A

# Web Query Reformulation via Joint Modeling of Latent Topic Dependency and Term Context

LIDONG BING, The Chinese University of Hong Kong
WAI LAM, The Chinese University of Hong Kong
TAK-LAM WONG, The Hong Kong Institute of Education
SHOAIB JAMEEL, The Chinese University of Hong Kong

An important way to improve users' satisfaction in Web search is to assist them to issue more effective queries. One such approach is query reformulation, which generates new queries according to the current query issued by users. A common procedure for conducting reformulation is to generate some candidate queries first, and then a scoring method is employed to assess these candidates. Currently, most of the existing methods are context-based. They rely heavily on the context relation of terms in the history queries and cannot detect and maintain the semantic consistency of queries. In this paper, we propose a graphical model to score queries. The proposed model exploits a latent topic space, which is automatically derived from the query log, to detect semantic dependency of terms in a query and dependency among topics. Meanwhile, the graphical model also captures the term context in the history query by skip-bigram and n-gram language models. In addition, our model can be easily extended to consider users' history search interests when we conduct query reformulation for different users. In the task of candidate query generation, we investigate a social tagging data resource, namely, Delicious bookmark, to generate addition and substitution patterns which are employed as supplements to the patterns generated from query log data.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query Formulation, Search Process*; I.5.1 [**Pattern Recognition**]: Models—*Statistical*

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Web query reformulation, graphical model, social tagging, query log

## 1. INTRODUCTION

In a typical Web search scenario, a user issues keyword-based queries to a search engine and the search engine retrieves a list of matched Web documents. The interface between the user and the search engine is the keyword query which is summarized by the user according to his or her information need. Inevitably, the issued keywords may be inappropriate to convey the information need or unable to perfectly match the desirable documents because of vocabulary gap [Gao et al. 2013; Weerkamp et al. 2012]. To tackle this issue, extensive research works have been conducted to help users issue

The work described in this paper is substantially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: CUHK413510) and the Direct Grant of the Faculty of Engineering, CUHK (Project Code: 4055034). This work is also affiliated with the CUHK MoE-Microsoft Key Laboratory of Human-centric Computing and Interface Technologies.
We acknowledge the suggestions received from Dr. Xuanhui Wang on the baseline method implementation.
Author's addresses: Lidong Bing, Wai Lam, and Shoaib Jameel, Dept. of Systems Engg. & Engg. Management, The Chinese University of Hong Kong. TAK-LAM Wong, Department of Mathematics and Information Technology, The Hong Kong Institute of Education.
Contact author's e-mail address: Lidong Bing, ldbing@se.cuhk.edu.hk.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

more effective queries. There are two major directions in this research, namely, query suggestion (recommendation) and query reformulation (refinement, modification, or rewriting).

In query suggestion, history queries are recommended to users according to the relation closeness with the current query issued by users. This suggestion can be performed based on history click graph [Deng et al. 2009; Mei et al. 2008], query flow graph [Boldi et al. 2008, 2009], or sequential query prediction [He et al. 2009]. However, it cannot provide new queries which are not contained in the query log data. On the other hand, in query reformulation, the original query is modified in a more delicate manner by exploiting the term dependency information in history queries or the implicit feedback from the retrieved results [Jones et al. 2006; Wang and Zhai 2008; He and Ounis 2007; Joachims et al. 2007]. Therefore, the generated query may be an existing query in the query log, or a totally new query. In the query log dataset used in our experiments, about 70% of the satisfactory queries (the queries that cause at least one URL clicked and locate in the end of search sessions with multiple queries) are previously unseen queries, which are generated by users with reformulation operations.

In this paper, we focus on the latter task, namely query reformulation in Web search scenario. This task needs to consider different features such as semantic consistency within different queries and query term substitution. These are not emphasized in the query reformulation of typical IR setting, where query expansion, term reweighing and ranking, the number of expansion terms, prior term necessity and mismatch prediction, extracting phrases or proximity features from the original queries are the major investigated problems [Billerbeck et al. 2003; Xue and Croft 2013; Bendersky et al. 2011; Huang et al. 2008; Sun et al. 2006; Crouch and Yang 1992; Qiu and Frei 1993; Xu and Croft 2000; Cao et al. 2008a; Ogilvie et al. 2009; Collins-Thompson and Callan 2005; Zhao and Callan 2012; Lee et al. 2009]. According to the operations performed, there are three major types of Web query reformulation, namely, substitution, addition, and deletion [Bing et al. 2011; Jones et al. 2006; Wang and Zhai 2008; Jones and Fain 2003]. A common procedure of carrying out reformulation is to generate some candidate queries first, and then a scoring method is used to assess the quality of these candidates [Bing et al. 2011; Wang and Zhai 2008]. We may directly consider any possible candidate queries in a single scoring stage. However, this manner will be extremely time consuming given that the number of possible candidate queries is very large.

