World Wide Web (2013) 16:273–297
DOI 10.1007/s11280-011-0151-3

An efficient approach to suggesting topically related web queries using hidden topic model

Lin Li · Guandong Xu · Zhenglu Yang · Peter Dolog · Yanchun Zhang · Masaru Kitsuregawa

Received: 31 August 2010 / Revised: 29 August 2011 /
Accepted: 23 November 2011 / Published online: 26 January 2012

Abstract Keyword-based Web search is a widely used approach for locating information on the Web. However, Web users usually suffer from the difficulties of organizing and formulating appropriate input queries due to the lack of sufficient domain knowledge, which greatly affects the search performance. An effective tool to meet the information needs of a search engine user is to suggest Web queries that are topically related to their initial inquiry. Accurately computing query-to-query similarity scores is a key to improve the quality of these suggestions. Because of the short lengths of queries, traditional pseudo-relevance or implicit-relevance based approaches expand the expression of the queries for the similarity computation. They explicitly use a search engine as a complementary source and directly extract

L. Li (✉)
School of Computer Science and Technolgoy, Wuhan University of Technology, Wuhan, Hubei China
e-mail: cathylilin@gmail.com

G. Xu · P. Dolog
Department of Computer Science, Aalborg University, Aalborg, Denmark

G. Xu
e-mail: xu@cs.aau.dk

P. Dolog
e-mail: dolog@cs.aau.dk

Z. Yang · M.Kitsuregawa
Institute of Industrial Science, The University of Tokyo, Tokyo, Japan

Z. Yang
e-mail: yangzl@tkl.iis.u-tokyo.ac.jp

M.Kitsuregawa
e-mail: kitsure@tkl.iis.u-tokyo.ac.jp

Y. Zhang
School of Engineering & Science, Victoria University, Melbourne, VIC, Australia
e-mail: yanchun.zhang@vu.edu.au

additional features (such as terms or URLs) from the top-listed or clicked search results. In this paper, we propose a novel approach by utilizing the hidden topic as an expandable feature. This has two steps. In the offline model-learning step, a hidden topic model is trained, and for each candidate query, its posterior distribution over the hidden topic space is determined to re-express the query instead of the lexical expression. In the online query suggestion step, after inferring the topic distribution for an input query in a similar way, we then calculate the similarity between candidate queries and the input query in terms of their corresponding topic distributions; and produce a suggestion list of candidate queries based on the similarity scores. Our experimental results on two real data sets show that the hidden topic based suggestion is much more efficient than the traditional term or URL based approach, and is effective in finding topically related queries for suggestion.

Keywords query suggestion · hidden topic model · latent Dirichlet allocation · Web search engine

1 Introduction

Until now various strategies have been studied to improve the quality of general Web search, such as personalized search, contextual search, collaborative search, query expansion, query suggestion and so forth. And also a number of techniques and approaches are adopted in real applications or commercial systems by these strategies, such as clustering, classification, semantics, ontology, and so on. A main problem, however, suffering users in using a search engine is that if a user is not familiar with some domains, she might fail to choose keywords appropriately and accurately for her information need, thus having a difficulty in formulating her input query. Furthermore, Web queries are usually short in length (a couple of words) [23] and ambiguous in meaning expression [12, 35]. Due to these difficulties, it is widely recognized that query formulation is a challenging obstacle in improving search quality.

Query suggestion is considered as an effective strategy in enhancing keyword based queries [25] that are widely used in search engines and Web search systems. A commonly used query suggestion method is to recommend related queries from a set of candidate queries which are submitted before and embody the collaborative opinions of previous users. Commercial search engines have implemented it using their respective techniques, such as "Related search terms" in Google, "Search Assist" in Yahoo!, and "Related Searches" in Bing Search. Current users can specify alternative related queries in their search processes in order to clarify their information needs. We categorize the research efforts on related query suggestion into the following three classes of strategies according to the feature space source used by them:

– **Query term based suggestion** is easily applicable, which intuitively considers the candidate queries having common words with a current input query to be related. However, the very small word overlap between short Web queries makes it hard to accurately determinate the similar terms. One reason is probably that a term may be meant for different meanings by different users (i.e. polysemy),

e.g., apple is related to fruit or computer. The other reason is that different terms may have a similar meaning (i.e. synonymy), e.g., car and automobile. Moreover, the common words cannot always give useful hints for users to refine their input queries. Therefore, many current studies focus on how to improve the performance of this naive suggestion strategy and find topically related queries.

- **Pseudo relevance based suggestion** induces the similarity between the two queries by enriching query expressions with additional features (e.g., terms and URLs) from the top results returned by search engines [18, 19, 28, 45]. It assumes that the meaning of a query is potentially relevant to its top search results. The pseudo relevance reflects the viewpoints of general search engines, but contains no evidence of individual users' judgments [32].
- **Implicit relevance based suggestion** extracts the additional features (e.g., terms and URLs) from the clicked search results of Web logs to calculate query-to-query similarity [1, 2, 4, 27, 31, 33, 45]. It attracts more researchers' interests than the pseudo relevance because the click choices made by a large number of users do suggest a certain level of relevance from the viewing point of actual users.

Although the query term based suggestion strategy, is the simplest and does not require any external sources, its suggestion results are always unsatisfactory. To improve the suggestion precision, the latter two strategies are widely used. At a first glance, the implicit relevance based suggestion is more practical than the pseudo relevance based one since it reflects the real opinions of web users, but it has a critical drawback. If the input query by a user is totally new to current Web logs, its click information is not accessible from the Web query logs for query similarity computation. In this case, the pseudo relevance can be used to expand the expressions of queries by features extracted from the top search results returned by search engines. Two traditional feature spaces are terms and URLs of these search results. However, there are two problems we should solve.

- In the case of a large-scale and real-time Web search, this task is particularly challenging because retrieving search results from a search engine is a time-consuming and high-cost process in the pseudo relevance based suggestion. In this paper, one of our objective is to improve the suggestion efficiency.
- If such burdens on search engines are acceptable with the technical development of server computation ability, using terms of search results obviously shows higher precision scores than using URLs. However, terms are not available at all the Web pages such as non-text pages like multimedia(image) files, sites with registration and so forth. It is crucial to improve the suggestion quality under such circumstances. The other objective of this paper is to design a general approach which is applicable no matter which kind of search results a query will retrieve.

Bearing the above two problems in mind, we propose a unified two-step approach to query suggestion in this paper. In the offline model-learning step, for gaining external knowledge from Web we collect an appropriate training dataset which is used to learn a generative probabilistic model for Web queries. Then, we employ an iterative estimation procedure to discover a latent semantic topic model, which is presumed to govern the co-occurrences of web queries. With this model, for each candidate query, its posterior distribution of hidden topics is determined. Furthermore in the online query suggestion step, the topic distribution is inferred

for an input query as well. We calculate the similarity between the input query and candidate queries based on their corresponding topic distributions. Finally, a suggestion list of candidate queries is produced based on the similarity scores, which gives the suggestions to search engine users in rephrasing their queries.

One strength of the proposed approach is that it does not require that the appearance of a whole input query in log data and the involvement of a search engine. In addition, its training step can be completed in an offline step, thus it is more applicable for large-scale real time Web applications than the traditional suggestion approaches.

We make the following contributions. First, the proposed probabilistic topic model learned from external sources allows us to represent Web queries at a latent topic level, which does not need to extract expressive feature spaces from the search results of queries. Second, experimental results show that the learned topic feature space is much more effective than the URL feature space. Moreover, it largely reduces the dimension of the term feature space (by 8,305 times in our experimental data), runs fast once the offline model-learning step is done and still produces the comparative suggestion performance against the term feature space. Third, we conduct evaluation comparisons on our proposed feature space (i.e., topic model) and the two traditional feature spaces (i.e., term and URL model). In addition, we empirically investigate the effect of the number selection of hidden topics on suggestion quality.

