World Wide Web (2014) 17:1161–1188
DOI 10.1007/s11280-013-0233-5

Finding similar queries based on query representation analysis

Yuan Wang · Jie Liu · Jimeng Chen · YaLou Huang

Received: 5 September 2012 / Revised: 16 May 2013 /
Accepted: 6 June 2013 / Published online: 29 June 2013

Abstract In order to understand user intents behind their queries, many researchers study similar query finding. Recently, the click graph has shown its utility in describing the relationship between queries and URLs. The previous approaches mainly either generate related terms or find relevant queries based on the co-clicked URLs. However, these approaches may suffer from the complexity of natural language processing and click-through data sparseness. In this paper, we tackle this problem through three query probability distribution representation models: Click Model, Term Model, and Semantic Model. The Click Model extracts credible transition probability from queries to URLs, and describes a query without considering web contents. The Term Model focuses on representing a query via term distribution over its main entities and purposes, which can better capture information needs behind short and ambiguous keyword queries. The Semantic Model learns potential intent distribution of queries to distinguish user intents behind a query. Among the three models, we apply pairwise similarity metrics and graph-based personalized pagerank to find similar queries. Compared to traditional representation models, our representation models are verified to be effective and efficient, especially for long tail queries.

Y. Wang · J. Liu (✉) · J. Chen
College of Information Technology Science, Nankai University, Tianjin, China
e-mail: jliu@nankai.edu.cn

Y. Wang
e-mail: yayaniuzi23@mail.nankai.edu.cn

J. Chen
e-mail: jeanchen@mail.nankai.edu.cn

Y. Huang
College of Software, Nankai University, Tianjin, China
e-mail: huangyl@nankai.edu.cn

Keywords Query log analysis · Click graph · Query representation · Query similarity · User intent

1 Introduction

In a typical search scenario, users search information via keyword-based queries. However, due to vocabulary mismatch and query ambiguity, it is a challenging task to provide appropriate and relevant documents according to the queries issued by users. Luckily, similar query finding is considered as an effective strategy to understand the original queries. It benefits various Information Retrieval (IR) applications, such as query understanding [11], query suggestion [30] and even Web advertisement recommendation [27, 31].

Previous methods for similar query finding can be separated into two major categories: *content-based metrics* and *behavior-based metrics*. In *content-based metrics*, similarity between queries is separately measured by words in queries or relevant documents of queries. By considering the relevant content of queries, some researchers propose the sequential dependence model [25], the latent concept expansion model [26], etc, to find similar queries. Meanwhile, others consider about the utility of top-k relevant documents from search engines to represent a query. As a result, kernel functions [29], topic models [15] are applied to compute query similarity. In *behavior-based metrics*, similarity between queries is extracted from a click graph [10, 23, 24, 30] or a query-flow graph [4–6]. They define the similarity of adjacent queries and propagate the similarity via users' behavior profile (click logs or sessions). However, both methods have some problems. *Content-based metrics* may suffer from the complexity of natural language processing (NLP) [7] and are hard to capture users' search intents. As to *behavior-based metrics*, click graphs are over dependent on users' click, thus it lacks of content understanding of queries. While the major problem of query flow graphs is that queries in a session may have different search intents, so that it may cause dissimilar connections and concept drift.

In order to solve above problems and find better similar queries, we analyze users' interaction process with search engines. Two typical search scenarios are as follows. In one case, a user wants to travel by aeroplane and she/he submits a query "aa" to search engines. Before viewing page contents, she/he only sees the snippets which introduce web pages in the search results list. And then, she/he is attracted by "American Airlines—Airline tickets and cheap flights at AA.com" (the title of "http://www.aa.com/"). So she/he clicks "http://www.aa.com/". In the other case, a user wants to find out information about "Alcoholics Anonymous". She/He may probably also submit the query "aa" to search engines. After that, she/he is attracted by "Alcoholics Anonymous" (the title of "http://www.aa.org/") and clicks this URL. Both click behaviors will be recorded in logs. These typical behaviors indicate that the motivation of click behaviors is largely because of high relevance between users' intents and snippets of URLs rather than the relevance between their intents and web page contents of URLs [22]. So the snippets, which are shown on the results list, make a great help for users' information retrieval and result in further their final click behaviors. These situations motivate us to capture the semantic relations between queries and URLs by using both users' click information and content information.

In this paper, we propose a two-stage method for similar query finding. In the first stage, we establish a hierarchical structure for query representation. In the second stage, we find similar queries to original queries.

In the first stage, we propose an effective query probability distribution representation framework based on both users' click bipartite graph and the content related to clicked URLs. The representation framework is a hierarchical structure. In the bottom layer, we construct a bipartite graph between queries and clicked URLs (called *Click Model*). Based on the bottom layer, we propose a method to capture relationships between queries and content of clicked URLs (called *Term Model*). Furthermore, we map queries into an intent space (called *Semantic Model*). First, *Click Model* provides a robust transition probability (or called URL description probability) for query representation. The model considers each query as a document which is filled with URLs. In another words, URLs are the basic elements that represent a query. Similar to $cf \cdot iqf$, we consider description ability of each URL and leverage clicks on click graphs. Second, *Term Model* is based on transition probability defined in *Click Model* and users' behavior analysis. *Term Model* uses the expected number of terms to represent queries. The expected number is not a simple counting process, but a process of computing term expectation based on the transition probability from a query to URLs. *Term Model* combines both click information and semantic information seamlessly. We denote this as a term expectation distribution for query representation. Third, from terms to intents, we analyze topic distribution of queries based on *Term Model* and treat queries as a process of generation (called *Semantic Model*). In this model, we apply topic modeling techniques on queries which have been represented by a bag of terms in *Term Model*. And then we map queries into a semantic space.

In the second stage, we find intent relevant and similar queries to queries issued by users with these three representation models. Based on the learned representation models, queries are represented as probability distribution over different morphemes in different representation spaces. Typically, we employ both pairwise similarity metrics and graph-based similarity metrics to find similar queries. Pairwise metrics (such as Cosine similarity or Jaccard similarity coefficient, et al.) measure query similarity based on the distance and positional information in query representation spaces. Graph-based metrics (such as personalized pagerank or random walk, et al.) mostly measure relationships between queries by query transition graphs. The two kinds of methods help us make use of query representation models to discover query relationship (query similarity in this paper).

Our models can provide an effective way to organize web log information. Different from the previous works, our models have three advantages. Firstly, we propose URL transition probability in Click Model to leverage the relationships between popularity of queries and description power of URLs for query representation. Secondly, we propose term expectation in Term Model to capture the main entities and intents of users' information needs. Thirdly, we have tackled the problem of vocabulary mismatch and synonymous query finding by mapping queries into semantic spaces in Semantic Model. In the end, we can discover more semantic related and similar queries for original queries issued by users.

The rest of the paper is organized as follows. The related work is reviewed in Section 2. Section 3 discusses three query representation models in detail. Then we present the similarity metrics to find similar queries in Section 4. Experimental results have been described in Section 5. Finally, we present our conclusion and future work in the last section.

2 Related work

Our work is highly related to the research on *query similarity* and *query representation*. Query similarity focuses on metrics of similarity between queries, while query representation concentrates on representing users' information needs.

2.1 Query similarity

Calculating query similarity is an interesting but challenging problem. A trustworthy query similarity metric can benefit various IR applications in most scenarios, such as query reformulation [11, 21], query suggestion [8, 13, 23, 29, 30] and even web advertisement recommendation [27]. However, since queries are short and ambiguous, it's too difficult to learn users' actual intent only based on terms in queries. So many previous works either take advantage of related content to calculate query similarity (*content-based metrics*), or make use of users' behavior to discover relevant semantic association paths from one query to other queries (*behavior-based metrics*).

2.1.1 Content-based metrics

Content-based metrics measure query similarity based on content similarity independently, where content often comes from queries themselves or related documents and is highly related to the original queries. Owing to vocabulary mismatch and shortness of queries, various data sources have been applied to enrich the description of queries, mainly including terms, bigrams, nouns and entities in content of queries and content from related documents. Some methods find similar queries by query content analysis. Metzler and Croft [25] introduced a sequential dependence model, which took advantage of phrase structure. While the latent concept expansion model [26] added new term proximity features, and then collected similar queries. Besides query content analysis, other works take advantage of content from related documents. Generally speaking, they collect content from search results to represent a query. On the one hand, many researchers apply similarity metrics on Vector Space Model (VSM) [1, 20], including Cosine similarity, Jaccard coefficient, term overlap and edit distance, to find similar queries. On the other hand, language models also can be employed to find similar queries. One of the typical works is Kernel function [29]. It has been proved effectively based on the TF-IDF weighted vectors of search result snippets. Guo et al. [15] leveraged Cosine similarity and query semantic models to calculate query similarity under each intent learnt from search result snippets of queries.

2.1.2 Behavior-based metrics

Behavior-based metrics measure query similarity based on the behavior relationship between queries, and the similarity between queries is defined on click graphs [10, 23, 24, 30] or query-flow graphs [4–6].

The click graph is a bipartite graph between queries and URLs. The basic idea is that user click behavior can be modeled as a bipartite graph, which is an essential technique to describe the semantic association information contained in the search logs. Craswell et al. [9] described two strategies of random walk to find similar queries. Mei et al. [24] proposed hitting time on the graph to evaluate the similarity between queries. Deng et al. [10] considered information entropy of each URL, and then proposed the $cf \cdot iqf$ model to adjust weights in click graphs. Besides, some researchers considered additional information [23], such as relationship between users and queries. Ma et al. [23] proposed a joint matrix factorization method to build a query similarity graph based on the low-rank features learnt from query-URL bipartite graphs and user-query bipartite graphs. Furthermore, due to sparseness of click graphs, Yi and Allan [33] added missing clicks to click graphs, and then calculated content similarity on the graphs. Liu et al. [22] used the snippet of clicked URLs to understand queries and built global and local snippet click models to discover related keywords for original queries. They proposed a detailed analysis on reasons of users' click behavior, which showed that a series of users' clicks was a process of "what you see and click is what you want". It is a similar work as ours, but we consider users' real intent by establishing a hierarchical graph including semantic and behavior knowledge instead.

