over-fitting, the under-fitting reduces the effects of noisy data on classification results.

In addition, for the decision tree classification algorithm, the accuracy of classification relies heavily on the pruning. Once the noisy data as a decision node will greatly reduce the accuracy of the classification results and the probability of this problem increases as the depth of the decision tree increases.

According to the above analysis and test results, logistic regression algorithm is more suitable for this problem.

7.3 Evalution of recommendation method

In order to ensure the accuracy and objectivity of the evaluation results, we have evaluted our recommendation method from two aspects, the recommendation results and the user experiences.

7.3.1 Comparison of the recommendation results

Ground Truth for test: We retrieved the 5,000 search goal shift query pairs from section 7.2, and selected the first query in each query pair (the user first submitted) and counted the number of occurrences. And then we selected the top 1000 queries as the test set, and finally collected the ground truth data for each selected query. Given a query q, we extracted all queries q′ (q and q′ are composed of a search goal shift query pair) from the 5000 search goal shift query pairs obtained in section 7.2 and taked them as ground truth data of q, such as $GT(q) = \{q'_1, q'_2, ..., q'_n\}$.

Baseline method: We compared our design recommendation method (GSES) with the following two methods that also support exploratory search based on query recommendation (mentioned in section 2.2).

1) Topic-Oriented Exploratory Search (TOES): TOES is a query recommdation method based on the topic semantic association graph. The topic semantic association graph has been built by hyperlinks on the Internet [14].
2) Exploratory Search Based on Search Task (STES). STES is a query recommdation method based on the task graph. The task graph is built by different search tasks that have been identified based on entities and syntactic structure pattens of queries from user logs [13].

Evaluation Metrics:

1) **Accuracy:** We used a well-known metrics: Normalized Discount Cumulative Gain (NDCG) [33] to measure the accuracy. In our work, the NDCG is calculated as:

$$NDCG@N(q) = \frac{\sum_{i=1}^{n} 2^{rel_i} - 1 \big/ \log(i+1)}{IDCG(q)} \quad (27)$$

where the Ideal DCG (IDCG) is calculated from the training set itself, and the rel_i is the relevance score of the i-th query in the recommendation result. It can be computed based on a Google distance [34] as follows:

$$Dis_{google}(r_i, GT(q)) = \frac{\sum_{j=1}^{c} 1 - \frac{\max\{\log N_{r_i}, \log N_j\} - \log N_{r_i \wedge j}}{\log N - \min\{\log N_{r_i}, \log N_j\}}}{C} \quad (28)$$

$$rel(i) = \begin{cases} 1 & if\ Dis_{google}(r_i, GT(q)) < threshold \\ 0 & otherwise \end{cases} \quad (29)$$

where C is the number of queries in groud truth. N represents the total number of webpages indexed by Google. N_{r_i} is the number of webpages matching the recommended query r_i. $N_{r_i \wedge j}$ is the number of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

webpages matching both the recommended query r_i and the j-th query in the ground truth.

2) **Novelty:** The recommendation result is successful when recommended queries are relevant to search goal as well as they can show new information to users. Novelty is defined as a metric to measure the ability to explore new knowledge [14].

Assuming that a query k is one of the queries in the ground truth and a query r is one of the queries in the recommendation reuslt. The difference between them is computed as follows:

$$\pi(k,r) = \frac{N_{r\wedge\neg k}}{N_k} \tag{30}$$

where $N_{r\wedge\neg k}$ is the number of webpages matching r but not k. Then, the novelty of the query q is computed as follows:

$$Novel(q) = \frac{\sum_{j}^{c}\sum_{i}^{m}\pi(k_j,r_i)}{C\times M} \tag{31}$$

where M is the number of queries in the recommendation result.

Experimental results and analysis:

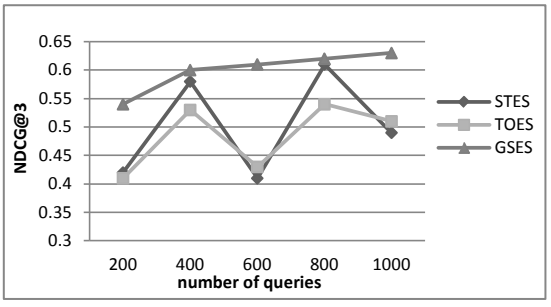

Fig. 14. Comparison result of NDCG@3

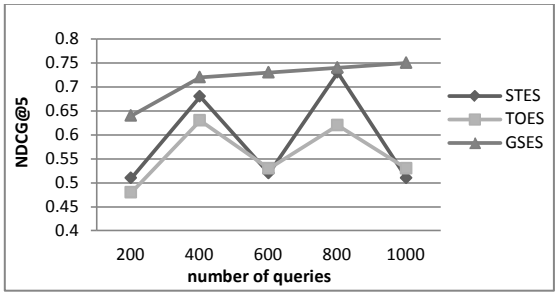

Fig. 15. Comparison result of NDCG@5

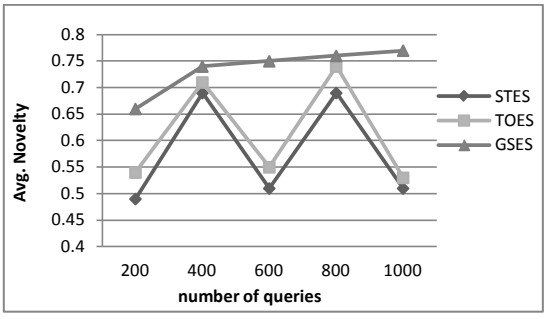

Fig. 16. Comparison result of Novelty

According to the comparison results in Fig. 14, Fig.15 and Fig. 16, our method can achieve higher accuracy and higher novelty than the other two methods. In addition, according to the accuracy curve and novelty curve it can be seen that our method from the beginning of the experiment to the end of the experiment has been relatively smooth. Unlike the other two methods are constantly fluctuating. This is because the other methods and our method all build a directed graph by query keywords, and then use the random walk algorithm to obtain the query recommendations in the graph, but the ways to add edges of the directed graph are different. In the TOES method we connect two nodes with an edge if there are one or more hyperlinks between the two webpage groups obtained by the keywords of the two nodes. In the SETS method we connect two nodes with an edge if the different queries between the two nodes belong to the same session. In our method we connect two nodes with an edge if there are one or more search goal shift query pairs between the two nodes.

In contrast to the STES method and the TOES method, our method can be considered as a filter for the edge set built by the other two methods. Only the search goal shift edges have been retained, which ensures that rank values of nodes are only affected by the search goal shift edges. Therefore, the probability of the top-k nodes becoming new search goals is increased.

When the filtered edges have little contribution to rank values of the top-k nodes, the recommended results provided by the other two methods are similar to ours. However, when the filtered edges have a greater contribution to the rank values of the top-k nodes, the recommended results provided by the other two methods are significantly worse than ours. The reason is that the recommended the top-k nodes are connected by too many non-search goal shift edges, which means that walking to these nodes doesn't need to change the current search goal and the probability of becoming new search goals is lower.

Therefore, when users are losing the interests of current search goal, our method always provides users with good recommendation results, while the recommended results of STES or TOES have large fluctuations.

7.3.2 Comparison of the user experiences

Experimental process: We recruited 120 students from our university to participate in this experiment. All participants were divided into three groups on average, and performed the four exploratory search tasks designed in Section 3. The first group adopted the TOES method to support the user's exploratory search, the second group adopted the STES method and the third group adopted the GSES method. In addition, in order to ensure that the participants' own knowledge does not affect the exploratory nature of the tasks, we conducted a background survey of the participants to ensure that the participants were not experts or researchers in the topics of the search tasks.

Baseline method:

1) Topic-Oriented Exploratory Search (TOES)
2) Exploratory Search Based on Search Task (STES)

Evaluation Metrics:

1) **Recommendation utilization rate:** It indicates the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2815544, IEEE Transactions on Knowledge and Data Engineering

proportion of recommended queries adopted by a user in a task.

$$Usage = \frac{Q_{recommend}}{Q} \times 100\% \quad (32)$$

where $Q_{recommend}$ represents the number of recommended queries adoted by the user. Q represents the number of all queries submitted by a user in a task.

2) **Search time:** It is the time spent by a user during the whole task.
3) **Rollback seach rate:** It indicates the proportion of rollback queries in a task.

$$Roll = \frac{Q_{rollback}}{Q} \times 100\% \quad (33)$$

where $Q_{rollback}$ represents the number of rollback queries.

Rollback Seach: As the user is not familiar with the background knowledge of the search task, when the user is lossing the interests of the current search goal, they often can not find related and novel search goals. They had to research with the previous query or even the initial query. For example, when a user performed the 4-th task, the search process is as follows:

1:20:15 PM query: French students birthday gift
1:20:55 PM query: French people like flowers
1:20:58 PM query: French students birthday gift

When the user took the "flower" as the current search goal, he/she did not find the final wanted information, and also did not find a new direction of exploration. The user had to fall back to the previous query "French students birthday gift" to refind a new direction of exploration, which means that a rollback search occurs.