The rest of the paper is organized as follows. First, The related work is reviewed in Section 2. Section 3 presents a framework for our approach. Then, we briefly introduce the proposed hidden topic model in Section 4. Section 5 explains the proposed two-step approach in detail. We report results of empirical studies in Section 6. Finally, the paper is concluded and the future research work is outlined in Section 7.

2 Related work

2.1 Commercial systems

Many advanced Web searching techniques have been developed and used in commercial Web search engines to find relevant information given a keyword based query. Google, Autonomy, and Smartlogic are three successful industrial companies of Web data management. They usually combine several strategies into one system to achieve better search results, and the technical details of their systems are not sufficiently and thoroughly described due to commercially confidential reasons and limitedly available references. In this paper, we are mainly focused on query suggestion which is one of the strategies to improve search performance, and ours is substantially different from those used in the above three systems. We will briefly address these in the following parts.

– Google has a query suggestion service which is developed to offer users "related" search suggestions to help users find what they are looking for more quickly.[1]

[1]Official Google blog: http://googleblog.blogspot.com/2008/06/fresher-related-search-suggestions.html

Other commercial search engines also have implemented query suggestion using their respective techniques, such as Search Assist in Yahoo!, and Related Searches in Live Search. Our paper follows this research direction and studies a basic problem that is how to compute the similarity between two queries by using an expanded feature space. Although the similarity computation between Web pages is widely discussed, Web queries are short in length (two or three keywords) and not rich enough in similarity computation. As pointed out in Section 1, the very small word overlap between short Web queries (Query term based suggestion) makes it hard to accurately estimate their similarity.
Therefore, many current studies focus on how to improve the performance of this naive suggestion strategy by enriching the expression form of query for finding topically related queries. In this paper we observe that utilizing hidden topic model can make an efficient suggestion which is our main contribution.

– Semaphore developed by Smartlogic uses semantic models in the form of taxonomies and ontologies to guide classification.[2] Usually those semantic models define concepts. Although Semaphore is also able to improve search quality, our approach belongs to query suggestion which is orthogonal to it. A common point between Semaphore and ours is the utilization of topics or concepts as the foundation which is popular to narrow the gap between the lexical level of what has been written and the semantic level of what was inferred to. But our approach is different from Smartlogic in the definition of topic, how to get topics and where to use topics.
– Autonomy[3] is on so called conceptual search. The conceptual extraction by Autonomy is performed by clustering the digital content, which is then used for various tasks, such as Automatic Taxonomy Generation, Intent-based Ranking, social search, and so on. However, in our scenario, we used the latent semantic learning to discover the topic space for query suggestion, which is different from clustering.

In a word, inspired by the query suggestion service of famous commercial search engines, our paper studies the problem of topically related query suggestion using a hidden topic model. Smartlogic and Autonomy make use of classification and clustering respectively, to extract topics or concepts which then facilitate search. Therefore, our research is different from them in both strategy and technique aspects.

2.2 Suggesting related queries

Finding topically related queries is becoming an increasingly important research topic that attracts considerable attention. Existing techniques differ from one another in terms of how to improve the naïve query term based suggestion. Usually they are interested in finding additional feature spaces to enrich query expression.

[2] http://www.smartlogic.com/index.php/what-we-do-and-why-it-works

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

2.2.1 Pseudo relevance feedback

Pseudo relevance feedback is widely used. In [16] the authors improve the effectiveness of a user-supplied query by identifying key terms from potentially relevant documents from past queries. In [19], the authors introduce a software agent that collects queries from previous users, and determined the query similarity based on the Web pages returned by queries, and not the actual terms in the queries themselves. In [28], the authors devise a novel rank mechanism for ordering the related queries based on the merging distances of a hierarchical agglomerative clustering (HAC).

As discussed in Section 1, the pseudo relevance based approaches show unsatisfied suggestion efficiency on the fly and low suggestion effectiveness when extracting the features from non-text pages. Our approach utilizes external knowledge from Web (i.e., Wikipedia) to learn a generative probabilistic topic model for the enrichment of Web query representations, which can optimize the suggestion performance of the pseudo relevance based approaches.

2.2.2 Implicit relevance feedback

Recent studies [1, 4, 7, 26, 27] are interested in Web logs where the click information provides available implicit feedback. Some of researchers cluster related queries based on the common clicked URLs two queries share [4, 43]. In [16], the authors propose a novel feature space, called Web community, that generates much better results than the URL feature space. On the other hand, in [1], the authors find related queries based on the content of clicked Web pages using click frequency as a weighting scheme. Their experiments show that the content information is more accurate to measure query similarity than the URL information.

The URL and content feature spaces should be studied as mutual complementation to enrich query representation as discussed in Section 6.8. As we discussed in Section 1, when the input query does not appear in current Web logs, it is difficulty to compute query-to-query similarity. Our approach can handle such queries since we do not need the click information.

There are some recently proposed methods which proceed other directions. Some studies view Web logs as a set of transactions where a single user submits a sequence of related queries in a time interval [7, 17, 37]. Those studies can make meaningful query suggestions by applying traditional data mining techniques, such as association rule mining. These techniques succeed because they mine "wisdom of the crowds" for query understanding. In [9], the authors define a new measure of the temporal correlation of two queries based on the correlation coefficient of their frequency functions, which requires a time-stamped query stream as input and can be used as a complementary approach to our approach and the above discussed methods.

2.2.3 Query expansion

This paper describes a query suggestion approach which suggests topically related queries to help search engine users, while query expansion is also an alternative to revise users queries. The conventional research efforts on query expansion are classified into three categories according to the information sources they use to

find relevant terms for query expansion: (1) corpus-based statistics analysis [42], (2) relevance feedback based recommendation [36], and (3) local context based recommendation [6, 41, 44]. In addition, some researches combined multiple sources of knowledge on term associations [10, 11, 40, 46]. Others used Web logs to bridge the term gap between user-centric query space and author-centric Web page space [12, 13, 44].

Most query expansion techniques suggest terms extracted from Web pages. However, some terms are difficult to be suggested because of their high document frequencies. If these terms appear in some past queries, the full term lists of past queries can be recommended to users. Thus, terms in topically related queries can also be an effective source of expansion terms. Further experiments on query expansion would be an interesting topic in our future work.

2.2.4 Latent topic information in IR

The latent topic information has been researched to analyze queries in the field of information retrieval. Huang et al. [22] compute the query-to-query similarity based on the topic information learned from LDA which training data is clicked documents. He et al. [20] utilize the Interaction Information (II) to detect topics of queries and then replace original queries. Carman et al. [8] apply LDA based personalized ranking formulas to a query log dataset. Takeharu et al. [14] evaluate the effectiveness of a semantic smoothing technique (i.e., LSI) to organize folksonomy tags.

The most related paper to ours is Fan et al. [15] who have proposed a query expansion method. It used a collection of documents retrieved by an input query and its contexts as a training data and then learned the topic distributions of the input query and the candidate terms. The calculation of the topical coherence between the query and each term produced a ranking list to suggest topic-based terms as query expansion. Our work differs from their work in the following aspects. First, the training data they used have to rely on a search process by retrieving search results from a search engine. Our training data is downloaded from Wikipedia in advance. Second, their candidate terms are a set of terms with the prefix of the term that the user is typing. Our candidates are the whole queries submitted before. Third, we estimate the topic-based similarity between queries, while they compute the topical coherence between a query and each candidate term. The common point between ours and theirs is that both utilize LDA to learn latent topics.

2.2.5 Summary

The approach developed in this paper has its own characteristics. First, we build a generative probabilistic model for Web queries beforehand instead of a simple term vector representation. Second, the hidden topics inferred from the learned model are used as a high level feature space over the conventional features like Web page terms when computing query-to-query similarity. Moreover, the proposed approach is fast and feasible in real-time Web applications unlike the traditional pseudo relevance based suggestion that requires the involvement of search engines and it overcomes the drawback of the implicit relevance based suggestion which has a difficulty in handing the queries not in current Web log.