Wang and Zhai proposed a contextual model for tackling the problem of query reformulation in Web search by investigating the context similarity of terms in history queries [Wang and Zhai 2008]. Then, two terms in a pair with similar contexts are used to substitute each other in new query generation. A context-based translation model is employed to score the generated candidate queries by substitution or addition. Jones et al. [2006] employed the hypothesis likelihood ratio to identify those highly related query phrase or term pairs in all user sessions. Then, these pairs are utilized to generate new queries for the input query. All the above methods make use of query log, exploit the context information to generate reformulation patterns, and score candidate queries. Table I shows top 15 output queries of an existing context-based method, which is used as the comparison baseline in our experiments, for the original query “wrestling ring instructions”. We can observe that the context-based method has shortcomings as discussed below.

The first shortcoming is that the context-based method cannot detect and maintain consistency of semantic meaning when scoring a new query. For example, as shown in Table I, a good output query is “wrestling ring manual”, but it is ranked 13th. The reason is that the scoring function highly depends on the co-occurrence of the terms in the contexts of history queries. The phrase “championship ring” is quite popular in

Table I. Top 15 results given by a context-based method for the original query "wrestling ring instructions". "◇" represents the unchanged original terms.

| Rank | Candidate query | Rank | Candidate query | Rank | Candidate query |
|---|---|---|---|---|---|
| 1 | ◇ rings ◇ | 6 | ◇ ◇ poncho | 11 | ◇ ◇ steps |
| 2 | championship ◇ ◇ | 7 | ◇ ◇ stitches | 12 | ◇ ◇ instruction |
| 3 | ◇ ◇ instrutions | 8 | ◇ ◇ scarf | 13 | ◇ ◇ manual |
| 4 | ◇ ◇ instuctions | 9 | ◇ ◇ loom | 14 | championships ◇ ◇ |
| 5 | ◇ ◇ afghans | 10 | ◇ ◇ afghan | 15 | ◇ blaylock ◇ |

query log owing to NBA. Consequently, "championship ring instructions" is assigned a high score although the semantic meaning of the terms in this reformulated query is not consistent.

The second shortcoming is that the candidate term generation process purely based on query context is easily affected by noise when it is dealing with ambiguous terms. For example, considering the term "instructions", due to the fact that "instructions" has very diverse contexts in history queries, a lot of noise is involved in its candidate list such as "afghans", "poncho", etc. Furthermore, typos such as "instrutions" and "instuctions" also degrade the performance when dealing with the typo-prone terms such as "instructions".

The third shortcoming is that when an unfamiliar or rare query is issued by a user, existing context-based methods cannot work well too. The main reason for ineffective handling of unfamiliar queries by existing context-based methods is that these methods mine the history queries to obtain hints for reformulating the issued query. When there is no or very little related information, the performance will inevitably be affected. One reason is that it is not easy to figure out the candidate terms to reformulate the query. Another reason is that the statistical information may not be reliable to differentiate the quality of different candidate queries. An investigation on the percentage of unfamiliar queries will be given in Section 8.3.

The fourth shortcoming is that existing methods are unable to take the interest profile of a user into consideration in the scoring of candidate queries. Thus, the same ranking list of the candidate queries is returned to different users. For the example given in Table I, if it is known that the current user is probably a businessman manufacturing boxing equipment from his history queries, we should give higher score to the candidate query "wrestling ring manual" since it is related to wrestling and also able to satisfy the "manual" information need of the manufacturer.

In summary, the term-context-based methods have limitations in both major tasks of query reformulation, namely, candidate query generation and candidate query scoring. In candidate generation, noise will be involved as exemplified in Table I for the terms with diverse meaning. In addition, the candidates cannot be generated properly when the context information for some terms is sparse. In query scoring, the context-based methods cannot detect and maintain the semantic consistency of the terms in the query. Furthermore, if the query is not familiar, the scoring task becomes even harder. These methods are also not able to take into account the personal interests of users.