Experimental results and analysis:

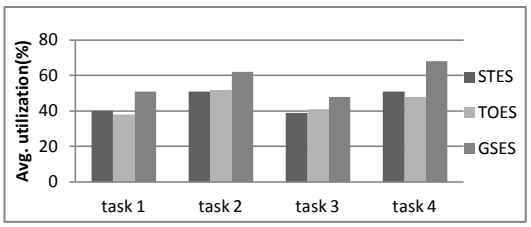

Fig. 17. Comparison result of utilization rate

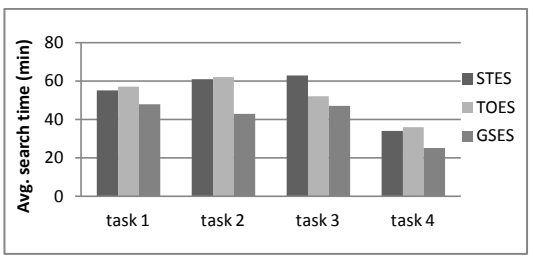

Fig. 18. Comparison result of search time

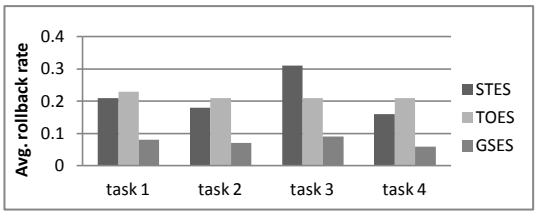

Fig. 19. Comparison result of rollback search rate

According to the comparison result in Fig. 17, the utilization rate of the method we designed is higher than the other two methods about 10%. This means that our method can better satisfy the user needs relative to the other two methods during the search goal shift processes. According to the comparison results shown in Figure 18 and Fig. 19, after using our method, the user's search time and the number of roallback search are less than the other two methods. This shows that users can effectively shorten the search time to improve search efficiency using our method.

8 Conclusion

In this paper, we studied the search goal shift which is one of the important behavioral characteristics of exploratory search, and designed a new query recommendation method based on the search goal shift to support exploratory search. The method uses machine learning to dig out all queries during search goal shift processes from search engine logs to build the search goal shift graph, and uses random walk algorithm to obtain query recommendations in the search goal shift graph. At the same time, we proved the effectiveness of the recommendation method by the comparative experiments with other methods.

9 Acknowledgments

This work was supported by the National Natural Science Foundation Program of China (61572116, 61572117, 61502089), the National key Techonlogy R&D Program of the Ministry of Science and Technology (2015BAH09F02), and the Special Fund for Fundamental Research of Central Universities of Northeastern University (N150408001, N150404009, N161606003).

References

[1] White R W, Roth R A. "Exploratory search: Beyond the query response paradigm," *Synth Lect Inf Concept Retr Serv*, vol. 1, no. 1, pp. 1-98, 2009.
[2] Baeza-Yates R, Hurtado C, Mendoza M. "Query recommendation using query logs in search engines," in *Proc. International Conference on Extending Database Technology*, pp. 588-596, 2004.
[3] Craswell N, Szummer M. "Random walks on the click graph," in *Proc. The 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 239-246, 2007.
[4] Cao H, Jiang D, Pei J, et al. "Context-aware query suggestion by mining click-through and session data," in *Proc. The 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp.875-883, 2008.
[5] Mei Q, Zhou D, Church K. "Query suggestion using hitting time," in *Proc. The 17th ACM conference on Information and knowledge management,* pp. 469-478, 2008.
[6] Boldi P, Bonchi F, Castillo C, et al. "Query suggestions using query-flow graphs," in *Proc. The 2009 workshop on Web Search Click Data,* pp. 56-63, 2009.
[7] Bates M J. "The design of browsing and berrypicking techniques for the online search interface," *Online review*, vol. 13, no. 5, pp. 407-424, 1989.

[8] Kuhlthau C C. "Inside the search process: Information seeking from the user's perspective." *Journal of the American society for information science*, vol. 42, no. 5, pp. 361-424, 1991.

[9] Byström K, Järvelin K. "Task complexity affects information seeking and use." *Information processing & management*, vol. 31, no. 2, pp. 191-213, 1995.

[10] Marchionini G. "Exploratory search: From finding to understanding." *Communications of the ACM,* vol. 49, no. 4, pp. 41-46, 2006.

[11] Donato D, Bonchi F, Chi T, et al. "Do you want to take notes?: identifying research missions in Yahoo! search pad," in *Proc.* The 19th ACM international conference on World Wide Web, pp. 321-330, 2010.

[12] Qvarfordt P, Golovchinsky G, Dunnigan T, et al. "Looking ahead: query preview in exploratory search," in *Proc.* The 36th international ACM SIGIR conference on Research and development in information retrieval, PP. 243-252, 2013.