3 Framework of our approach

Suppose that we have already obtained a set of candidate queries which are past queries submitted by previous users, our task is to suggest the most topically related candidate queries for a target input query.

First, in Figure 1 we illustrate the overview of the traditional pseudo relevance based suggestion which uses the term feature space.

In the offline step, the top search results of past queries are retrieved by a search engine, and terms are extracted from the search result pages. In this manner, each past query is represented by the terms of its search result pages. In the online step, the same process is used to get the term expression of an input query, which requires the work of a search engine. The query to query similarity is computed based on their term expressions. The past queries with the highest similarity scores to the input query are recommended to users. Likewise, pseudo relevance based suggestion approaches using the URL feature space work in a similar way shown in Figure 1. The only difference is taking the place of terms with URLs in query expression. From Figure 1, we easily see that the tradition pseudo relevance based approach is a time-consuming and high-cost process because it needs to retrieve the top search results from a search engine for each input query.

The overall framework of our approach as shown in Figure 2 consists of two steps.

In the offline model-learning step, we first choose a training data. Then, a model estimation procedure applies the topic analysis on the training data, which can be performed by using one of the well-known hidden topic analysis models such as probabilistic latent semantic analysis (pLSA) [21] or Latent Dirichlet Allocation (LDA) [5]. Here we chose LDA because it has a better defined generative model of co-occurrence observations than pLSA and an advantageous capability of conducting topic probability assignments for a previously unseen text. LDA will be briefly introduced in Section 4. With the LDA model, we are able to estimate the latent topic model for candidate queries.

In the online suggestion step, we use the trained topic model to infer the topic distribution for a target input query. With the inferred topic distribution, the query in an original expression of lexical occurrence is transformed into a reduced and latent topic space, which captures the semantic meaning of the query. We can use the transformed query expression to measure the similarity between various queries.

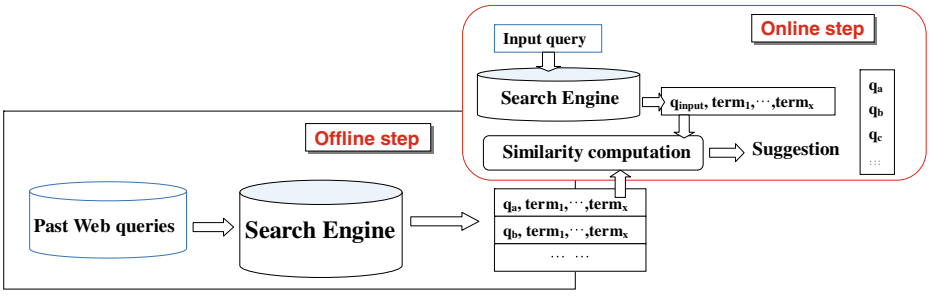

Figure 1 Framework of the traditional pseudo relevance based suggestion using the term feature space.

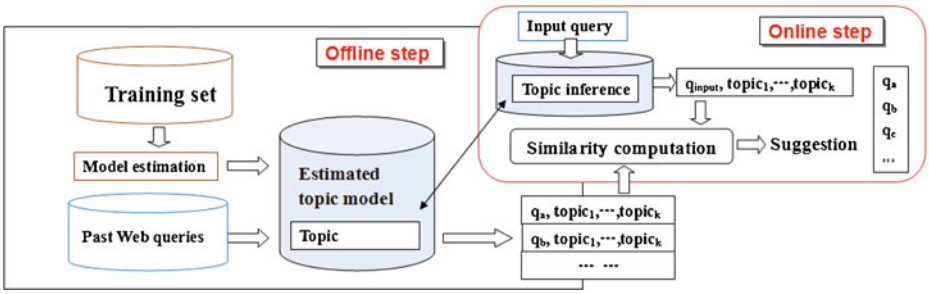

Figure 2 Framework of our approach.

After the similarity score between the input query and each of candidate queries is calculated based on their topic distributions, our approach suggests the most topically related queries with the highest similarity scores to the current users from the candidates. Our approach does not need to use a search engine at online step, which is good for large-scale real time Web applications for high efficiency and acceptable suggestion precision.

As seen from Figures 1 and 2, the difference between our approach and traditional pseudo based query suggestion approaches is where and how to integrate a feature space. In our experiment section, we will show how different feature spaces affect the suggestion performance.

4 LDA model

LDA is a probabilistic generative model for a text corpus. The basic idea is that documents are represented as random mixtures over latent topics and each topic is characterized by a distribution over words. More theoretically, LDA is based on the assumption that there exists an unseen structure of "topics" or "themes" in the text corpus, which governs the co-occurrence observations. As such, the intuition behind LDA is to discover this latent topic structure via estimating the probability distribution of the original co-occurrence activities. The notations used in the algorithm are described in Table 1.

Table 1 Notations of LDA model.

M:	The number of documents
K:	The number of topics
V:	The size of vocabulary
α, β:	Dirichlet parameters
θ_m:	The topic distribution vector of the document m, i.e. each element in the vector denotes the probability estimate of the document m on the corresponding topic k
Θ:	The topic estimates of the corpus, i.e. a $M \times K$ matrix of a collection of θ_m
φ_k:	The word distribution vector of the topic k, i.e. each element in the vector represents the word assignment of topic k with respect to the correlated word n
Φ:	The word assignments of the topics, i.e. a $K \times V$ matrix of a collection of φ_k

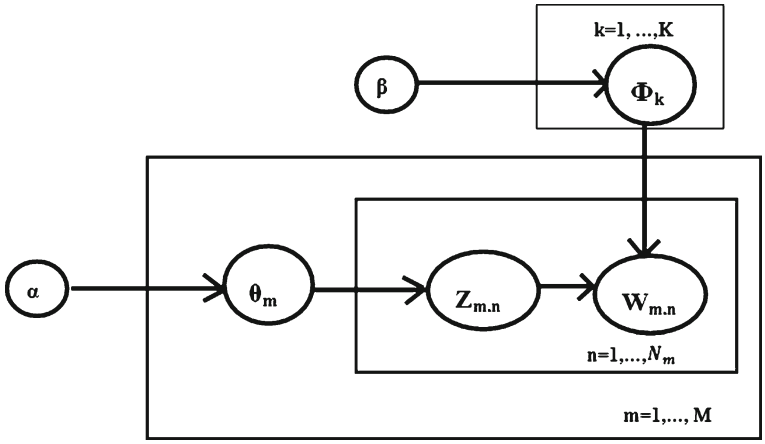

Figure 3 The illustration of LDA.

The generative procedure of LDA model is shown in Figure 3 and the pseudo codes of its implementation algorithm are shown in Table 2. The model formulation is also described as follows.

In LDA, a document $d_m = \{w_{mn}, n = 1, ..., N_m\}$ is generated by picking a distribution over the topics from a Dirichlet distribution ($Dir(\alpha)$). And given the topic distribution, we pick the topic assignment of each specific word. Then the topic assignment for each word placeholder $[m, n]$ is calculated by sampling a particular topic $z_{m,n}$ from the multinominal distribution of $Mult(\theta_m)$. And finally, a particular word of $w_{m,n}$ is generated for the placeholder $[m, n]$ by sampling its weight from the multinomial distribution of $Mult(\varphi_{z_{m,n}})$. Known from the above description, given Dirichlet parameters α and β, we can formulate a joint distribution of a document d_m, a topic mixture of d_m, i.e. θ_m , and a set of N_m topics, i.e. z_m as follows:

$$P_r(\theta_m, z_m, d_m, \Phi \,|\alpha, \beta) = P_r(\theta_m \,|\alpha)\, P_r(\Phi \,|\beta) \prod_{n=1}^{N_m} P_r\left(w_{m,n} \left|\varphi_{z_{m,n}}\right.\right) P_r\left(z_{m,n} \,|\theta_m\right)$$