In this paper, we propose a graphical model for scoring candidate queries, which has several appealing characteristics. First, the graphical model exploits a latent topic space, which is automatically derived from the query log, to detect semantic dependency of terms in a query and dependency among topics. It is achieved with the sequence of hidden nodes representing the latent topics of the corresponding terms. This characteristic also enables the graphical model to conduct reliable scoring for the candidate queries of an unfamiliar query, in which the term contexts may be too sparse to conduct reliable scoring. Second, the graphical model is able to capture a rich term context including skip-bigram and n-gram language models. This is achieved through

the sequence of nodes representing the terms in a candidate query. Third, our model can be easily extended to take the users' history search interests into consideration when it conducts query reformulation for different users. The above characteristics enable our graphical model to conduct better and more flexible candidate query scoring, so that the overall performance of the proposed query reformulation framework is upgraded.

To tackle the task of candidate generation, we investigate a social tagging data resource, namely, Delicious bookmark, to generate addition and substitution patterns which are employed as supplements to the patterns generated from query log data. Social tagging is an important application and generates a valuable semantic repository, whose potential in enhancing Web search performance has been studied in personalized search [Cai and Li 2010] and query suggestion [Guo et al. 2010]. In a social tagging system, the users store lists of Internet resources (e.g. URLs, photos, and videos) that they find useful, and these lists are accessible to the public. Currently, millions of users are involved in some famous social tagging systems such as Delicious[1] and Flickr[2]. They tag and share their favorite online resources. Moreover, the users regularly maintain their own tagging data and correct some typos as well as inaccurate tags, so social tagging data contains less noise than a query log.

The proposed framework in this paper substantially extends our previous work [Bing et al. 2011]. Although [Bing et al. 2011] also investigated latent topic with a graphical model for a candidate query, it is restricted to capture bigram relations in the term sequence of queries. In this paper, our proposed graphical model considers a rich term context including skip-bigram and n-gram language models. We also investigate the addition pattern generation from social tagging data and query log data so as to conduct term addition operation for reformulating a query, which is not considered in our previous work. In addition, our previous work is not able to conduct personalized scoring for different users with different interests that are revealed by their search history. Furthermore, more extensive experiments are conducted in this paper, including the analysis of user behavior in query reformulation, the impact of query length on reformulation performance, the effectiveness verification of query reformulation in Web search, the impact of topic number on reformulation performance, etc.

The remainder of the paper is organized as follows. The overview of our framework is presented in Section 2. Our graphical-model-based candidate query scoring method is described in Section 3. The latent topic detection and the parameter learning for the graphical model are presented in Section 4. After that, our personalized scoring method with derived user interest profile is presented in Section 5. The generation of candidate patterns from social tagging data and query log is described in Section 6. The experimental setup and results are given in Sections 7, 8 and 9, respectively. After some related works are reviewed in Section 10, we conclude the paper and provide some possible directions for the future work in Section 11.

## 2. OVERVIEW OF OUR FRAMEWORK

As previously mentioned, the problem of query reformulation involves two major tasks. The first task is to generate some candidate queries for the original query via operations such as term substitution and term addition. The second task is to evaluate the generated candidate queries and rank them according to the predicted quality.

In the first task, we focus on two types of query reformulation operations, namely, term substitution and term addition. To generate candidate queries by substitution, we first determine a set of possible substitution terms for each term of the original

[1] http://www.delicious.com/
[2] http://www.flickr.com

query. These substitution terms are utilized to substitute one or more terms of the original query. To generate candidate queries by addition, we first determine a set of possible addition terms according to the existing terms of the original query. Then, one or a few of these terms are added into the original query. As discussed in Section 1, the noise and the lack of sufficient context information are the major problems in using query log to generate candidate addition and substitution patterns.

We investigate a social tagging data resource, namely, Delicious bookmark, to generate addition and substitution patterns, which are employed as supplements to the patterns generated from query log data. Based on the observation that each bookmark of a URL given by an individual user can be regarded as a high-level abstraction of the information specified by the URL, as well as an expression of the user's specific information need, we can extract some addition patterns by considering the co-occurrence of tags among individual bookmarks. For example, {video, editing, tutorial} is a bookmark that is often used by different users to summarize video editing related URLs. From such tagging behaviors, the addition term patterns ("editing", "tutorial") and ("video", "editing") can be discovered. Based on the observation that the terms with similar meaning are often used to tag the same resource by different users, we can extract some substitution patterns. For example, the substitution term pattern ("film", "movie") is often observed in the same tag set of movie related URLs. Therefore, if two tags co-occur frequently in different URLs' tag sets, they are very likely to have closely related meaning. A filtering method is proposed with a delicately designed similarity function to filter out noisy substitution pairs such as "north" and "carolina", "free" and "music".