[13] Hassan Awadallah A, White R W, Pantel P, et al. "Supporting complex search tasks," in *Proc.* The 23rd ACM International Conference on Conference on Information and Knowledge Management, pp. 829-838, 2014.

[14] Sun H C, Jiang C J, Ding Z J, et al. "Topic-Oriented Exploratory Search Based on an Indexing Network." *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no.2, pp. 234-247, 2016.

[15] Ksikes, A. "Towards exploratory faceted search systems." Doctoral dissertation, University of Cambridge, 2014

[16] Zhang Y, Cheng G, Qu Y. "Towards exploratory relationship search: A clustering-based approach," in *Proc.* Joint International Semantic Technology Conference. Springer International Publishing, pp. 277-293, 2013.

[17] Bron M, Van Gorp J, Nack F, et al. "A subjunctive exploratory search interface to support media studies researchers," in *Proc. ACM SIGIR conference on Research and development in information retrieval*, pp. 425-434, 2012.

[18] Bespinyowong R, Chen W, Jagadish H V, et al. "ExRank: an exploratory ranking interface." Proceedings of the VLDB Endowment, vol. 9, no. 13, pp.1529-1532, 2016.

[19] Peltonen J, Strahl J, Floréen P. "Negative Relevance Feedback for Exploratory Search with Visual Interactive Intent Modeling," in *Proc.* The 22nd ACM International Conference on Intelligent User Interfaces, pp.149-159, 2017.

[20] Kules B, Capra R. "Designing exploratory search tasks for user studies of information seeking support systems," in Proc. *The 9th ACM/IEEE-CS joint conference on Digital libraries*, pp. 419-420, 2009.

[21] Hassan A, White R W, Dumais S T, et al. "Struggling or exploring?: disambiguating long search sessions," in *Proc. The 7th ACM international conference on Web search and data mining*, pp. 53-62, 2014.

[22] Real R, Vargas J M. "The probabilistic basis of Jaccard's index of similarity." Systematic biology, vol. 43, no. 4, pp. 380-385, 1996.

[23] Levenshtein V I. "Binary codes capable of correcting deletions, insertions, and reversals." Soviet physics doklady, vol. 10, no. 8, pp. 707-710, 1966.

[24] Leacock, Claudia, and Martin Chodorow. "Combining local context and WordNet similarity for word sense identification." WordNet: An electronic lexical database, vol. 49, no. 2, pp. 265-283, 1998.

[25] Turney, Peter D., and Patrick Pantel. "From frequency to meaning: Vector space models of semantics," Journal of artificial intelligence research, vol. 37, pp. 141-188, 2010.

[26] Mikolov T, Yih W, Zweig G. "Linguistic regularities in continuous space word representations," in Proc. The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 746-751, 2013.

[27] Bouma, Gerlof. "Normalized (pointwise) mutual information in collocation extraction," in Proc. Proceedings of GSCL, pp. 31-40, 2009.

[28] Yue, Yisong, Rajan Patel, and Hein Roehrig. "Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data," in Proc. The 19th international conference on World wide web. ACM, pp. 1011-1018, 2010.

[29] Fox, Steve, et al. "Evaluating implicit measures to improve web search." ACM Transactions on Information Systems (TOIS), vol. 23, no. 2, pp. 147-168, 2005.

[30] Fogaras D, Rácz B, Csalogány K, et al. "Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments," *Internet Mathematics*, vol. 2, no. 3, pp. 333-358, 2005.

[31] Keerthi, S. Sathiya, et al. "A sequential dual method for large scale multi-class linear SVMs," in Proc. the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 408-416, 2008.

[32] Quinlan, J. Ross. "C4. 5: programs for machine learning." Elsevier, 2014.

[33] Järvelin, Kalervo, and Jaana Kekäläinen. "IR evaluation methods for retrieving highly relevant documents," in Pro. The 23rd annual international ACM SIGIR conference on Research and development in information retrieval. ACM, pp. 41-48, 2000.

[34] Cilibrasi, Rudi L., and Paul MB Vitanyi. "The google similarity distance." IEEE Transactions on knowledge and data engineering, vol. 19, no. 3 , 2007.

Chao Ma received his MS degree in software engineering from the Dalian University of Technology in 2005. Now he is a PhD candidate at the College of Information Science and Engineering, Northeastern University. His current research interests include web engineering and optimization algorithm.

Bing Zhang received his PhD degree in computer science from the Northeastern University. Now he is a professor of Northeastern University. His current research interests include service computing, cloud computing and web engineering.