And integrating over θ_m, $\phi_{z_{m,n}}$ and summing over z_m , we obtain the likelihood of the document d_m:

$$P_r(d_m \,|\alpha, \beta) = \int \int P_r(\theta_m \,|\alpha)\, P_r(\Phi \,|\beta) \prod_{n=1}^{N_m} P_r\left(w_{m,n} \left|\varphi_{z_{m,n}}\right.\right) P_r\left(z_{m,n} \,|\theta_m\right) d\theta_m d\Phi$$

Table 2 Algorithm: Generation Process of LDA.

for each of topics
  sample the mixture of words $\phi_k \sim Dir(\beta)$
end
for each of documents $m = 1 : M$
  sample the mixture of topics $\theta_m \sim Dir(\alpha)$
  sample the lengths of documents $N_m \sim Poiss(\xi)$
  for each of words $n = 1 : N_m$ in the document m
    sample the topic index of $z_{m,n} \sim Mult(\theta_m)$
    sample the weight of word $w_{m,n} \sim Mult(\phi_{z_{m,n}})$
  end
end

Finally the likelihood of the document corpus $D = \{d_m, m = 1, ..., M\}$ is a product of the likelihood of all documents in the corpus, defined as

$$P_r(D|\alpha, \beta) = \prod_{m=1}^{M} P_r(d_m|\alpha, \beta). \quad (1)$$

Under this generative model, documents can be modelled as probability distributions over multiple topics via an estimation procedure. In other words, the inputs of LDA model are terms of documents and its outputs are topic distributions.

5 Our two-step approach

We dedicate this section to describe our query suggestion approach in detail. The strength of query suggestion is to facilitate users who are not very familiar with a certain domain, refining and formulating their needed queries from related queries that have been searched by previous users, and hence getting the Web pages they want.

5.1 Offline model-learning step

5.1.1 Collecting a training dataset

Our idea is to build a latent topic model for Web queries. First we need to find a suitable data collection for model training. Since search engine users submit keyword-based queries to satisfy their information needs from Web, it is easily to obtain a query-term dataset at first for the purpose of model training. The query, however, is always in a representation of short words, thus it is not applicable to utilize the terms included in the query alone for training. Here we intend to refer to external data sources. There are a number of potential data sources usable as training data, such as Wikipedia, Open Directory Project (ODP), Web archive crawled before, Web data in TREC Web track, and so on. The only requirement for choosing the training data is that the concepts of Web queries should be covered in such Web corpora. For example, a log file of a search engine stores a large-scale Web archive which can be used as a potential training set to satisfy the diversity and the coverage of queries.

Section 6 discusses an implementation on how to construct a training dataset using the featured articles of Wikipedia. Once the training dataset is collected in a form of document-term array where each row denotes a document and each column corresponds to a term, we can then employ the estimation procedure of LDA to derive the latent topic model from the training dataset.

5.1.2 Model estimation

In order to use the topic model obtained by LDA, we need to solve the problem of computing the posterior distribution over the hidden topics given a query. In fact each query is equivalent to an especially short document with a few keywords, thus we employ model estimation on this sparse document-term array. However, estimating the parameters of LDA by directly and exactly maximizing the likelihood of the whole data collection in (1) is intractable. The solution to this is to use alternative approximate estimation methods. Here we employ the variational EM

Table 3 An example of EM execution in terms of likelihood.

Iteration no.	1	2	...	30	31
l_k	−157796775.1	−109932143.4	...	−100171946.3	−100162498.2
Diff. ratio	3.03E-01	4.07E-02	...	1.02E-04	9.43E-05

algorithm [5] to find the variational parameters that maximize the total likelihood of the corpus with respect to the model parameters of α and β:

$$(\alpha_{\text{est}}, \beta_{\text{est}}) = \max l(\alpha, \beta) = \max \sum_{m=1}^{M} \log Pr(d_m \mid \alpha, \beta)\,. \tag{2}$$

The variational EM algorithm is briefly described as follows:

1. (E-step) For each document, find the optimizing values of variational parameters θ_m^* and φ_m^*.

2. (M-step) Maximize the resulting low bound on the likelihood with respect to model parameters α and β. This corresponds to finding maximum likelihood estimates with the approximate posterior which is computed in the E-step.

The E-step and M-step are executed iteratively until a maximum likelihood value reaches. Meanwhile, the calculated estimation parameters can be used to infer topic distribution of a new document by performing the variational inference. For example, for the Wikipedia training dataset we used to estimate the topic model, the EM algorithm was iterated for 31 times with the monotonic increase of the likelihood until the convergence of the likelihood value (i.e., the iterations repeat until $(l_{k+1} - l_k)/l_k < 1e - 5$). Below Table 3 lists the changes of two consecutive likelihood values.

5.1.3 Topic inference

After doing the topic estimation using the training dataset, we eventually obtain the posterior parameters of the LDA model α and β as well as the topic assignments of documents and terms. Then the trained topic model could be used for topic inference for a target input query (here a query is considered as a document with a few terms). In this case, the inference is done through the variational algorithm [5]. As a result, an inferred topic distribution of the target query is calculated, which reflects the likelihood of various topic assignments of the query. As the dimensionality of the topic space is much smaller than that of the original term space, the LDA operation could be viewed as a dimensionality reduction method where the latent semantic topic of the document is optimally approximated. In other words, after the LDA model estimation, the expression of the input query is transformed from a high-dimensional sparse term space to a low-dimensional latent topic space.

To illustrate the how the inferred topic distribution could be used to differentiate the queries from each other. In Figure 4, we also give an example of query # 6 from KDDCUP dataset in terms of topic distribution. From the figure, we can clearly see the topic correlation of this query to a few specific topics like 7, 31, 38 and so on.

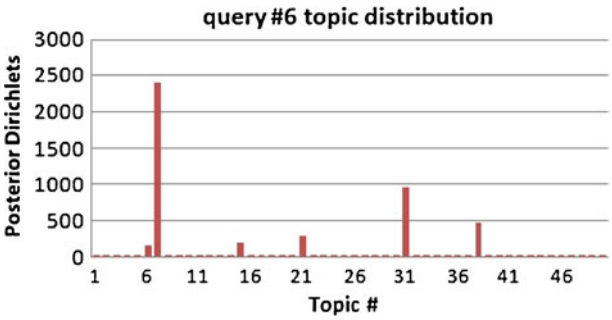

Figure 4 An example of query topic distribution.

5.2 Online Suggestion Step

After completing the offline step consisting of model estimation and topic inference, each input query is re-expressed by its topic distribution. Then, based on the transformed expression, we can measure the similarity between various queries.

5.2.1 Similarity computation

As discussed above, each input query is expressed as a probability distribution over the latent topics (i.e. θ_{ij}), by selecting a threshold value μ, we can obtain the following vector expression with respect to significant topic distribution alone: $q_i = q_{tp} = \{s_{i1}, ..., s_{ik}, ..., s_{iK}\}$, where

$$s_{ij} = \begin{cases} 0 & : \quad \theta_{ij} < \mu \\ \theta_{ij} & : \quad otherwise \end{cases} .$$

The parameter μ depends on the topic distribution learned from the training data. In our experiments, we set μ to be 0.1. The similarity function used is the well-known cosine coefficient, defined as:

$$sim(q_i, q_j) = \cos(q_i, q_j) = \frac{(q_i \cdot q_j)}{\|q_i\| \times \|q_j\|} .$$

The cosine function is widely employed in information retrieval [32].

Finally, our approach outputs a ranked list of candidate queries according to their similarity scores to the input query. In our experiments, we evaluate the suggestion performance using this kind of query expressions and compare it with others using traditional terms and URLs as features.

5.2.2 Other available similarity measures

We would like to note that our work can easily incorporate other similarity and specificity measures. For example, L_1, Ward and Jensen-Shannon (JS) divergence [29] are common to measure the distance (similarity) between distributions. In text processing, a natural measure of similarity of two documents is the similarity between their word conditional distributions. Roughly speaking, documents with similar conditional word distributions would be related. This idea was first intro-

duced in [34] and was called "distributional clustering". Similarly, in our problem we can recommend related queries with similar conditional distributions of features in each feature space.