The query-flow graph is a directed graph with queries as nodes, query co-occurrence relationship in sessions as edges. The graph is constructed based on the assumption that two queries are relevant when they have appeared in the same sessions. So the similarity between two queries can be propagated to any associated queries on the graphs. Bordino et al. [5] applied random walk to find similar queries based on the transition probability between queries on the query-flow graphs. Subsequently, Bordino [6] projected the graph on a low-dimensional Euclidean space, and then mapped graph nodes into geometric spaces so that the distance distortion was minimized. In addition, Song et al. [30] employed pagerank and learning to rank methods on a topic-based term-transition graph in order to leverage user re-query feedbacks for query suggestion. But the same session may contain queries in different intents, we haven't considered session information here.

There are also some works that leverage both click graphs and query-flow graphs to understand queries. In [17], Hu et al. represented queries as major subtopics, where each subtopic was represented by a cluster filled with related URLs and keywords. In [8], Chen et al. applied $cf \cdot iqf$ on both behavior graphs and semantic graphs to find similar queries.

Many previous researches adopt either *content-based metrics* or *behavior-based metrics*. However, they haven't addressed that how models learn users' information needs. In *content-based metrics*, because queries are really short and unintelligibly in most instances, term based methods may suffer from the complexity of natural language processing(NLP) [7]. Usage of search results enhances the search engines' IR ability, but ignores users' actual intents underlying queries. The above two problems may neglect both the users' actual information needs and find queries with irrelevant intents. While in *behavior-based metrics*, click graphs are over dependent on users' clicks, but the graphs lack the semantic understanding of users' actual information needs. As to query-flow graphs, they suffer from the major problem that queries in the same session may have different search intents (or called "concept drift"), which will cause dissimilar connections.

To find similar queries reasonably and effectively, we take both content information and behavior information into consideration. In this paper, we apply a scoring rule, it is used extensively in information retrieval to leverage the importance and description ability of URLs on the query-URL bipartite graph. Therefore we model users' information needs from the content of clicked documents, and directly apply pairwise similarity metrics or random walk to find semantic similar queries. In this way, we generate similar queries to make clear and explain users' original information needs.

2.2 Query representation

Understanding users' real information need underlying a short query has long been treated as an important and difficult part of effective information retrieval. Despite this, early query representation models focus on query reformulation to overcome the situation of synonymy and polysemy, which easily lead to vocabulary mismatch and ambiguity. Thus query representation models are devoted to make relationships between queries clear. These previous methods can be divided into three categories: *query term based* [2, 25, 26], *search results based* [15, 29] and *click data based* [10, 18].

Query term based methods depend on the assumption that similar queries have relationship with each other when they share common words, chunks or nouns. Croft's earlier work focused on relevance between terms: the sequential dependence model [25] added phrase structure to queries issued by users, and the latent concept expansion model [26] added new term proximity features and words. In 2012, Bendersky and Croft [2] built a hypergraph structure using various morphemes such as terms, bigrams, nouns and entities to represent a query. In [32], Xue et al. generated a reformulation tree based query representation to organize multiple sequences of reformulated queries as a tree structure.

Search results based techniques tackle the problem by enriching expressions with additional features from top-k results returned by the search engines [15, 29]. Researchers collected content or URLs in search results lists to represent a query. Furthermore, topic models [15] and kernel functions [29] also have been applied to model a query.

Click data based methods get abundant information from click logs. In [17], Hu et al. leveraged users' click information and mined major subtopic intents for query representation, where each subtopic was represented by a cluster. Deng et al. [10] represented queries as a URL vector based on query-URL bipartite graph.

Query representation is a basic and crucial work in IR. An effective and appropriate query representation model can benefit lots of applications and researches. Previous representation models introduce terms, search results and click data to comprehend users' information needs. However, in *query term based* methods, it's very common for two similar queries with less or no overlap in terms. So methods have evident advantages on long queries rather than short queries. Search results provide rich contextual information for queries automatically, but it is not a substitute for users' judgments. Meanwhile, search results still need a better way to organize and display. *Click data based* methods are more reliable than the other two. They stimulate us to propose a transition probability from a query to URLs and to use titles of clicked URLs to represent users' information needs released through their interaction with search engines.

3 Query representation models

The basic idea of our models is that users' real information needs are released via the titles of their clicked URLs. We dedicate this section to describe our query representation models in detail, namely Click Model, Term Model and Semantic Model. In each model, we regard each query as a document. The morphemes in these documents (here are queries) are URLs, terms, and intents, respectively. Conversely, our representation framework includes three levels. The first level is behavior level (Click Model), which shows users' clicking behavior information. The second and third levels are both semantic levels. The second level is fine-grained semantic level (Term Model), which indicates users' actual semantic information needs using terms in the retrieval process. The third level is coarse-grained intent level (Semantic Model), which represents users' search intents with distinguishing features (also called topics in text mining).

In this section, we firstly introduce the preliminaries and notations, and then discuss about models in detail. Secondly, we try to give the answers to following questions: how can we leverage the relationship between queries and URLs contained in the entire click log effectively (in Section 3.2), why do titles contain the information needs of users (in Section 3.3), and how to combine both behavior relationship and semantic information for query representation (in Section 3.4).

3.1 Preliminaries and notations

We assume that there is a collection of M unique queries $Q = \{q_1, \cdots, q_M\}$ sharing a set of K potential search intents $S = \{s_1, \cdots, s_K\}$. Let $U = \{u_1, \cdots, u_D\}$ be the set of D URLs clicked for these queries. The contents of these URLs share the same set of V terms $T = \{t_1, \cdots, t_V\}$. The term frequency of term t_j in the content of URL u_k is denoted as tf_{kj}. The query-URL bipartite graph is an undirected graph $G = (Q \bigcup U, E)$. For each pair $(q_i, u_j) \in E$, the weight w_{ij} of the edge is the raw number of times that the URL u_j was clicked in response to the query q_i. We count the number of different queries connected with u_j and denote it as $n(u_j)$. We consider the total number of clicks after the submission of query q_i as len_i. Considering query q_i as a document (here is called "a query document") composed of the clicked URLs as disordered terms ("a bag of URLs"), len means the "query document length".

For query representation models, we use Q^C to mark the representation matrix in Click Model, where q_{ij}^C means the weight of URL u_j that describes query q_i. We use Q^T to mark the representation matrix in Term Model, where q_{ij}^T means the weight of term t_j that describes query q_i. Q^S is used for marking the representation matrix in Semantic Model, where q_{ij}^S means the weight of intent s_j that describes query q_i. We summarize notations in Table 1.

3.2 Click model

In this section, we propose Click Model to leverage the relationship between queries and URLs contained in the entire click log effectively. Due to shortness and ambiguity of queries, it's difficult to learn similar intent under different keywords. Luckily, users' behaviors introduce a new way of representation similar queries. In

Table 1 Notations of our model

Symbol	Description
M	The number of unique queries
K	The number of potential search intents
D	The number of unique URLs
V	The number of terms
Q	The set of queries
S	The set of intents
U	The set of URLs
T	The set of terms
tf_{kj}	The term frequency of term t_j in the content of web page u_k
w_{ij}	The raw number of times that URL u_j connected with query q_i
$n(u_j)$	The number of different queries connected with URL u_j
len_i	The total number of clicks after the submission of query q_i
Q^C	The representation matrix in Click Model
Q^T	The representation matrix in Term Model
Q^S	The representation matrix in Semantic Model

click logs, we find that users will click similar URLs to their similar queries. So a large-scale bipartite graph is constructed to describe relevance between queries and URLs by using the click log.

Figure 1 is a toy example, with 4 queries and 3 URLs. As Figure 1 shows, queries can be explained by all the URLs that are connected to them on the graph. On the basis of frequency-based function, the raw click numbers can be normalized to represent a query. However, each URL contains different capacity to describe a query. The more queries a URL connects with, the weaker discriminatory power it has to describe a unique query. This situation has been already proved in [10]. For example, query "apple" has an equal number of clicks to "http://www.google.com/mobile/ipad/" and "http://www.apple.com". So in frequency-based function, u_1 and u_2 will get a same score of 0.5. But in fact, u_1 also describes and connects with "ipad" and "Google mobile". That means u_1 can hit the target of more search intents and isn't an exact description for "apple". Even though URL u_1 and URL u_2 share the

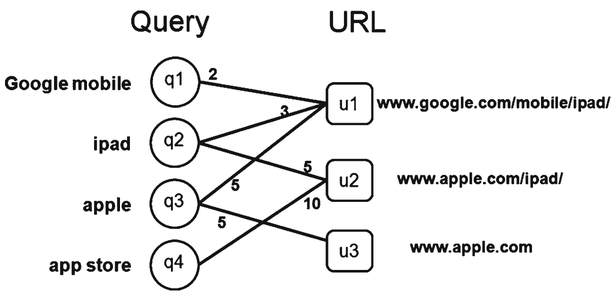


Figure 1 An example of query-URL bipartite graphs

same number of clicks to q_1, URL u_1 has a weaker discriminative power for query q_1 compared against URL u_2. The circumstance is very similar to the reason why researchers proposed a numerical statistic of TF-IDF in text mining.