In the second task, we propose a graphical model to evaluate the quality of the candidate queries generated in the first step. The proposed model exploits a latent topic space, which is automatically derived from the query log, to detect semantic dependency of terms in a query and dependency among topics. There are two groups of variables in the graphical model. One group corresponds to the terms of a query, and the other group corresponds to the topical concepts of the terms. Unlike the purely context-based methods, we evaluate the quality of queries by simultaneously taking into account the semantic consistency of the terms in the latent topic space and the term context of queries. We first derive a set of pseudo-documents from the history query log data. Then, these pseudo-documents are used to generate a latent topic space. The graphical model can capture the concept consistency of two consecutive terms with the hidden concept variables. Therefore, even when the term context information is not sufficient for the unfamiliar queries in the query log, our model can still give reliable score by considering the semantic consistency of the consecutive terms in the latent topic space. To capture the term context quality of a candidate query, our graphical model incorporates skip-bigram and n-gram language models of the search queries with the variables corresponding to the terms. Another characteristic of our scoring model is that it can generate a tailor-made ranking of the candidate queries for a user according to his or her interest. To do so, a topic-based interest profile of a user is automatically derived from the user's history queries in the latent topic space generated above. Then, the personalized scoring on the candidate queries is conducted by incorporating the user's interest profile into the graphical model.

We will first present in Sections 3, 4, and 5 the details of the graphical-model-based method for candidate query scoring since it is a major component involving some technical merits. Then, we present the candidate generation step including the generation of addition patterns and substitution patterns in Section 6.

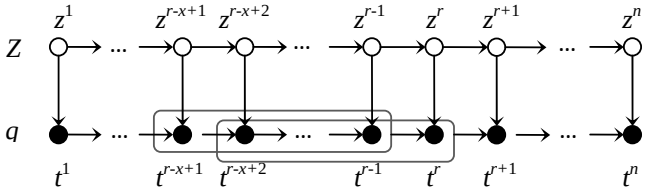


Fig. 1. The graphical model for scoring a query.

## 3. GRAPHICAL-MODEL-BASED SCORING

### 3.1. The Graphical Model

A query $q$ is composed of a sequence of terms, which can be denoted by $q: t^1 \cdots t^n$ (or $t^{1:n}$ for short), where the superscripts indicate the position of the terms, and $t^r (1 \leq r \leq n)$ may take any term in the vocabulary. We propose a graphical model which considers the dependency among latent semantic topics of the terms. Fig. 1 depicts the design of the graphical model. The query terms, i.e. $t^r$, are observable and represented by shaded nodes. The latent topics, denoted by $z^r$, of the corresponding terms are unobservable and represented by unshaded nodes. The detection of latent topics will be discussed later. Each term depends on its corresponding latent topic and the preceding $x-1$ terms, where $x$ is named *window size* of the model. The probability of obtaining the term $t^r$ given its latent topic $z^r$ and the preceding term sequence $t^{r-x+1:r-1}$ is denoted by $P(t^r|z^r, t^{r-x+1:r-1})$, which is independent of the position $r$. Each topic depends on its preceding topic. The probability of obtaining the topic $z^r$ given its preceding topic $z^{r-1}$ is denoted by $P(z^r|z^{r-1})$, which is also independent of the position $r$. When $r=1$, we have $P(z^1|z^0) = P(z^1)$, where $P(z^1)$ denotes the probability of obtaining a particular topic at the beginning of the latent topic sequence.