In addition, recent random walk based rank methods are popular. SimRank [24] measured the object-to-object relationship by recursively scoring the similarity of their related objects. Sun et al. [39] employed random walk with restarts and graph partitioning to solve two problems, neighborhood formation and anomaly detection. The two studies reflect the global similarity since they work by iteratively transmitting weights through the whole graph. The approach proposed by Ma et al. [30] is based on Markov random walk and hitting time analysis on the query-URL bipartite graph. It can prevent semantically redundant queries from receiving a high rank, hence encouraging diversities in the results. Song et al. [38] combined both the click and skip information from users and uses a random walk model to optimize the query correlation.

Due to the fact that this paper focuses on an empirical study of different *feature spaces* for query enrichment to improve the quality of finding related queries, we omit the further discussion on more complex similarity measures. In addition, experimental results show that our similarity equation can effectively find related search queries.

5.3 Optimization of similarity computation

For the online suggestion, our approach needs to compute the similarities between a query and all candidates. If there are larger numbers of candidates, we should consider how to do efficient suggestion, especially for large mounts of Web data. Given that the topic based query representation is obtained in our offline step, should we have to compute similarities for all pairs of queries that will count a quadratic number of values in the online process?

We introduce an optimization technique to reduce the cost of similarity computation. Recall that we are interested in pairs of queries whose similarity is above a specified threshold, a high quality collection. The recent work [3] addressed this scalability issue without relying on approximation methods or extensive parameter tuning, and described different monotone minimum constraints for different similarity measures before computing the similarity scores. The authors of [3] have described how to get constraints for *Overlap*, *Dice*, and other popular measures in details (L_1 norm can be regarded as a kind of *Dice* measure).

Here we use Cosine measure as an example to explain their basic idea. We have pairs of queries whose similarity is above δ'. For a given vector x we denote the maximum value $x[i]$ over all i as $maxweight(x)$. By definition, if two vectors q_i and q_j meet the similarity score threshold then we get

$$\cos(q_i, q_j) \geq \delta' \tag{3}$$

Now, note that the following inequality is established:

$$\cos(q_i, q_j) \leq maxweight(q_i)|q_j| \tag{4}$$

Thus, we have that

$$maxweight(q_i)|q_j| \geq \delta'$$

$$|q_j| \geq \frac{\delta'}{maxweight(q_i)} \tag{5}$$

The above inequality (5) is a size constraint which tells us if $|q_j| \leq \frac{\delta'}{maxweight(q_i)}$, their similarity score does not need to be stored to reduce the amount of candidate pairs. Therefore, we can sort queries in the decreasing order of $maxweight(q_i)$ to save on computation time. This preprocessing means that if $|q_j| \leq \frac{\delta'}{maxweight(q_i)}$ is met, the queries q_j to last query can be skipped without doing similarity computation with q_i, thus accelerating our approach.

6 Experiments

In this section, we conduct experiments on two real data sets for the comparisons of suggestion quality from two aspects. One aspect is to investigate the suggestion effectiveness of the three feature spaces (i.e., topic, term and URL) and the optimized parameter selections to find the best parameter set for each one. Because the term and URL feature spaces are frequently used in both of the traditional pseudo relevance based and implicit relevance based approaches, the other is to compare the effectiveness and the efficiency of our topic feature space based approach with them.

6.1 Evaluation measures

Two evaluation metrics, $Precision@N$ and $MAP@N$ are introduced [32]. The former computes the percentage of related queries at the top N queries of a suggestion list. Since query suggestion actually is a ranking problem we also utilize the latter (Mean Average Precision) which takes into account the positions of related queries. Given a query q_i, its average precision (AP) is defined as:

$$AP_i@N = \frac{\sum_{j=1}^{N}(Precision@j * pos(j))}{\#\ of\ related\ queries\ of\ q_i\ at\ Top\ N} \tag{6}$$

where N, as an evaluation parameter, is the number of suggested queries in a suggestion list (e.g., 10, 20 and so on). $Precision@j$ is the precision score at position j, and $pos(j)$ is a binary function to indicate whether the query at position j is relevant. Then, we obtain MAP scores by using the average value of the AP values of all the test queries which number is Q. MAP is defined as:

$$MAP@N = \frac{\sum_{i=1}^{Q} AP_i@N}{Q} \tag{7}$$

The difference between Precision and MAP is that Precision only counts the number of right answers while MAP computes the average of their positions. We investigate not only the number of related queries in a suggestion list, i.e., *Precision*, but also the order in which the queries are returned, i.e., MAP, in the evaluation of query suggestion.

6.2 Data preparation for LDA

The first component of our LDA approach as shown in Figure 2 is the training data. Given the set of testing queries, choosing an appropriate training data is important. This is because the topics analyzed from this training data directly influence the learning and suggesting performance of our approach. Our training data is from the featured articles of Wikipedia. Featured articles are considered to be the best articles in Wikipedia, as determined by Wikipedia's editors. The contents of these articles are downloaded from Wikipedia website. After removing HTML tags and stop words, we stem the left terms and use the traditional term frequency (TF) as the term weight. Finally, we get 2,724 Web pages and 415,266 distinct terms which are used as our training data. We assume that the crawled Wikipedia articles working as a small scale Web archive can cover the content topics that are relevant to the queries themselves for topic model learning and inference.

Last, we report the computing setting parameters of LDA model. At the initialization stage, α is set to be 0.1 and β is accordingly generated by the program shown in Table 2 and based on α and the training data. Then, the LDA model estimation uses the variational EM algorithm [5] to optimally choose the variational parameters that maximize the total likelihood of the corpus. The formula is given in (2) where the EM algorithm is executed iteratively until a maximum likelihood value reaches (i.e., the iterations repeat until $(l_{k+1} - l_k)/l_k < 1e - 4$).

6.3 Comparisons with the pseudo relevance based approach

We compare our proposed topic model with the URL model and the term model both of which extract terms or URLs from top retrieved search results. Here, search results are regarded as the pseudo relevances. In this part, we first present an optimized parameter selection study, and then investigate suggestion performance derived by using these various feature models. The comparisons of query suggestion performance and related discussion are given as well.

6.3.1 Experimental data

Our goal is to find topically related queries from a set of candidates given an input query. The query data are a collection of the 800 categorized queries provided by KDD cup 2005 for the reason of easily and accurately evaluating. The KDD cup data can be downloaded.[4] The 800 queries are search engine queries from end user Internet search activities. We should judge whether two queries are topically related or not. Here an automatic judgment method is introduced rather than manually examining each query by human experts. In KDD cup 2005 data, each query is labeled into at most five topic categories by three human labellers, if two queries share at least one same category, we regard them as related queries. Also, when one query is used as an input query, the left 799 queries are considered as the candidate queries from which a suggestion list is made. We report the evaluation results averaged by 800 queries and the three labellers in the following part.

[4] http://www.acm.org/sigs/sigkdd/kddcup/index.php

6.3.2 Results of The topic model

Since LDA produces the likelihood of various topic assignments for each query, we first conduct experiments by varying the number of latent topics (K). The evaluation results are show in Figure 5 in terms of Precision and MAP at top N= 5, 10, 15, and 20.

From Figure 5, we find that the best suggestion results are generally obtained at K= 50. Especially, the highest MAP scores are 0.5802 ($N = 5$), 0.5575 ($N = 10$), 0.5389 ($N = 15$), and 0.5235 ($N = 20$) which are all achieved at $K = 50$.