According to the analysis on query-URL graphs, we apply OKAPI scoring function [28] to adjust the weight of edges from queries to URLs. Here we just choose the basic OKAPI function to illustrate our idea. Besides, many BM25-style functions also can be applied to our model. We define cf (click frequency) and iqf (inverse query frequency) to measure the descriptive ability of URL u_j to a special query q_i. The formulas are:

$$cf\left(q_i, u_j\right) = \frac{w_{ij} \cdot (k_1 + 1)}{w_{ij} + k_1 \cdot \left(1 - b + b \cdot \frac{len_i}{avglen}\right)}, \quad iqf\left(u_j\right) = \log \frac{M - n(u_j) + 0.5}{n(u_j) + 0.5}, \quad (1)$$

where k_1 and b are free parameters in BM25 [28] (usually chosen as $k_1 \in [1.2, 2.0]$ and $b = 0.75$). The parameters k_1 and b are used together for considering the length of query documents and the raw frequency of the URLs. k_1 is used for leveraging these two aspects. While, b is used to adjust the influence of the unique document length compared with average document length in corpus. The bigger we set b, the greater effect on relevance the length of this unique document has, and vise versa. In Click Model, $cf\left(q_i, u_j\right)$ mainly leverages the total number of clicks in query document of q_i and the raw click frequency from query q_i to URL u_j. Meanwhile, $iqf\left(u_j\right)$ evaluates discriminative description power of URL u_j. Thus, the query representation in Click Model is defined based on cf and iqf:

$$q^C_{ij} = \frac{cf\left(q_i, u_j\right) \cdot iqf\left(u_j\right)}{\sum_{u_t \in U} cf\left(q_i, u_t\right) \cdot iqf\left(u_t\right)}, \quad (2)$$

where u_t iterates the URL set U. We show the matrix transformation of the Figure 1 in Table 2. Table 2b and c show that u_1 has the lowest iqf and the expressive power of u_1 has been decreased for query representation in Click Model.

In Click Model, we leverage the relationship between queries and URLs by an adjusted click number of URLs to a query (cf) and description power of each URL (iqf). So, the model provides a credible weight for query representation by URLs. However, users' information need is not satisfied by the URLs they have clicked. Users' final click behavior is decided by the judge over the relevance between their information needs and the short content displayed in the results list. So we want to understand users' information needs by semantic analysis.

Table 2 Matrix representation for Figure 1

(a) Raw Matrix

w_{ij}	u_1	u_2	u_3
q_1	1	0	0
q_2	3	5	0
q_3	5	0	5
q_4	0	10	0

$\rightarrow$

(b) cf and iqf

$cf\left(q_i, u_j\right)$	u_1	u_2	u_3
q_1	0.814	0	0
q_2	0.753	0.822	0
q_3	0.8	0	0.8
q_4	0	0.88	0
$iqf\left(u_j\right)$	0.522	0.738	0.951

$\rightarrow$

(c) Click Model Matrix

q^C_{ij}	u_1	u_2	u_3
q_1	1	0	0
q_2	0.394	0.606	0
q_3	0.353	0	0.647
q_4	0	1	0

3.3 Term model

In this section, we take both behavior information and semantic information into query modeling. Click Model represent queries only by using users' behavior, but doesn't consider that users' click behavior is based on the judge over the relevance between their information needs and the short content (titles and related abstracts which are sensitive to queries) displayed in the results list. That means the semantic information of our search goals in web search can be revealed by the terms displayed around clicked URLs. These terms can describe users' needs in a fine-grained way. So we propose Term Model to represent queries by terms appropriately this section.

Besides, there is also another reason that prompts us to learn users' real information needs through terms of these short content. On the internet, there is a lot of duplicate content generated by copy-paste, mainly existing in news or blogs. This is a pretty common situation. For example, we search for some news of "the death of Neil Alden Armstrong". The news has appeared on pages of *Reuters*, *Chicago Tribune*, *New York Times* and et. al. Similar queries of this search intent submitted by different users, and these users may have preference on different news sites, which can cause different URL transition probability distributions between these similar queries. In spite of this, titles of the clicked web pages will probably indicate the similar information need via the same or similar terms. Luckily, Huang et al. [19] in 2010 performed a large-scale of research on examining the similarities and differences in language model properties between queries and web documents. The analysis has been conducted with three types of text streams related to web documents: the body, the title, and the anchor text. Their information theoretical analysis shows that queries seem to be composed in most similar ways to anchor texts or titles. So titles (showing on the search results list) of the clicked URLs not only show users' information need more explicitly than queries themselves, but also are composed in a similar language to queries. Meanwhile, although anchor texts are similar to queries as well, not all URLs have available anchors. Above all, the title provides the main idea and the brief summarization of a web document which can explain users' intent much better. So we select titles of the clicked web pages as a transferring language to represent queries in our Term Model.

In Term Model, queries are represented by terms in the titles of clicked web pages. We define a term expectation to represent a query based on the Click Model, which has considered description power of URLs to a query (here, we consider the presentation distribution over URLs as a transition probability from a query to URLs). So, the weight q^T_{ij} of term t_j for query q_i is defined as mathematical expectation of terms:

$$q^T_{ij} = \sum_{u_k \in U} q^C_{ik} \cdot tf_{kj} \tag{3}$$

The matrix format of (3) is:

$$\begin{bmatrix} q^C_{11} & q^C_{12} & \cdots & q^C_{1D} \\ q^C_{21} & q^C_{22} & \cdots & q^C_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ q^C_{M1} & q^C_{M2} & \cdots & q^C_{MD} \end{bmatrix} \begin{bmatrix} tf_{11} & tf_{12} & \cdots & tf_{1V} \\ tf_{21} & tf_{22} & \cdots & tf_{2V} \\ \vdots & \vdots & \ddots & \vdots \\ tf_{D1} & tf_{D2} & \cdots & tf_{DV} \end{bmatrix} = \begin{bmatrix} q^T_{11} & q^T_{12} & \cdots & q^T_{1V} \\ q^T_{21} & q^T_{22} & \cdots & q^T_{2V} \\ \vdots & \vdots & \ddots & \vdots \\ q^T_{M1} & q^T_{M2} & \cdots & q^T_{MV} \end{bmatrix} \tag{4}$$

In (4), it is clear that D is larger than V, because each term must included at least one title of a URL.

Term Model mainly has two advantages. Firstly, the model proposes term expectation to leverage click information and semantic understanding via the web pages' titles for representing search intent. Secondly, information overlapping on the internet has been well handled during modeling.

3.4 Semantic model

Due to vocabulary mismatch, Term Model have no way to find synonyms for the original queries. But we know, synonymous queries must share the same topics or intents. So in this section, we propose Semantic Model to represent queries at intent level to tackle this problem.

In both Click Model and Term Model, we think of a query as a document. And the basic components in the query documents are URLs and terms in the titles of clicked URLs respectively. Especially in Term Model, queries are represented as a document of terms with mixed semantic information. Term Model describes main entities and the possible purposes of a query based on term expectation. But we know that the term representation structure cannot discriminatively explain the multiple intents of a query, which will make the query representation models become invalid for synonymous names and statements. For example, "baseball" and "softball", in many cases, both are a kind of sports and their basic rules are always the same. That means they refer to the same thing. But due to vocabulary mismatch and query-sensitive titles of clicked results, Click Model and Term Model may become invalid. The same situation happens when representing query "yang tao" and query "kiwi fruit" (a kind of fruit). So we want to find more discriminative and semantic features to represent queries. In other words, the representation model should have the ability to judge and tell which categories queries belong to.

Based on term representation for queries in Term Model, we discover the intent distribution for queries. We assume that our query set consists of some intents, which has the same meanings as topics in a collection of documents. The intents are the aspects of queries. Take the query "Jane Eyre" as an example. The intents of "Jane Eyre" perhaps are "books", "movies" or "novels". So there comes an idea that we can represent a query with probability distribution over these discriminative intents. We apply Latent Dirichlet Allocation (LDA) [3], an algorithm for soft clustering to train our Semantic Model on queries. LDA is an unsupervised generative topic model. This model can discover underlying semantic structure of a document collection based on a hierarchical Bayesian analysis of the original texts. Based on LDA, we can discover intents contained in our query collections. And each intent is represented as a probability distribution over a fixed vocabulary of terms, where terms used to describe the same thing will appear with high probability in a unique topic. In Term Model, queries are query documents made up of terms, so a query collection is like a document collection. In Semantic Model, we choose topics learnt from LDA as semantic features (intents) to represent queries.

Next, let's look at some important details of Semantic Model. One is the estimation of parameters in Semantic Model, the other one is how we select the number of topics here.

At first, we show a list of the notations only used in this subsection in Table 3.

Table 3 Notations for Semantic Model

Symbol	Description
α, β	The hyperparameters of the Dirichlet prior on the per-query topic distributions and on the per-topic word distribution.
θ_i	The topic distribution for query q_i, denoted as q_i^S as well
ϕ_l	The term distribution for topic s_l
s_{ij}	The topic for the *j*th term in query q_i
t_{ij}	The specific *j*th term in query q_i

Our Semantic Model use topics (also called intents in this paper) probability distributions θ to represent queries. We model the generation process of Term Model by the following steps, just like the generation process of documents in LDA:

1. K, α, β has been predefined
2. For each of topic $l = 1 : K$, sample the mixture of terms $\phi_l \sim Dir(\alpha)$
3. For each of query $i = 1 : M$, sample the mixture of topics $\theta_i \sim Dir(\beta)$

   For each term $j = 1 : len_i$ in the Term Model of query q_i

   (a) sample a topic $s_{ij} \sim Multinomial(\theta_i)$
   (b) sample a term $t_{ij} \sim Multinomial\left(\phi_{s_{ij}}\right)$

The generative procedure is shown in Figure 2.

$$P(s_i = j|t_i, s_{-i}, \alpha, \beta) \propto \frac{n_{-i,j}^{(t_i)} + \beta}{n_{-i,j}^{(\cdot)} + V\beta} \cdot \frac{n_{-i,j}^{(q_i)} + \alpha}{n_{-i,\cdot}^{(q_i)} + K\alpha}, \tag{5}$$

where $s_i = j$ means the topic of term i is s_j, and t_i here only means a sign for the term i. s_{-i} represents all the assignment for s_k $(k \neq j)$. $n_{-i,j}^{(t_i)}$ is the count of term t_i assigned to the topic j, and $n_{-i,j}^{(\cdot)}$ is the total number of terms that are assign to topic j. $n_{-i,j}^{(q_i)}$ is the count of terms in query q_i being assigned to the topic j without t_i, and $n_{-i,\cdot}^{(q_i)}$ is the total number of terms that have already been assigned to topics not including the current one.