When we score a query with this graphical model, the latent topic sequence of the query is used as hidden evidence to guide the semantic consistency assessment. For queries with good semantic consistency, their corresponding topic sequences will increase the score of their quality. For these queries, the transition probability between the consecutive topics in the topic sequence is higher. As a result, the consecutive terms tend to coexist in the same topic. Let $QS(q)$ denote the score of a query $q: t^{1:n}$, which is calculated as:

$$QS(q) = P(t^{1:n}) = \sum_{z^r \in Z, 1 \leq r \leq n} P(t^{1:n}, z^{1:n}), \tag{1}$$

where $Z$ is the set of possible latent topics. We marginalize over all possible latent semantic topic sequences to calculate the overall probability of generating the query $q$. Based on the structure of the designed graphical model, $P(t^{1:n}, z^{1:n})$ can be calculated as in Equation (2) by the chain rule:

$$P(t^{1:n}, z^{1:n}) = \prod_{1 \leq r \leq n} P(z^r|z^{r-1}) P(t^r|z^r, t^{r-x+1:r-1}). \tag{2}$$

If $r-x+1 < 1$, we have $P(t^r|z^r, t^{r-x+1:r-1}) = P(t^r|z^r, t^{1:r-1})$. To capture the impact of the preceding terms $t^{r-x+1:r-1}$ on the probability of obtaining $t^r$, we design two strategies, namely, n-gram strategy and skip-bigram strategy, which will be discussed in Section 4.

This graphical model shares some resemblance to Hidden Markov Models (HMM). However, one major difference is that there are dependencies among the terms, which capture their co-occurrence. This graphical model can also be viewed as a kind of Bayesian network. The difference between our model and general Bayesian networks

is that our model aims at calculating a marginalized joint probability of an observation sequence, i.e. a query, while, general Bayesian networks usually support the calculation of some unobservable events given the observable ones in the inference. Considering the characteristics of our model, we develop a more efficient algorithm to train the model and conduct inference to obtain the score for a candidate query.

### 3.2. Scoring Algorithm

As previously mentioned, the scoring function needs to instantiate the latent topic sequence in the graphical model depicted in Fig. 1. The possible latent topic paths compose a trellis structure. To marginalize the joint probability $P(t^{1:n}, z^{1:n})$ over all possible paths, we employ a dynamic programming method, known as the forward algorithm. The forward variable $\alpha_r(i)$ is defined as:

$$\alpha_r(i) \equiv P(t^{1:r}, z^r = z_i), \tag{3}$$

which is the score of the partial query $t^1 \cdots t^r$ with the topic $z_i$ at the position $r$. Note that subscripts are employed to indicate instantiated topics and terms. Each $z_i$ belongs to the topic set $Z$. When $r = 1$, $\alpha_1(i) = P(z^1 = z_i)P(t^1|z^1 = z_i)$ is the probability that $t^1$ is the first term typed by a user. The recursive calculation of $\alpha$ is:

$$\alpha_r(i) = \Big[ \sum_{z_j \in Z} \alpha_{r-1}(j) P(z^r = z_i | z^{r-1} = z_j) \Big] P(t^r | z^r = z_i, t^{r-x+1:r-1}), \tag{4}$$

where $P(z^r = z_i|z^{r-1} = z_j)$ and $P(t^r|z^r = z_i, t^{r-x+1:r-1})$ are independent of the position $r$, and equal to $P(z_i|z_j)$ and $P(t^r|z_i, t^{r-x+1:r-1})$, respectively. Finally, the score of the query $q$, denoted by $QS(q)$, can be calculated by summing over all possible $z_i$ for $\alpha_n(i)$:

$$QS(q) = P(t^{1:n}) = \sum_{z_i \in Z} \alpha_n(i). \tag{5}$$

To conduct the scoring as discussed above, we need three types of model parameter, namely, $P(z_i)$, $P(z_i|z_j)$ and $P(t_{l_I}|z_j, t_{l_{I-x+1}:l_{I-1}})$ $(1 \leq I \leq x)$, where $t_{l_{I-x+1}:l_{I-1}}$ is a sequence of terms and $t_{l_y}$ indicates any possible term when $y > 0$. If $I = 1$, $P(t_{l_I}|z_j, t_{l_{I-x+1}:l_{I-1}}) = P(t_{l_I}|z_j)$. If $2 \leq I \leq x$, $P(t_{l_I}|z_j, t_{l_{I-x+1}:l_{I-1}}) = P(t_{l_I}|z_j, t_{l_1:l_{I-1}})$.

This scoring function can be effectively used to compare the quality of candidate queries in the same length. Such capability is already useful in practice since, as shown by some statistics in Section 7.3, about 31% of the successful query reformulations carried out by the users perform substitution operation on one term of the original queries. Nevertheless, we further extend the scoring function to compare the quality of candidate queries in different lengths in Section 5.1.