6.3.3 Results of the URL model

We use URLs to express queries and compute query-to-query similarity. The work mechanism is the same as that shown in Figure 1 by replacing terms by URLs. Let q_i and q_j be two queries, and $U(q_i)$ and $U(q_j)$ be the two sets of URLs returned by a search engine (here i.e., Google) in response to the two queries q_i and q_j respectively. In this case, Jaccard is an intuitive and suitable similarity measure, defined as

$$Jaccard(q_i, q_j) = \frac{U(q_i) \cap U(q_j)}{U(q_i) \cup U(q_j)}. \tag{8}$$

The value of the defined similarity between two queries lies in the range [0, 1]: 1 if they are exactly the same URLs, and 0 if they have no URLs in common. For each query in KDD cup data set, a suggestion list is generated according to (8). The Precision and MAP scores are shown in Figure 6a and b respectively.

M is the number of top search results used in similarity computation. From Figure 6, the highest Precision and MAP scores are achieved at $M = 100$.

6.3.4 Results of the term model

The work mechanism of the term model is shown in Figure 1. We report the suggestion results of using terms as features. Given the 800 testing queries, we downloaded the snippets and titles of their top search results and extracted terms from these pages to represent queries. The total number of extracted terms is 137,389. The computation of query similarity uses the cosine similarity measure by the traditional TF*IDF term weighting scheme. The suggestion results are shown in Figure 7.

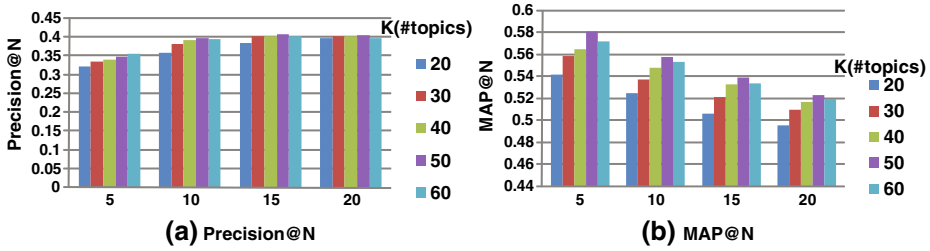

Figure 5 Results of using latent topics.

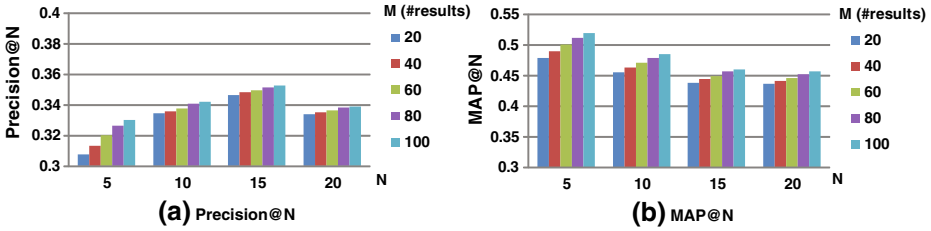

Figure 6 Results of using URLs.

The best suggestion results are generally obtained at $M = 40$ (M is the number of top search results). Especially, the highest MAP scores are 0.7763 ($N = 5$), 0.7314 ($N = 10$), 0.6967 ($N = 15$), and 0.6713 ($N = 20$) which are all achieved at $M = 40$.

6.3.5 Comparisons among the three feature models

Here, we put the highest *Precision* and MAP scores of the three features together for performance comparisons, as listed in Tables 4 and 5. It is clear that our topic space improves the suggestion performance over the URL space.

Topics are learned from training data. Using latent topics in query expression largely reduces the dimension of the term space. Our experimental results also verify this. In this experimental data, the number of topics is 50 while that of terms is 415,266, so the dimension of topic feature space has been reduced by 8,305 times compared to term space. Tables 4 and 5 show the topic feature space produces acceptable results compared with the term space in terms of *Precision*@N and $MAP@N$.

6.4 Comparisons with the implicit relevance based approach

The implicit relevance based approach usually uses the terms or URLs extracted from clicked search results to enrich query representation. In this part, we compare our proposed topic model with the URL model and the term model based on clicked search results. The query-to-query similarity measures are the same as those used in the above pseudo relevance based approach.

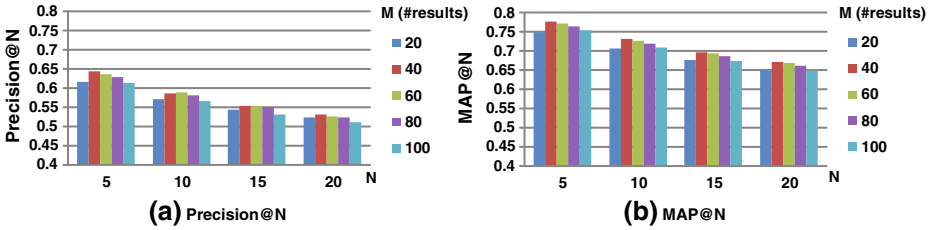

Figure 7 Results of using terms.

Table 4 Comparisons among three features (Precision).

N	URL	Term	Topic
5	0.33	0.6437	0.3486
10	0.3424	0.587	0.3980
15	0.3529	0.5543	0.4023
20	0.3390	0.5300	0.4066

6.4.1 Experimental data

We have carried out experiments on the MSN search spring query logs and click through data from May 1, 2006 to May 31, 2006. There are 132,257 queries and 199,798 URLs involved in the dataset, called MSN Data. To clean the data we first discarded all queries which did not result in a click on a URL. We then removed the queries which terms do not appear in our Wikipedia training data. All punctuation was removed and Porters algorithm was used for stemming. We did not remove any stopwords. The resulting dataset contained 63,375 queries and had a vocabulary of 15,619 words extracted from the clicked Web pages (i.e., search results). We randomly selected 10,000 issued queries as the candidate set. Our task is to recommend top related queries from the 10,000 candidates given an input query and the evaluated queries are randomly selected 100. The relevance of each suggested query is manually labeled in a blind test manner, that is, the human evaluator did not know which algorithm returned the terms. We qualify the effectiveness through popular measures of $Precision@10$ and $MAP@10$.

6.4.2 Results of the three feature models

Table 6 lists the experimental results of our topic based approach (i.e., topic model) and the implicit relevance based approach (i.e., term model and URL model).

These results are similar to those reported in Tables 4 and 5. The term model produces the best suggestion results and our topic model is much better than the URL model. In addition. the topic space express queries in less dimension than the term space. In this experimental data, the number of topics is 50 while that of terms is 15,619, which means the query expression dimension has been reduced by 312 times. Table 6 says that the topic model works well and shows comparable suggestion quality compared with the term model in terms of $Precision@10$ (0.582 vs. 0.597). Although its $MAP@10$ scores are lower than those of the term topic model, the number of good suggestion results are more important than their positions in the suggestion list due to the shortness of queries. The length of a keyword-based query is much shorter than that of a Web page, Web users can read the top 10 or 20 recommended queries much more quickly than they browse and click the total top Web pages one by one.

Table 5 Comparisons among three features (MAP).

N	URL	Term	Topic
5	0.5203	0.7763	0.5802
10	0.4856	0.7314	0.5575
15	0.4612	0.6967	0.5389
20	0.4567	0.6713	0.5235

Table 6 Comparisons among three features.

	URL	Term	Topic
Precision@10	0.1620	0.5970	0.5820
MAP@10	0.0792	0.8363	0.4978

6.4.3 Quality of the top one suggestion

The users behavior in using and interacting with the search engines tend to be quite simple and straightforward. For example, it often happens that a search engine suggests no more than one related keyword. In order to evaluate the relatedness of the top suggested term to the input query, we conducted experiments to compute how many suggested queries at the top place of the suggestion lists are judged as related ones on MSN search log data. The results of the URL, term and topic spaces are 8/100, 29/100, 47/100. Thus our topic space shows the best performance, i.e., its *Precision*@1 is the best. Due to the phenomenon of polysemy of word, in some cases the systems might suggest two or more terms dependent on the different meanings of input query such as Java, Puma etc.