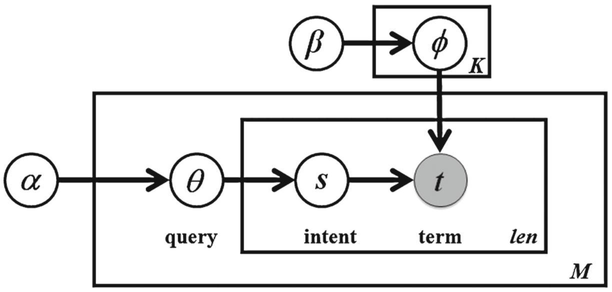


Figure 2 A plate representation of LDA for Semantic Model. Nodes denote random variables, shaded nodes stand for observed random variables and edges denote dependence between variables. The plates represent replication. The outer plate represents queries, while the inner plate represents the repeated sampling of topics and terms in each query document

Then Gibbs sampling algorithm starts with a random assignment of terms to topics and approximates posterior distribution of a topic over all terms. Thus, the parameters are estimated as follows:

$$\hat{\phi}_l^{(t)} = \frac{n_l^{(t)} + \beta}{n_l^{(\cdot)} + V\beta}, \tag{6}$$

$$\hat{\theta}_i^{(s)} = \frac{n_i^{(s)} + \alpha}{n_i^{(\cdot)} + K\alpha}, \tag{7}$$

where n represents a count matrix, $n_l^{(t)}$ is the number of term t assigned to topic s_l, $n_l^{(\cdot)}$ is the total number of terms assigned to topic l in the whole corpus, $n_i^{(s)}$ is the number of terms assigned to topic s in query q_i, $n_i^{(\cdot)}$ is the number of all times topics are assigned to query q_i. See more detailed derivation in [14].

Before parameter estimation, we should decide the number K of topics. In LDA training, the number of potential search intents is an important parameter and should be predefined. We select it by a process of clustering. After clustering, the number of clusters will be set as the number of topics. We use DBSCAN [12] after features selection to get the number of topics in our log. DBSCAN is one of automatic density-based clustering algorithms. Based on the term representation in Term Model, the density-based clustering algorithm is suitable for the topic number decision.

In the paper, we use terms in Term Models as basic features of each query and use q^T as the initial weights of features. Because different features have different distinguishability, we select high distinguishability terms for clearing the difference between queries. Firstly, we normalize the term expectation value calculated in Section 3.3 into the range of [0, 1], marked as $Q^{T'}$. Secondly, we calculate quality of terms as follows:

$$Quality\left(t_j\right) = \sum_{q_i \in Q} (q_{ij}^{T'})^2 - \frac{(\sum_{q_i \in Q} q_{ij}^{T'})^2}{M} \tag{8}$$

Thirdly, we rank the terms by the scores of $Quality\left(t_j\right)$ and select top-ranking terms (here we choose 10 % of terms). In $Q^{T'}$, we keep the original weight of selected terms, and the weights of other terms are set to 0. After clustering by DBSCAN, we get 18 clusters. So 18 is pre-defined number of topics for our Semantic Model at last.

In Figure 3, we show intent distribution of the query “The Chinese Paladin 2”.

3.5 Summary

The query representation models (Click Model, Term Model and Semantic Model) are proposed in this section. Each model is defined based on the former one sequentially, and all these three models think of queries as documents, where components in the documents are URLs, terms and intents, respectively. Firstly, Click Model considers the number of clicks from URLs to queries and the description power of each URL to smooth query-URL bipartite graphs. This model can be employed in other bipartite graphs, such as user-item graphs, traffic network graphs (easily modeled as bipartite graphs). Secondly, Term Model takes advantages of content of clicked web pages by click behaviors analysis and defines an expectation distribution

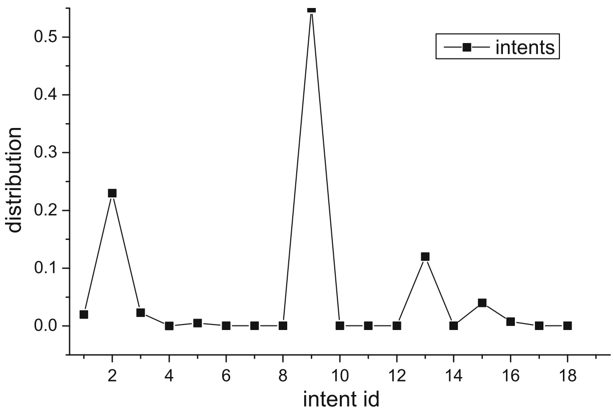


Figure 3 An example of Semantic Model. "The Chinese Paladin 2" is the name of a famous Role Play Games in China, the name of a television series and the name of a bestselling novel as well, where we can see three main intents (id = 2, 9, 13) in the line chart

over terms to describe queries. Thirdly, Semantic Model inferred from LDA maps ambiguous queries into a distinguishing feature (intent) space.

Details on similar queries finding based on query representation models are discussed in the next section.

4 Finding similar queries

Query representation models can be helpful for many IR applications, such as similar query finding, query suggestion and query reformulation, etc. With distribution representation mentioned above (URL transition probability distribution, term distribution and semantic intent distribution), we can apply common measurements to find similar queries. In this paper, two kinds of metrics are chosen: pairwise similarity and graph-based random walk measures.

4.1 Pairwise similarity measure

As queries are represented by a vector of URLs, terms or intents, two vector-based similarity metrics are chosen: Cosine similarity and Jaccard coefficient.

The Cosine similarity is a measure of similarity between two vectors by measuring the cosine of the angle between them in space. The bigger the value of Cosine similarity is, the more similar two queries are. The similarity between q_i and q_j is defined as:

$$Cos\left(q_i, q_j\right) = \frac{\mathbf{q_i} \cdot \mathbf{q_j}}{\parallel \mathbf{q_i} \parallel \parallel \mathbf{q_j} \parallel} \tag{9}$$

The Jaccard coefficient is widely used in similar queries finding [1, 10]. The metric is defined to measure similarity and dispersion between two sets. The Jaccard coefficient is the value of the intersection divided by the value of union of the vectors:

$$Jac\left(q_i, q_j\right) = \frac{\mathbf{q_i} \cdot \mathbf{q_j}}{\parallel \mathbf{q_i} \parallel^2 + \parallel \mathbf{q_j} \parallel^2 - \parallel \mathbf{q_i} \parallel \parallel \mathbf{q_j} \parallel} \tag{10}$$

In both metrics, $\mathbf{q_i}$ and $\mathbf{q_j}$ denotes the representation models of query q_i and query q_j.

4.2 Graph-based random walk measure

For graph-based measures, the similarity between queries is defined on query graphs [5, 6]. A typical query graph is the query transition graph, which is a directed graph $G = (Q, E, P^{qq})$, where Q is the set of queries, $E = \{(q_i, q_j)\}$ is the set of edges, and P^{qq} is the transition probability matrix between queries, where $p_{ij}^{qq} = p(q_j|q_i)$ is the transition probability from q_i to q_j. In our models, Q^U, Q^T and Q^S are all representation matrixes for query representation. The rows of these matrixes are representation for each query. The columns of these matrixes are items used for representing queries. They are URLs in URL Model, terms in Term Model and intents in Semantic Model, respectively. We normalize these matrixes by row and get transition probability from a query to each item, denoted as the transition matrix P^{q2i}. Analogously, we normalize these matrixes by column to get transition probability from a item to each query, denoted as the transition matrix P^{i2q}. So the transition probability from q_i to q_j is defined as:

$$p\left(q_j|q_i\right) = \sum_k p_{jk}^{i2q} p_{ki}^{q2i}, \tag{11}$$

where k iterates over each item in the representation models, p_{ki}^{q2i} is the transition probability from query i to item k and p_{jk}^{i2q} is the transition probability from item k to query j.

The personalized pagerank [16] is usually used to rank vertices by random walk on the graph in a query dependent way. The rank strategy is showed in (12):

$$R_i^{n+1} = (1-\alpha) R_i^0 + \alpha \cdot \sum_j p\left(q_i|q_j\right) R_j^n, \tag{12}$$

where R_i^0 is a personalized initial values for vertex i (the vertex i means query q_i on the graph), and n is the steps of random walk. We set $R_i^0 = 1$ if vertex i is the given query and 0 for others. The parameter α is set to 0.7 as the previous study [10].

In the end, we provide a ranked candidate queries on the top of the similarity score list. In our experiments, we judge the performance from the view of list-based. We compare representation models with each other, and the traditional similar queries finding based on terms of queries is our baseline representation method (see Section 5.3). The experimental results of similar query finding by pairwise similarity metrics is shown in Section 5.3, and the evaluations of query recommendation by personalized pagerank in Section 5.4.