## 4. MODEL TRAINING

To learn the model parameters, the history query log containing a set of history queries is employed as the training data. Let $\theta$ denote the parameters of our graphical model, and $Q$ denote the training query set. To learn the parameters, we maximize the following log likelihood function:

$$\ell(\theta; Q) = \sum_{q \in Q} \big( \log \sum_{z^r \in Z, 1 \leq r \leq |q|} P(t^{1:|q|}, z^{1:|q|}; \theta) \big), \tag{6}$$

where $|q|$ denotes the length of $q$. We describe the details of the learning process in the following subsections. The latent topic space is first detected from the history log data in Section 4.1. The detected latent sematic topics are transferred to initialize the model parameters in Section 4.2. In Section 4.3, we derive our training algorithm by

refining the standard Baum-Welch algorithm [Baum et al. 1970], which is a special case of the Expectation Maximization (EM) algorithm.

### 4.1. Latent Topic Detection

A typical format of the records in query log can be represented as a triple $(query, clicked\_url, time)$. One direct way of using query log data for latent topic analysis is to treat each $clicked\_url$ as a single document unit. This way suffers from the data sparsity problem since most of URLs only involve very small number of queries. Therefore, we aggregate all the queries related to the same host together, and construct one pseudo-document for each host, denoted by $\mathcal{H}$. For example, the pseudo-document of "www.mapquest.com" consists of the queries such as "mapquest", "travel map", "driving direction", etc. Some general Web sites such as "en.wikipedia.org" and "www.youtube.com" are not suitable for latent topic analysis, because they involve large amount of queries which cover very diverse aspects. To tackle this problem, we sort the host pseudo-documents in descending order according to the number of distinct query terms they have, and the top ones are eliminated.

We employ the standard Latent Dirichlet Allocation (LDA) algorithm [Blei et al. 2003] to conduct the latent semantic topic analysis on the collection of pseudo-documents. In particular, GibbsLDA[3] package is used to generate a set of latent topics $Z = \{z_1,\ z_2,\ \ldots,\ z_{|Z|}\}$. Each topic $z_i$ is associated with a multinomial distribution of terms, denoted by $P(t_k|z_i)$, where $t_k$ is a term.

### 4.2. Parameter Initialization via Latent Topic Transfer

We transfer the latent topics discovered above to initialize the model parameters in the beginning of training phase. $\hat{P}(z_i)$ and $\hat{P}(z_j|z_i)$ are initially estimated as:

$$\hat{P}(z_i) = \frac{1}{|Z|}, \tag{7}$$

$$\hat{P}(z_j|z_i) = \frac{\exp\left(-KLD(z_j||z_i)\right)}{\sum_{z_k \in Z} \exp\left(-KLD(z_k||z_i)\right)}, \tag{8}$$

where $KLD(z_j||z_i) = \sum_{t_k} P(t_k|z_j) \log \frac{P(t_k|z_j)}{P(t_k|z_i)}$ is the Kullback-Leibler divergence between two distributions $P(\cdot|z_i)$ and $P(\cdot|z_j)$. When $z_i$ and $z_j$ are highly related topics, the value of $\hat{P}(z_j|z_i)$ is large. As previously mentioned, we design two strategies to capture the impact of the preceding terms $t^{r-x+1:r-1}$ on $t^r$. Accordingly, we have two different ways to estimate the initial parameters of $\hat{P}(t_{l_I}|z_j, t_{l_1:l_{I-1}})$.

*N-gram Initialization*. $\hat{P}(t_{l_I}|z_j, t_{l_1:l_{I-1}})$ can be estimated with the maximum likelihood estimation from the $I$-grams ($2 \leq I \leq x$) appearing in the history queries. This initialization is denoted by $\hat{P}_1(t_{l_I}|z_j, t_{l_1:l_{I-1}})$ and calculated as:

$$\hat{P}_1(t_{l_I}|z_j, t_{l_1:l_{I-1}}) = \frac{\sum_{q \in Q} w(q) \sum_{I \leq r \leq n} \mathbf{I}_{\{z(t^r|q)=z_j\}} \prod_{0 \leq k \leq I-1} \mathbf{I}_{\{t^{r-k}=t_{l_{I-k}}\}}}{\sum_{q \in Q} w(q) \sum_{I \leq r \leq n} \mathbf{I}_{\{z(t^r|q)=z_j\}} \prod_{1 \leq k \leq I-1} \mathbf{I}_{\{t^{r-k}=t_{l_{I-k}}\}}}, \tag{9}$$

where $\mathbf{I}_{\{\}}$ is an indicator function, $z(t^r|q)$ is the inferred topic of $t^r$ in $q$ with the inference algorithm of LDA, and $w(q)$ is the weight of $q$. $w(q)$ is calculated as:

$$w(q) = F(q) + F_{click}(q) + F_{click \wedge end}(q), \tag{10}$$

[3] http://gibbslda.sourceforge.net/

where $F(q)$ is the frequency of $q$ issued, $F_{click}$ is the frequency of $q$ resulting in at least one URL clicked, and $F_{click \wedge end}$ is the frequency of $q$ resulting in at least one URL clicked and meanwhile $q$ is the last query of the corresponding search session.