6.5 Summary of effectiveness evaluation

From the above results of effectiveness evaluation, we find that our proposed topic feature space represents Web queries in a low-dimensional latent topic space instead of a high-dimensional snippet term space, but it still exhibits good performance. Although using the term feature space produces the highest precision and MAP scores, it has two drawbacks. One is that we have to retrieve top search results at the online step. The other is that term is a content-sensitive feature space which is sometimes not applicable in scenarios such as image or video Web pages, and the content processing is very time-consuming as well. Our topic feature space, in contrast, only utilizes the terms of query as an input along with the topic model trained in the offline stage, thus, is able to substantially handle the two mentioned drawbacks.

Furthermore, we find that directly using URLs for query expression shows much lower Precision and MAP scores than using topics or terms. For example, in Table 6 the Precision score of using URLs is only 0.162 while the MAP scores of using terms and topics are 0.5970 and 0.5820 respectively. Topic and term are more effective than URL. It might be due to that even though the URLs of two Web pages are different, their contents may be related to each other or they may cover similar topics.

6.6 Evaluation of efficiency

Last we report the run time of online query suggestion when using each feature space in Table 7.

The experiments are conducted on a PC with 2G memory and a 3.0 GHz CPU.

Table 7 Online run time of using each feature space.

Run time	Topic	URL	Term
KDD Data	37.5 ms	405.3 ms	1024.5 ms
MSN Data	1.08 s	3.43 s	367.2 s

In KDD Data, the run time of using URLs includes retrieving top search results, URL based similarity computation and sorting; the run time of using terms consists of retrieving top search results, text processing, cosine similarity computation and sort. If the two approaches are run on the server of a search engine, the time of retrieving top search results can be reduced since it will not be affected by unwanted Internet traffic and searching. A report from Google Official Blog[5] says that ranking is done by Google in a few milliseconds. Averaged by 800 queries of KDD Data, the suggestion times of using URLs and using terms are about 405.3 and 1024.5 ms, respectively at client side. In contrast, our approach only needs to do the topic inference, compute the topic based similarity and sort the similarity scores once the topic model training is completed offline.

As seen from Table 7, the efficiency of our approach is much better than the URL and term space. This observation is attributed to two aspects. First, the number of topics (i.e., 50) we used is smaller than that of URLs (i.e., 100) and that of terms (i.e., 415,266), which lowers the run time of similarity computation. Second, we do not rely on a search engine for getting search results. In particular, at the online suggestion stage our approach does not require text processing which costs much time for query suggestion. The suggestion time of our averaged approach is 37.5 ms.

Although using MSN Data for the implicit relevance based approach dose not need to retrieve search results online, the candidates increase to 10,000. Therefore, as shown in Table 7, its run times of the three feature spaces all become longer than using KDD Data. The run time level changes from millisecond to second. The dimension of our topic model is still only 50 and its cosine computation is the fastest. The Jaccard similarity computation in the URL model is faster than the cosine similarity computation in the term model which dimension is 15,619.

The experimental results on running costs verify that the suggestion efficiency of our topic feature space is much higher than the traditional term or URL feature space.

6.7 The examples of suggested queries

Here, we also present a few examples of related queries suggested via the proposed topic feature space. Tables 8 and 9 lists the top five suggestion results corresponding to each input query.

In Table 8, the first input query is “airline carryon restrictions” which may mean that a user wants to find the rules or regulations of carryon restrictions before taking a flight. The suggestion of related queries is composed of a variety of airline companies from which the carryon restrictions could be found. The suggestion results of the second query are all about jewellery or other ornaments because the input query is “anniversary ring”. The last input query and its related queries are about sightseeing places. In Table 9, the suggested queries are “bmw”, “Iseki tractor” and other vehicle related ones when the input query is “baxter chrysler”, a jeep dodge.

Recall that our two test query sets are 800 and 10,000, respectively. The task of our experiments is to find the most related queries from a test set given one input query, which means that the suggestion results produced by our approach are local optimal and are the best answers among 800 or 10,000 queries. The larger the experimental data set is, with the higher probability the global optimal results can be found.

[5] http://googleblog.blogspot.com/2008/05/introduction-togoogle- search-quality.html

Table 8 The top 5 suggestion results of each input query (KDD Data).

Input query	The top 5 suggestion results				
Airline carryon restrictions	Great lake aviation	Airline reservations	Chalks airlines	Southeast airlines	Southwest airlines
Anniversary ring	Mens wedding rings	Cross pendant	Gold chains	Silver necklace	Sterling custom cabinetry pa
Cacapon resort west	Virginia north Carolina mountains	North county toyota	Balmorhea blue agate	Canton ohio old pewter plate plate	Waynesville nc

6.8 Discussions

Based on the above experimental results, we have further discussions on the following three issues:

- There are various ways for training data selection in real applications. The best one should be rich enough to cover as broad and diverse as possible words, concepts, and topics that are relevant to Web queries. Some public universal Web sources are available, such as ODP, Wikipedia, and so on. In the Web search engine side, Web pages has already crawled and stored locally, and the meanings of all Web queries inputted by users are assumed to be encoded in these Web pages. Therefore, if LDA is run at the search engine side, all the crawled Web pages can be used as training data. Of course, there are also domain-specific corpora for domain-specific applications. In this paper, since our object is a small part of Web search engine queries, we have chosen featured articles of Wikipedia as training data instead of a large-scale Web corpus.
- Our approach can be also available if using the implicit relevance which requires click information beforehand. The training data can be the clicked Web pages by users. The disadvantage of the implicit relevance is that the click data is not available when the query is not submitted before.
- Our experimental results show that the term feature space shows the best quality in terms of precision. However, term are content-sensitive feature spaces which are sometimes not applicable, at least in principle, in settings like non-text pages like multimedia (image) files, Usenet archives, sites with registration requirement, and dynamic pages returned in response to a submitted query and so forth. In contrast, URL is a content-independent feature, which is generally available in all types of Web pages. In addition, our topic model also does not

Table 9 The top 5 suggestion results of each input query (MSN Data).

Input query	The top 5 suggestion results				
Baxter chrysler	BMW	Iseki tractor	Yamaha rhino products	American lemans race	Car games
Ebay	Music store	Free birthday cards	Ebay motors	Michael Duwayne clay	Michael D clay
CNBC	Elenor Roosevelt quotes	Netflix	TV guide	David Blaine	Josh Groben

require the textual contents of search results. Therefore, these three feature models have distinct strengths and drawback - the execution cost of the topic model is the smallest with the comparable suggestion performance, while the term space is the most time-consuming even it could achieve the best suggestion results.

7 Conclusion and future work

In this paper, we presented a novel approach to related query suggestion using hidden topic model. Unlike previous methods, our approach provides the hidden topics of Web queries to represent their semantic meanings. Moreover, based on the learned topic model using LDA we can give accurate and fast suggestion for a short query no matter whether the input query was appeared in the past query archive or not. The experimental results on two datasets clearly demonstrated that the topic feature space obtained by our approach is effective in terms of suggestion precision and efficient in terms of online suggestion run time. In the future, we plan to study the effect of different Web corpora on query suggestion. Also, how the terms extracted from the whole search results affect on suggestion results is an interesting topic. We can easily extend our approach to the Question & Answer system which is to match the question issued by a user to the most relevant questions previously answered by human experts.

Acknowledgement This paper is supported by NSFC Project 61003130.