5 Experiments

In this section, we conduct experiments to verify the effectiveness of our representation models for similar query finding on a Chinese click-through data set from a Chinese search engine.

The task focuses on finding the most semantic similar queries for the seed query based on click graphs. In the rest of this section, firstly, we introduce our data set, the assessments and evaluation metrics. Secondly, we compare three representation models (Click Model, Term Model, Semantic Model) with traditional Original Model

Table 4 Examples in our click-through data (explanations in parentheses)

Query	Title of the URL	URL
sina	xin lang shou ye (*sina home page*)	http://www.sina.com.cn/
hui sheng hui ying	hui sheng hui ying X4 zhuan ye shi pin bian ji ruan jian (*a professional video editing software called Ulead VideoStudio*)	http://www.corel.com/servlet/Satellite/cn/cs/Product/1175714228541
da wei bei ke han mu	da wei bei ke han mu–bai du bai ke (*David Beckham in Baidu's Encyclopedia*)	http://baike.baidu.com/view/2298.htm

(introduced in Section 5.3). Thirdly, this section also includes a detailed analysis of effectiveness on different frequency queries. Finally we show some examples of similar queries. Because our queries are in Chinese, we show them in Chinese Pinyin with explanations in parentheses throughout this article.

5.1 Data collection and analysis

We obtain a click-through data set randomly sampled from a widely used Chinese search engine, Baidu,[1] during three months in 2010. These samples contain about 108 million records. We show some examples of records in Table 4. In the table, queries are submitted by users. The titles in the data come from the web page of users' clicks, such as "xin lang shou ye (*sina home page*)" is the title of "http://www.sina.com.cn/".

The data is cleaned and trimmed as follows. Firstly, we only keep well-formatted, English or Chinese queries (queries which only contain English or Chinese characters and space). Secondly, we segment queries into terms (we use ICTCLAS tools,[2] because Chinese queries don't have blank spaces between each term.) to segment queries in this paper and remove stop words. Thirdly, we filter out queries and corresponding click-through data containing adult content, privacy information and market information, so these records are not taken into account in our study. Fourthly, the query-URL pairs that have appeared at least 5 times, are retained. By the means above, we reduce possible noises and keep generality as well. Finally, we get 190,614 queries, 175,141 URLs and 117,894 terms. As usual, a general query contains 5.06 characters on average and 3.04 terms after segmentation. A total of 1,763,335 edges are observed on our query-URL bipartite graph, which indicates that each query has 4.1 distinct URLs, and each URL is clicked by 1.58 different queries. The times of query submission have been averaging out 1,804 times (varying from 5,925,555 times to 15 times) and the average number of clicks for each URL is 239 times (varying from 683,341 times to 5 times). Therefore, our data set contains various kinds of queries with different frequencies and different amount of corresponding clicked URLs.

[1]www.baidu.com

[2]http://www.ictclas.org/

5.2 Experimental data and settings

In our experiments, we select queries that have connected more than 3 distinct URLs to construct our three representation models, and then 16,531 queries are chosen in total. With different representation models, we apply two pairwise measures to find similar queries and apply one graph-based measure mentioned in Section 4 for query recommendation

We take $tf \cdot idf$ weighted word vectors of original queries (called Original Model, denoted as Q^O) to represent queries as the baseline of representation models, and then apply the same similarity measures on vectors to find similar queries. In Click Model, we set $k_1 = 2.0$ and $b = 0.75$ as default [28]. Click Model extracted from the bipartite graph is calculated off line. Term Model obtains terms in titles that are segmented in the same way of query segmentation. Regarding Semantic Model, we set the number K of topics as 18 learnt from the results of clustering (mentioned in Section 3.4), where the parameters of DBSCAN are set as default: epsilon to 0.9, minpoint to 6. The parameters of LDA are: α to 0.3, β to 0.1. For Gibbs Sampling, we find that when the sampling time is reasonable large (i.e. $\geq$ 10,000), the performance is comparatively stable.

We randomly collect 90 seed queries from 3 groups which are classified by their submission frequency (popular queries, normal queries and long-tail queries). We choose 30 queries from each group. The details are shown in Table 5 (the average number of submissions and the average number of different URLs connected with them, denoted as ANS and ANU, respectively).

For each seed query, we generate similar candidates via our models (we output no more than 10 candidates with nonzero similarity to be evaluated). Evaluating the quality of semantic similarity between queries is difficult, in particular for these UGC (User Generated Content). In this paper, we ask five experts for judging work. For finding similar queries, human judgers' work is to rate similarity candidates for seed queries, and for query recommendation ,their work is to click the recommended queries. Before rating, we show experts top ten search results of the seed query returned by search engines for understanding the original one. We define 3 levels (0, 1 and 2) to measure the relevance between the seed queries and the candidates, in which 0 means "totally irrelevant", 1 means "weak related or similar with the seed query" and 2 indicates "entirely relevant". The final similarity judges of query pairs use majority vote to decide. In this way, we obtained a similar query set with total 5,032 distinct queries and 6,710 pairs to be judged. In our experimental data, 1,106

Table 5 Snapshots for seed queries

Group	ANS	ANU	Examples
q-tail	15.5	3	Lenka (*a singer*), ao wei si (*a parttime job site called Ao Wei Si*), duo pu da shou ji lun tan (*Dopod smart phone Forum*)
q-norm	23165	4	HeFei lun tan (*HeFei Forum*), uuu9 (*a popular game portal in China*), kuai bo (*a video player program called QvodPlayer*)
q-pop	35118.7	38.77	Zhong guo yi dong (*China Mobile*), qian cheng wu you (*a job-hunting site called future*), nan fei shi jie bei kai mu (*2010 FIFA World Cup opening ceremony*)

pairs are mostly similar (labeled 2, 16.5 % in total), 1369 pairs are similar (labeled 1, 20.4 % in total), and 4,235 pairs are totally irrelevant (63.1 % in total).

For the task of query recommendation, we show the judgers seed queries and mixed recommendations from four models to ensure that they don't know which model recommendations come from. We define 3 levels (0, 1 and 2) to measure the satisfaction of recommendations. 0 means that nobody has clicked on this recommended query, 1 means that 1 to 3 persons have clicked on it and 2 means that more than 3 persons have clicked on this recommended query. After clicking, we get 5,313 recommended query pairs for all seed queries. In our evaluation data, 2,332 pairs are labeled 2 (43.9 % in total), 1,186 pairs are labeled 1 (22.3 % in total), and the rest are labeled 0 (33.78 % in total).

5.3 Evaluation on similar queries finding

In the evaluation part, we compare our representation models (Click Model as Q^C, Term Model as Q^T, Semantic Model as Q^S) with the Original Model (Q^O, details in Section). We evaluate the effectiveness of our models by Coverage Ratio (CR), precision ($P@n$), Mean Average Precision (MAP) and show examples for them. The following sections abbreviate "Similar Queries Finding" to *SQF*.

5.3.1 Evaluation for SQF on coverage ratio

Coverage Ratio (CR) means the average number of similar queries output from the model divided by the default output number. The Single Coverage Ratio (SCR) for each query is defined as

$$SCR\,(q_i) = \frac{r\,(q_i)}{R}, \tag{13}$$

where $r\,(q_i)$ is the number of candidates with nonzero similarity as the models output for query q_i, and R is the default output number of similar queries (here is 10). The CR is defined on the whole test set:

$$CR = \frac{\sum_{q_i \in SQ} SCR\,(q_i)}{|SQ|}, \tag{14}$$

where SQ means the set of seed queries and $|SQ|$ means the size of the set. The results are showed in Table 6:

In Table 6, we report comparative results of CR on different models and consider whether our methods can boost the performance when using representation models. We find that the best CR is 0.996 of Semantic Model with Cosine similarity. As expected, our proposed methods outperform the baseline model, at least have 14.2 % up (Click Model to Original Model on Cosine similarity), at most have increased 24.4 % (Semantic Model to Original Model on Jaccard coefficient). We can see that

Table 6 Comparison between different representation models on CR

Method	Q^O	Q^C	Q^T	Q^S
Cosine similarity	0.753	0.894	0.994	**0.996**
Jaccard coefficient	0.419	0.561	0.661	**0.663**

Bold entries are the best results

Cosine similarity is better for *SQF* on CR, when we use vector space model to represent queries. CR increases greatly from Original Model, Term Model, Click Model to Semantic Model. Term Model outperforms Click Model and has a sharp increase. The reason is that Term Model shares two key points. One is that it has redefined the description power of each query-URL edge besides the original weight, and the other one is that the model use probability statistical distribution of term expectation to represent a query appropriately. Moreover, Semantic Model gets a higher CR than others. Because in Semantic Model, queries are represented as distinct intents in a high-dimension space, such that topics here are coarse-grained categories for users' intents, such as movies, games, digital products and so on. In Section 5.3.3, we will have a detailed discuss about the similar queries found via Semantic Model.

5.3.2 Evaluation for SQF on accuracy

Two evaluation metrics, *P@N* and *MAP* are employed to measure the accuracy of *SQF*. *P@N* shows the percentage of similar queries at the top N of the similar query list. Due to reasons that *SQF* is still a ranking question, we also use a position considered metric, *MAP*, to analysis. MAP here is defined as follows:

$$MAP = \frac{\sum_k avgP_k}{|SQ_{SCR(q)>0}|}, \tag{15}$$

where $|SQ_{SCR(q)>0}|$ means the number of seed queries which has at least one similar query, $avgP_k$ is the average precision of similar queries when current query k is posted. It is defined as:

$$avgP_k = \sum_{j=1}^{C(q_k)} \frac{p\,(j) \cdot l\,(j)}{C\,(q_k)}, \tag{16}$$

where $p\,(j)$ is the accuracy of the jth similar query, $l\,(j)$ is label information. $C\,(q_k)$ is the number of similar queries to the seed query q_k. MAP reflects the accuracy of *SQF* and evaluates the global effectiveness of query representation models.

We look into the accuracy on whole test queries at first, and then display performance on queries with high, middle and low frequency using different models.

In the Table 7, there is 6 to 7 % increase from Original Model to Click Model. It verifies that Click Model is more effective on the task of similar queries finding. We learn that Click Model improves over Original Model by up to 9.75 % $((0.822 - 0.749)/0.749 = 0.0975)$ on Cosine similarity and 7.41 % $((0.797 - 0.742)/0.742 = 0.0741)$ on Jaccard similarity. Term Model gets higher performance over other models including Semantic Model which is constructed based on it. So we can infer that understanding search intent can't ignore the fact that queries are mainly made of terms and semantic understanding can not be separated from terms. We also learn

Table 7 Accuracy for different models

Method	Q^O	Q^C	Q^T	Q^S
Jaccard coefficient	0.749	0.822	**0.845**	0.570
Cosine similarity	0.742	0.797	**0.840**	0.590

Bold entries are the best results

that it is unsuited to directly apply coarser-grained topic-level description (term-level description is finer-grained) on the task of similar query finding. Even though they are both on the semantic level, the similar queries found by Semantic Model may easily go far from the intrinsic queries. For example, “China Mobile” and ‘China Telecom” are both service providers. But the two queries are referred to different companies. Semantic Model learns the intent distribution of the two which are remarkably similar to each other, because they provide the same service and both do the network construction. Thus, Semantic Model provides us the view of queries from the semantic categories. That is to say, Semantic Model can find the queries under similar category distribution, which provides us with more class information of the original queries. So, when users don’t clear about their intents and only have keyword queries, Semantic Model is a good choose for query recommendation.

In Figure 4, we show the comparison of different models using $P@n$. In Figure 4b, Original Model drops in performance sharply. Click Model, Term Model and Semantic Model change slowly and are more stable in position, especially Semantic Model. According to the experimental results, considering term expectation distribution over queries (Term Model) is very essential for query representation. It can effectively manage and organize the main intents of queries.

Next, we conduct a detailed analysis about performance of models on different frequency queries.

Performance of popular, normal, long-tail queries on MAP shows in Figure 5. In our experimental data, popular queries are mainly about current buzz (such as college entrance examination, real estate) or integrated information web sites. Original Model, Click Model and Term Model all have good performance on both pairwise similarity metrics (Jaccard coefficient and Cosine similarity).

For normal queries, which are mainly related to social networks or navigation web sites for a single purpose (software finding or image search), Term Model outperforms the other models. An interesting thing is that, Click Model and Term Model can successfully boost similar query finding for long-tail queries, which compose most of queries in search logs. That is to say, Click Model and Term Model have a strong modeling skill on representing queries, especially for long-tail queries.

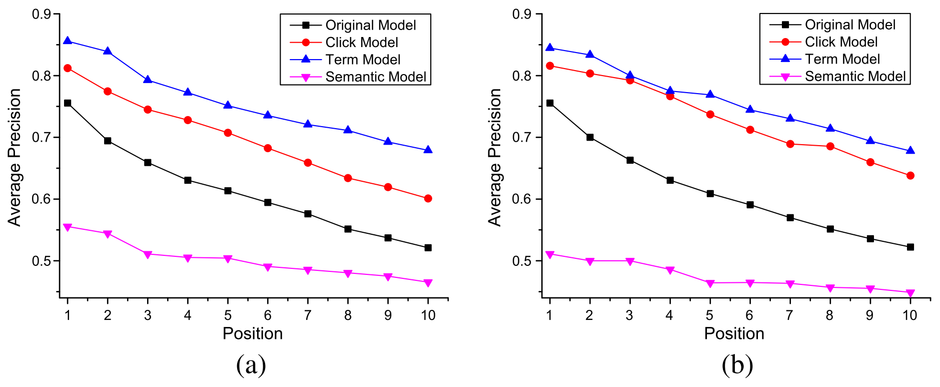
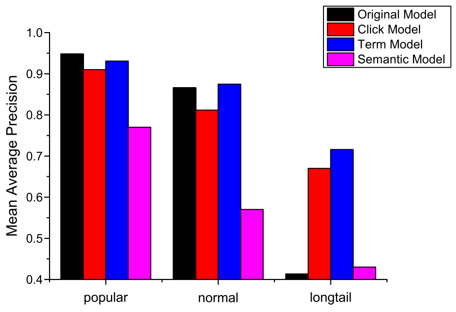
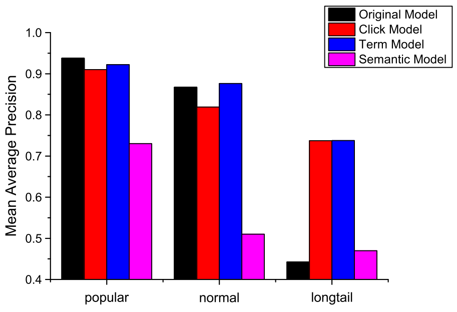


Figure 4 The performance comparison of different models on P@n. **a** Cosine similarity, **b** Jaccard coefficient



(a)



(b)

Figure 5 The performance of popular, normal, long-tail queries on MAP. **a** Cosine similarity, **b** Jaccard coefficient

For popular queries, users often can write words correctly and precisely, so the traditional model has a good performance. For normal queries, Term Model is better than the traditional one. For long tail queries, click information and term expectation are effective for modeling information need. Owing to the fact that traditional model has a low CR, which means our models can find similar queries for more queries, models proposed in this paper are much better on the whole.

5.3.3 The examples for similar queries

In this part, we present some examples of similar queries in Tables 8, 9 and 10. And we pursue discussions over the results.

For *long tail queries* (shown in Table 8), we show two queries. One is a query that searches for a part-time job institution "ao wei si", and the other one is a query searching for a singer "Lenka Kripac". The two queries connect 3 URLs with total 15 clicks, respectively. "ao wei si" and "Lenka" are both names of entities. In our data set, we find that there are no expanded queries for these two queries.

Table 8 The top 5 results of long tail queries: 1. ao wei si (*a part time job site*), 2. Lenka (*a singer*)

Model	ao wei si	Lenka
Q^O	wang wei da shi (*a software*)	None
	wang si yi (*a super star*)	
	wei tang (*a software*)	
	jiang wei (*a General*)	
	ya si (*IELTS*)	
Q^C	None	None
Q^T	1010 jian zhi wang (*1010 part-time site*)	xia zai ge qu (*music download*)
	jian zhi wang zhan (*part-time web site*)	ge qu xia zai mian fei (*songs free download*)
	jian zhi lun tan (*part-time forum*)	shi pin ge qu xia zai (*music download*)
	jian zhi wang (*part-time web site*)	yin yue xia zai mp3 (*music download mp3*)
	jian zhi zhao pin (*part-time recruitment*)	mp4 ge qu xia zai (*mp4 songs download*)
Q^S	zhao pin wang zhan (*recruitment web site*)	lao ge jing xuan (*selections of old melodies*)
	zhao gong zuo wang zhan (*Job search sites*)	fan chang ge qu (*covers*)
	zheng zhou ren cai (*Zhengzhou Talent*)	hao ge tui jian (*music recommendation*)
	zhi lian zhao pin (*zhaopin.com*)	zai xian ting ge (*music online*)
	ren shi xin xi wang (*personnel site*)	jing dian lao ge qu (*classic oldies*)

Table 9 The top 5 results of normal queries: 1. Hefei lun tan (*Hefei Forum*), 2. uuu9 (*a game site*)

Model	Hefei lun tan	uuu9
Q^O	Hefei tian qi yu bao yi zhou (*Hefei weather forecast in a week*)	uuu9.com mo shou di tu xia zai (*map of Warcraft download*)
	Hefei fang di chan (*Hefei real estate*)	cctv9 (*China Central Television 9*)
	Hefei di tu (*Hefei maps*)	ie9 (*a browser*)
	qq lun tan (*QQ Forum*)	u9 wang (*a game site*)
	shi jie bei lun tan (*the World Cup Forum*)	9 ku ying yue wang (*a music site*)
Q^C	Hefei (*the capital of Anhui Province*)	u9 mou shou (*u9 Warcraft*)
		mou shou rpg di tu (*Warcraft rpg map*)
		mou shou zheng ba guan wang (*Warcraft official website*)
		war3
		dota shi pin (*competition video of dota*)
Q^T	Hefei gong ye da xue (*Hefei University of Technology*)	u9 mo shou (*u9 Warcraft*)
	Hefei tian qi yu bao yi zhou (*Hefei weather forecast in a week*)	dota guan wang (*dota official website*)
	Hefei fang di chan jiao yi wang (*Hefei real estate trading web sites*)	uuu9.com mou shou di tu xia zai (*map of Warcraft download on uuu9*)
	Hefei (*the capital of Anhui Province*)	mou shou rpg di tu (*Warcraft rpg map*)
	Hefei di tu (*Hefei Map*)	dota 6.67c (*a version of dota*)
Q^S	XiCi Hutong (*Xici–the No.1 portal of Chinese community*)	mou shou zheng ba di tu (*map of Warcraft*)
	Hefei (*the capital of Anhui Province*)	mou shou di tu xia zai (*map of Warcraft download*)
	shang du wang (*a news portal of Shangdu.com*)	dota guan wang (*dota official web sites*)
	xin xi wang (*information sites*)	mou shou fang shou di tu (*Warcraft maps in defense mode*)
	jin ti wang (*community service web sites*)	rpg di tu xia zai (*Warcraft rpg map download*)