References

1. Baeza-Yates, R.A., Hurtado, C.A., Mendoza, M.: Improving search engines by query clustering. J. Am. Soc. Inf. Sci. Technol. **58**(12), 1793–1804 (2007)
2. Balfe, E., Smyth, B.: An analysis of query similarity in collaborative Web search. In: Advances in Information Retrieval, 27th European Conference on IR Research, (ECIR'05), pp. 330–344. Santiago de Compostela, Spain (2005)
3. Bayardo, R.J., Ma, Y., Srikant, R.: Scaling up all pairs similarity search. In: Proceedings of the 16th International Conference on World Wide Web (WWW'07), pp. 131–140. Banff, Alberta, Canada (2007)
4. Beeferman, D., Berger, A.L.: 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. Boston, MA (2000)
5. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
6. Buckley, C., Salton, G., Allan, J., Singhal, A.: Automatic query expansion using smart. In: Proceedings of Text REtrieval Conference (TREC'03), pp. 69–080. Gaithersburg, Maryland (2003)
7. Cao, H., Jiang, D., Pei, J., He, Q., Liao, Z., Chen, E., Li, H.: Context-aware query suggestion by mining click-through and session data. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD'08), pp. 875–883. Las Vegas, Nevada (2008)
8. Carman, M.J., Crestani, F., Harvey, M., Baillie, M.: Towards query log based personalization using topic models. In: Proceedings of the 19th ACM Conference on Information and Knowledge Management(CIKM'10), pp. 1849–1852. Toronto, Ontario (2010)
9. Chien, S., Immorlica, N.: Semantic similarity between search engine queries using temporal correlation. In: Proceedings of the 14th international conference on World Wide Web, (WWW'05), pp. 2–11. Chiba, Japan (2005)

10. Chirita, P.A., Firan, C.S., Nejdl, W.: Personalized query expansion for the Web. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07), pp. 7–14. Amsterdam, The Netherlands (2007)
11. Collins-Thompson, K., Callan, J.: Query expansion using random walk models. In: Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management (CIKM'05), pp. 704–711. Bremen, Germany (2005)
12. Cui, H., Wen, J.R., Nie, J.Y., Ma, W.Y.: Query expansion by mining user logs. IEEE Trans. Knowl. Data Eng. **15**(4), 829–839 (2003)
13. Dolog, P., Stuckenschmidt, H., Wache, H., Diederich, J.: Relaxing rdf queries based on user and domain preferences. J. Intell. Inf. Syst. **33**(3), 239–260 (2009)
14. Eda, T., Yoshikawa, M., Uchiyama, T., Uchiyama, T.: The effectiveness of latent semantic analysis for building up a bottom-up taxonomy from folksonomy tags. World Wide Web **12**(4), 421–440 (2009)
15. Fan, J., Wu, H., Li, G., Zhou, L.: Suggesting topic-based query terms as you type. In: Advances in Web Technologies and Applications, Proceedings of the 12th Asia-Pacific Web Conference(APWeb'10), pp. 61–67. Buscan, Korea (2010)
16. Fitzpatrick, L., Dent, M.: Automatic feedback using past queries: social searching? In: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97), pp. 306–313. Philadelphia, PA (1997)
17. Fonseca, B.M., Golgher, P.B., Pôssas, B., Ribeiro-Neto, B.A., Ziviani, N.: Concept-based interactive query expansion. In: Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management, (CIKM'05), pp. 696–703 (2005)
18. Fu, L., lian Goh, D.H., boon Foo, S.S.: The effect of similarity measures on the quality of query clusters. J. Inf. Sci. **30**(5), 396–407 (2004)
19. Glance, N.S.: Community search assistant. In: Proceedings of the 2001 International Conference on Intelligent User Interfaces (IUI'01), pp. 91–96. Santa Fe, NM (2001)
20. He, X., Yan, J., Ma, J., Liu, N., Chen, Z.: Query topic detection for reformulation. In: Proceedings of the 16th International Conference on World Wide Web (WWW'07), pp. 1187–1188. Banff, Alberta (2007)
21. Hofmann, T.: Probabilistic latent semantic analysis. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99), pp. 289–296. Stockholm, Sweden (1999)
22. Huang, S., Zhao, Q., Mitra, P., Giles, C.L.: Hierarchical location and topic based query expansion. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI'08), pp. 1150–1155. Chicago, Illinois (2008)
23. Jansen, B.J., Spink, A., Bateman, J., Saracevic, T.: Real life information retrieval: a study of user queries on the Web. SIGIR Forum **32**(1), 5–17 (1998)
24. Jeh, G., Widom, J.: Simrank: a measure of structural-context similarity. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02), pp. 538–543. Edmonton, Alberta (2002)
25. Kelly, D., Cushing, A., Dostert, M., Niu, X., Gyllstrom, K.: Effects of popularity and quality on the usage of query suggestions during information search. In: Proceedings of the 28th International Conference on Human Factors in Computing Systems(CHI'10), pp. 45–54. Atlanta, Georgia (2010)
26. Li, L., Otsuka, S., Kitsuregawa, M.: Query recommendation using large-scale web access logs and Web page archive. In: Proceedings of 19th International Conference on Database and Expert Systems Applications (DEXA'08), pp. 134–141. Turin, Italy (2008)
27. Li, L., Otsuka, S., Kitsuregawa, M.: Finding related search engine queries by Web community based query enrichment. World Wide Web **13**(1–2), 121–142 (2010)
28. Li, L., Yang, Z., Liu, L., Kitsuregawa, M.: Query-url bipartite based approach to personalized query recommendation. In: Proceedings of the 23rd AAAI Conference on Artificial Intelligence,(AAAI'08), pp. 1189–1194. Chicago, Illinois (2008)
29. Lin, J.: Divergence measures based on the shannon entropy. IEEE Trans. Inf. Theory **37**(1), 145–151 (1991)
30. Ma, H., Lyu, M.R., King, I.: Diversifying query suggestion results. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI'10). Atlanta, Georgia (2010)
31. 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. Napa Valley, California (2008)
32. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)

33. Mei, Q., Zhou, D., Church, K.W.: Query suggestion using hitting time. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, (CIKM'08), pp. 469–478. Napa Valley, California (2008)
34. Pereira, F.C.N., Tishby, N., Lee, L.: Distributional clustering of English words. In: Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL'93), pp. 183–190 (1993)
35. Ravid, G., Rafaeli, S.: Popularity and findability through log analysis of search terms and queries: the case of a multilingual public service web site. IEEE Trans. Inf. Theory **33**(5), 567–583 (2007)
36. Salton, G., Buckley, C.: Improving retrieval performance by relevance feedback. J. Am. Soc. Inf. Sci. **41**(4), 288–297 (1990)
37. Shi, X., Yang, C.C.: Mining related queries from web search engine query logs using an improved association rule mining model. J. Am. Soc. Inf. Sci. Technol. **58**(12), 1871–1883 (2007)
38. Song, Y., wei He, L.: Optimal rare query suggestion with implicit user feedback. In: Proceedings of the 19th International Conference on World Wide Web (WWW'10), pp. 901–910. Raleigh, North Carolina (2010)
39. Sun, J., Qu, H., Chakrabarti, D., Faloutsos, C.: Neighborhood formation and anomaly detection in bipartite graphs. In: Proceedings of the 5th IEEE International Conference on Data Mining (ICDM'05), pp. 418–425. Houston, Texas (2005)
40. Sun, R., Ong, C.H., Chua, T.S.: Mining dependency relations for query expansion in passage retrieval. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06), pp. 382–389. Seattle, Washington (2006)
41. Vechtomova, O., Wang, Y.: A study of the effect of term proximity on query expansion. J. Inf. Sci. **32**(4), 324–333 (2006)
42. Voorhees, E.M.: Query expansion using lexical-semantic relations. In: Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94), pp. 61–69. Dublin, Ireland (1994)
43. Wen, J.R., Nie, J.Y., Zhang, H.: Query clustering using user logs. ACM Trans. Inf. Sys. **20**(1), 59–81 (2002)
44. Xu, J., Croft, W.B.: Improving the effectiveness of information retrieval with local context analysis. ACM Trans. Inf. Sys. **18**(1), 79–112 (2000)
45. Yang, J.M., Cai, R., Jing, F., Wang, S., Zhang, L., Ma, W.Y.: Search-based query suggestion. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, (CIKM'08), pp. 1439–1440. Napa Valley, California (2008)
46. Zhu, Y., Gruenwald, L.: Query expansion using Web access log files. In: Proceedings of the 16th International Conference on Database and Expert Systems Applications (DEXA'05), pp. 686–695. Copenhagen, Denmark (2005)