So, traditional method shows us some queries with several same characters but no semantic similarity. Due to the fact that they share no same URLs with others, Click Model is invalid to find similar queries. This is a universal situation for long-tail queries. According to statistics, the number of URLs in our data is 175,141 and the number of queries is 16,531. The average number of edges observed for each query is 4.1, and for each URL is 1.58. That means the number of queries which share at least one URL with a unique query is only $4.1 \cdot (1.58 - 1) = 2.38$ on average. So it

Table 10 The top 5 results of popular queries: 1. qian cheng wu you (*a job-hunting site called future*), 2. yi che wang (*a famous portal site about cars*)

Model	qian cheng wu you	yi che wang
Q^O	qian cheng wu you zhao pin wang (*Future, A job-hunting website*)	wang yi (*NetEase*)
	wu you wang (*a job-hunting site called Wuyou wang*)	yi wang (*EasyNet*)
	wu ren jia shi (*unmanned*)	wang yi you xiang (*Netease E-mail*)
	san guo wu shuang (*a YOKA game*)	wang yi xin wen (*Netease News*)
	zhan guo wu shuang 2 xiazai (*a game of "zhan guo wu shuang" download*)	wang yi you xi (*Netease Game*)
Q^C	51job (*Another web site for job search*)	biao zhi (*PEUGEOT car*)
	qian cheng wu you zhao pin wang (*a job-hunting web site called future*)	qi che bao jia wang (*car prices web sites*)
	51job zhao pin wang (*51job*)	qq che bao jia (*an Internet price quote for QQ car*)
	wu you wang (*a job-hunting site called Wuyou wang*)	
	zhao pin wang zhan (*Job web sites*)	
Q^T	51job.com	qi che bao jia wang (*car prices web sites*)
	qian cheng wu you zhao pin wang (*a job-hunting site called future*)	qi che lun tan (*cars forum*)
	zhi lian zhao pin (*Zhaopin.com, a job search web site*)	qi che bao jia ji tu pian (*pictures and price quote for cars*)
	zhao gong zuo (*Finding a job*)	qi che xiao you xi (*car games*)
	zhao pin wang zhan (*job-hunting web sites*)	qi che yin yue (*car audio*)
Q^S	zhao gong zuo wang zhan (*Job websites*)	qi che bao jia ji tu pian (*pictures and price quote for cars*)
	wu han zhao pin (*Wuhan recruitation*)	xin ao tuo bao jia ji tu pian (*pictures and price quote for cars*)
	shen zhen ren cai wang (*Shenzhen Talent*)	da zhong qi che bao jia ji tu pian (*pictures and price quote for cars*)
	Suzhou yuan qu ren cai xin gan xian (*Suzhou Talent Network*)	zhong hua qi che bao jia ji tu pian (*pictures and price quote for cars*)
	Qing dao she bao wang (*The web site of Qingdao social security administration*)	chang cheng qi che bao jia ji tu pian (*pictures and price quote for cars*)

is common for long-tail queries share no same clicked URLs with others. Based on Click Model, the two semantic models (Term Model and Semantic Model) can find out some queries about part-time jobs. Take "Lenka" for example, the two models can lead users to search for more music (mp4 or mp3) about this singer.

For *normal queries* (shown in Table 9), we show two queries: "Hefei lun tan (*Hefei Forum*)" and "uuu9 (*a game site*)". For query "Hefei Forum", users probably want to know information about Hefei, the capital of Anhui Province in Eastern China. Original Model can get some related queries, but also get ones without semantic relations, such as "qq Forum" and "the World Cup Forum". Based on clicks, only "Hefei" has nonzero similarity with "Hefei lun tan". It indicates the situation mentioned in Term Model (in Section 3.3): the same or related information about Hefei occurs on different sites, but users prefer different web sites. So Click Model has a lower performance than other two models. After modeling term expectation, "Hefei real estate", "Hefei University of Industry" and "Hefei weather forecast" have been found out.

In Semantic Model, we provide similar queries with the intent of "forums". Another query "uuu9" is about a famous game site.[3] A large number of users accessing this site like playing "Warcraft". Through our Click Model, "Warcraft official web site" and "competition video of dota" can be helpful to lead most of users to a destination. Term Model and Semantic Model also outperform the traditional method and provide many related products: "dota official web site", "New version of dota" and so on.

For *popular queries* (shown in Table 10), we show "qian cheng wu you (*a job-hunting site called future*)" and "yi che wang (*a famous portal site about cars*)". Owing to the same word "wu", "qian cheng wu you" in Original Model resembles irrelevant "san guo wu shuang". In Term Model and Semantic Model, another relevant job web site, "Zhaopin.com", and even some social insurance web sites have been found(in the line of Semantic Model). For "yi che wang", Click Model captures the primary intent of it and provides some similar queries with search intent of finding car price information online. In Term Model, we maintain click and semantic information via term expectation distribution. Hence, similar queries that share close semantic relation but have less co-click relation can be found, such as forums, pictures and audio about cars. Semantic Model gives us queries about popular car brands in China and the main intent underlying this query (finding the pictures and prices about cars).

5.4 Evaluation on query recommendation

In this subsection, we apply our approach on query recommendation via the personalized pagerank with Original Model, Click Model, Term Model and Semantic Model. The seed queries are the same ones as the queries used in the task of SQF. For better understanding our models, we generate a recommendation list for the given query with different number of steps (from 2 to 14). We evaluate the effectiveness of our models on query recommendation by CR, $P@n$ and MAP. The following parts abbreviate "Query Recommendation" to QR.

5.4.1 Evaluation for QR on coverage ratio

The Coverage Ratio (CR) is similar to the definition in Section 5.3.1. The difference is that $r(q_i)$ here means the number of recommendations for query q_i with nonzero similarity and R is the default output number of recommended queries (here is 10). The Figure 6 illustrates the CR of four models for different steps. The CR of four models all become stable after 3 steps. Obviously, Term Model and Click Model can output more recommendations than Original Model and Semantic Model.

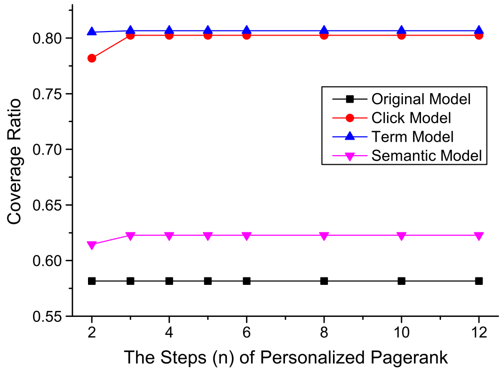
5.4.2 Evaluation for QR on accuracy

Two evaluation metrics are employed to measure the accuracy of QR: Precision@n ($P@n$) and Mean Average Precision (MAP). $P@n$ and MAP are defined as description in Section 5.3.2. The different here is that we use recommendation labels to replace similarity labels. First, we evaluate global performance on all test queries. Second, we show the performance of different frequency queries.

The Figure 7a illustrates that MAP of models converges very quickly with the increasing of steps. All models improve their performance slightly with the increase

[3]http://www.uuu9.com/

Figure 6 The performance of personalized pagerank models on Coverage Ratio



of the number of steps, and they achieve the best performance when n equals 10. As shown in the figure, it is clear that Click Model, Term Model and Semantic Model outperform the Original Model in every step. We also show the performance of different frequency queries on MAP in Figure 7b. It reveals that our models get a higher user satisfaction in three kinds of frequency queries than the baseline model. Click Model improves over Original Model by more than 30 %. The results also re-confirm that click graphs can better capture semantic relations between queries than original terms of queries do. Term Model suits the task of query recommendation better than the other models on both popular and normal queries. Because Term Model considers users' actual information need through understanding historical click behaviors and represents a query with its main purpose and entities appropriately. For long-tail queries, as we discuss in Section 5.3.3, Semantic Model improves over Original Model by up to 39.64 %, over Term Model by up to 14.65 % , over Click Model by up to 3.37 %. It confirms that Semantic Model helps for query understanding from the user's perspective via modeling queries as distributions over intents.

We also compare our models with Original Model to see the improvement trend of performance. Table 11 further depicts the precision scores at position one, position three and position seven for each method. Our models beat the baseline, but three models have different performance on different kinds of queries. They suffer from different changed trend cross the positions. For popular queries, Click Model and

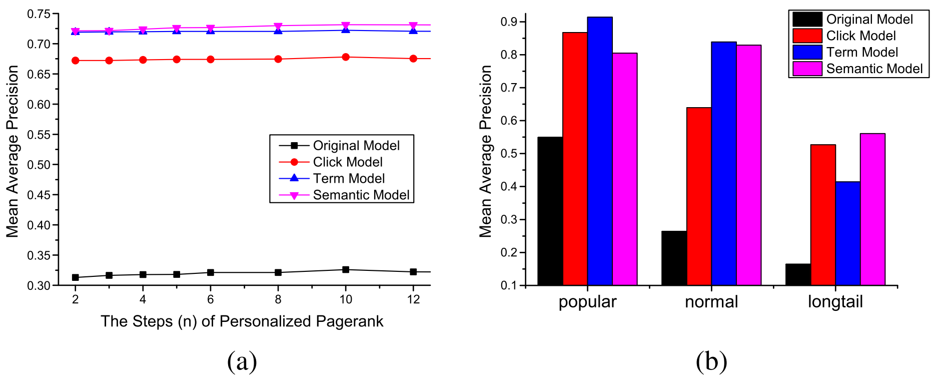


Figure 7 The performance of models on MAP. **a** MAP on all test queries, **b** MAP on different frequency queries (in Step 10)

Table 11 The comparison of different methods on different frequency queries by P@1, P@3 and P@7

Method	Popular			Normal			Long-tail		
	P@1	P@3	P@7	P@1	P@3	P@7	P@1	P@3	P@7
Q^O	0.514	0.489	0.434	0.233	0.193	0.160	0.201	0.192	0.160
Q^C	0.772	0.744	0.734	**0.836**	0.789	0.744	**0.533**	0.445	0.404
Q^T	**0.934**	**0.891**	**0.887**	0.834	**0.817**	**0.757**	0.433	0.394	0.395
Q^S	0.867	0.859	0.847	0.667	0.561	0.590	0.513	**0.510**	**0.426**

Bold entries are the best results

Semantic Model have more steady performance than Term Model. While, Term Model improves the performance of Click Model a lot and beats Semantic Model on popular queries. For normal queries, Click Model gets the best predictions on $P@1$, but it decreases more quickly than Term Model after the first position. For long-tail queries, Click Model can suggest better queries in the first position than the others. But due to the reason that long-tail queries display more personal or special information need, it's better to understand queries on topic level when recommending. From the results in Table 11, it confirms that Click Model is applicable in adjusting weights of click graphs, which is useful to model the relationships between two kinds of nodes in bipartite graphs.

5.5 Discussions

Based on the above experimental analysis, we verify our models have three advantages:

- URL Model has a strong capacity to represent users' actual intent underlying queries. Query representation based on query-URL bipartite graphs takes users' judgment and preference into query modeling, which gives an effective distribution description of queries.
- Term Model and Semantic Model both give queries a semantic explanation and description. Term Model (a fine-grained representation) proposes a novel term expectation to represent queries, in which content of web pages and the relationship between query and URLs have been considered carefully. Semantic Model (a coarse-grained representation) maps a query into a high dimension space to resolute queries into topics. The two representation models both combine behavior information and content information seamlessly than before.
- Our approach has a good performance for the task of similar queries finding, especially for normal queries and long-tail queries. We find similar queries to help users reformulate their queries and make users' information need clear.

6 Conclusion

In this paper, we proposed a two-stage framework to find semantic similar queries with query representation models. We evaluated three query representation models. Unlike previous methods, we provided a probability distribution over URLs, an expectation distribution over terms and a probability distribution over intents for query representation. Based on analysis and modeling, pairwise similarity metrics and graph-based similarity metrics were applied on these models to compute similarity of two irregular and short queries. Furthermore, our methods can also be used in many fields (sponsored search, image retrieval, Q&A and so on). The work described in this paper has raised several interesting questions and opens up several lines of exciting research work in the future. As we know, a search process is a sequence of user actions by submitting queries and clicking URLs. It is not a simple collection of keywords. So learning users' information need via sequence modeling will also be a reasonable and interesting work.

Acknowledgement This research is supported by the National Natural Science Foundation of China under Grant No. 61105049.

References

1. Beeferman, D., Berger, A.: Agglomerative clustering of a search engine query log. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '00, pp. 407–416. ACM, New York, NY (2000). doi:10.1145/347090.347176
2. Bendersky, M., Croft, W.B.: Modeling higher-order term dependencies in information retrieval using query hypergraphs. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12, pp. 941–950. ACM, New York, NY (2012). doi:10.1145/2348283.2348408
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
4. Boldi, P., Bonchi, F., Castillo, C., Donato, D., Gionis, A., Vigna, S.: The query-flow graph: model and applications. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08, pp. 609–618. ACM, New York, NY (2008). doi:10.1145/1458082.1458163
5. Boldi, P., Bonchi, F., Castillo, C., Donato, D., Vigna, S.: Query suggestions using query-flow graphs. In: Proceedings of the 2009 Workshop on Web Search Click Data, WSCD '09, pp. 56–63. ACM, New York, NY (2009). doi:10.1145/1507509.1507518
6. Bordino, I., Castillo, C., Donato, D., Gionis, A.: Query similarity by projecting the query-flow graph. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '10, pp. 515–522. ACM, New York, NY (2010). doi:10.1145/1835449.1835536
7. Castillo, J.J.: A wordnet-based semantic approach to textual entailment and cross-lingual textual entailment. IJMLC **2**(3), 177–189 (2011). doi:10.1007/s13042-011-0026-z
8. Chen, J., Wang, Y., Liu, J., Huang, Y.: Modeling semantic and behavioral relations for query suggestion. In: Web-Age Information Management, pp. 678–690. Springer (2013)
9. 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, SIGIR '07, pp. 239–246. ACM, New York, NY (2007). doi:10.1145/1277741.1277784
10. Deng, H., King, I., Lyu, M.R.: Entropy-biased models for query representation on the click graph. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09, pp. 339–346. ACM, New York, NY (2009). doi:10.1145/1571941.1572001
11. Dou, Z., Hu, S., Luo, Y., Song, R., Wen, J.R.: Finding dimensions for queries. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM '11, pp. 1311–1320. ACM, New York, NY (2011). doi:10.1145/2063576.2063767
12. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), pp. 226–231. Portland, Oregon, USA. AAAI Press 1996. ISBN 1-57735-004-9 (1996)
13. Fujita, S., Dupret, G., Baeza-Yates, R.A.: Learning to rank query recommendations by semantic similarities. CoRR. arXiv:abs/1204.2712 (2012)
14. Griffiths, T.: Gibbs sampling in the generative model of Latent Dirichlet Allocation. Tech. rep., Stanford University (2002). www-psych.stanford.edu/∼gruffydd/cogsci02/lda.ps
15. Guo, J., Cheng, X., Xu, G., Zhu, X.: Intent-aware query similarity. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM '11, pp. 259–268. ACM, New York, NY (2011). doi:10.1145/2063576.2063619

16. Haveliwala, T., Kamvar, S., Jeh, G.: An analytical comparison of approaches to personalizing pagerank. Technical Report 2003-35, Stanford InfoLab (2003)
17. Hu, Y., Qian, Y., Li, H., Jiang, D., Pei, J., Zheng, Q.: Mining query subtopics from search log data. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12, pp. 305–314. ACM, New York, NY (2012). doi:10.1145/2348283.2348327
18. Huang, C.K., Chien, L.F., Oyang, Y.J.: Relevant term suggestion in interactive web search based on contextual information in query session logs. JASIST **54**(7), 638–649 (2003). doi:10.1002/asi.10256
19. Huang, J., Gao, J., Miao, J., Li, X., Wang, K., Behr, F., Giles, C.L.: Exploring web scale language models for search query processing. In: Proceedings of the 19th International Conference on World Wide Web, WWW '10, pp. 451–460. ACM, New York, NY (2010). doi:10.1145/1772690.1772737
20. Ji-Rong, W., Jian-Yun, N., Zhang, H.J.: Query clustering using user logs. ACM Trans. Inf. Syst. **20**(1), 59–81 (2002). doi:10.1145/503104.503108
21. Jones, R., Rey, B., Madani, O., Greiner, W.: Generating query substitutions. In: Proceedings of the 15th International Conference on World Wide Web, WWW '06, pp. 387–396. ACM, New York, NY (2006). doi:10.1145/1135777.1135835
22. Liu, Y., Miao, J., Zhang, M., Ma, S., Ru, L.: How do users describe their information need: query recommendation based on snippet click model. Expert Syst. Appl. **38**(11), 13,847–13,856 (2011). doi:10.1016/j.eswa.2011.04.188
23. Ma, H., Yang, H., King, I., Lyu, M.R.: Learning latent semantic relations from clickthrough data for query suggestion. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08, pp. 709–718. ACM, New York, NY (2008). doi:10.1145/1458082.1458177
24. Mei, Q., Zhou, D., Church, K.: Query suggestion using hitting time. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08, pp. 469–478. ACM, New York, NY (2008). doi:10.1145/1458082.1458145
25. Metzler, D., Croft, W.B.: A markov random field model for term dependencies. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '05, pp. 472–479. ACM, New York, NY (2005). doi:10.1145/1076034.1076115
26. Metzler, D., Croft, W.B.: Latent concept expansion using markov random fields. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07, pp. 311–318. ACM, New York, NY (2007). doi:10.1145/1277741.1277796
27. Radlinski, F., Broder, A., Ciccolo, P., Gabrilovich, E., Josifovski, V., Riedel, L.: Optimizing relevance and revenue in ad search: a query substitution approach. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '08, pp. 403–410. ACM, New York, NY (2008). doi:10.1145/1390334.1390404
28. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M., Gatford, M.: Okapi at TREC-3. In: Proceedings of the 3rd Text REtrieval Conference, pp. 109–126. Department of Commerce, National Institute of Standards and Technology (1994)
29. Sahami, M., Heilman, T.D.: A web-based kernel function for measuring the similarity of short text snippets. In: Proceedings of the 15th International Conference on World Wide Web, WWW '06, pp. 377–386. ACM, New York, NY (2006). doi:10.1145/1135777.1135834
30. Song, Y., Zhou, D., He, L.w.: Query suggestion by constructing term-transition graphs. In: Proceedings of the 5th ACM International Conference on Web Search and Data Mining, WSDM '12, pp. 353–362. ACM, New York, NY (2012). doi:10.1145/2124295.2124339
31. Wang, H., Liang, Y., Fu, L., Xue, G.R., Yu, Y.: Efficient query expansion for advertisement search. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09, pp. 51–58. ACM, New York, NY (2009). doi:10.1145/1571941.1571953
32. Xue, X., Croft, W.B.: Generating reformulation trees for complex queries. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12, pp. 525–534. ACM, New York, NY (2012). doi:10.1145/2348283.2348355
33. Yi, X., Allan, J.: Discovering missing click-through query language information for web search. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM '11, pp. 153–162. ACM, New York, NY (2011). doi:10.1145/2063576.2063604