*Skip-bigram Initialization*. Another way of initializing $\hat{P}(t_{l_I}|z_j, t_{l_1:l_{I-1}})$ is to employ the skip-bigram model in the history queries. This initialization is denoted by $\hat{P}_2(t_{l_I}|z_j, t_{l_1:l_{I-1}})$ and calculated as:

$$\hat{P}_2(t_{l_I}|z_j, t_{l_1:l_{I-1}}) = \sum_{1 \le p \le I-1} \lambda_p \hat{P}(t_{l_I}|z_j, t_{l_{I-p}}), \tag{11}$$

where $\sum_{1 \le p \le I-1} \lambda_p = 1$ and $\lambda_p$ is calculated as:

$$\lambda_p = \frac{1/p}{\sum_{1 \le p' \le I-1} 1/p'}. \tag{12}$$

$\hat{P}(t_{l_I}|z_j, t_{l_{I-p}})$ is the probability of generating the term $t_{l_I}$ given that its topic is $z_j$ and its $p$-preceding term is $t_{l_{I-p}}$. $\hat{P}(t_{l_I}|z_j, t_{l_{I-p}})$ can be estimated with the maximum likelihood estimation:

$$\hat{P}(t_{l_I}|z_j, t_{l_{I-p}}) = \frac{\sum_{q \in Q} w(q) \sum_{I \le r \le n} \mathbf{I}_{\{z(t^r|q)=z_j \wedge t^r=t_{l_I} \wedge t^{r-p}=t_{l_{I-p}}\}}}{\sum_{q \in Q} w(q) \sum_{I \le r \le n} \mathbf{I}_{\{z(t^r|q)=z_j \wedge t^{r-p}=t_{l_{I-p}}\}}}. \tag{13}$$

$\hat{P}_2(t_{l_I}|z_j, t_{l_1:l_{I-1}})$ is the weighed combination of $\hat{P}(t_{l_I}|z_j, t_{l_{I-p}})$'s. The rationale of skip-bigram strategy is that data sparsity may result in zero frequency of the $I$-gram pattern $t_{l_1:l_I}$, which results in $\hat{P}_1(t_{l_I}|z_j, t_{l_1:l_{I-1}}) = 0$. $\hat{P}_2(t_{l_I}|z_j, t_{l_1:l_{I-1}})$ can help overcome this problem and provide a reasonable calculation of $\hat{P}(t_{l_I}|z_j, t_{l_1:l_{I-1}})$. When $I = 2$, the above two strategies result in the same strategy, i.e., bigram.

### 4.3. Model Parameter Training

To continue the training, we refine the standard Baum-Welch algorithm [Baum et al. 1970]. Recall that we utilize the history queries as the training sequences. Given a set of history queries in the form of search sessions, we preprocess them so that the weight, i.e.,$w(q)$, of distinct queries is aggregated and collected. Since we assume that one query is independent of others, the intermediate calculation can be illustrated with a single query. Then, the intermediate results of individual queries are aggregated to obtain the final updating formulas with the similar way in [Levinson et al. 1983].

For a query $q : t^{1:n}$, we define backward variable $\beta_r(i)$:

$$\beta_r(i) \equiv P(t^{r+1:n}|z^r = z_i, t^{r-x+2:r}), \tag{14}$$

which is the score of the partial query $t^{r+1:n}$ given the topic at the position $r$, namely $z^i$, and the sequence of the preceding $x-1$ terms, namely $t^{r-x+2:r}$. We define $\beta_n(i) = 1$. The recursive calculation of $\beta$ is:

$$\beta_r(i) = \sum_{z_j \in Z} P(z_j|z_i) P(t^{r+1}|z_j, t^{r-x+2:r}) \beta_{r+1}(j). \tag{15}$$

Then, the joint probability of a query $q$ and $z^r = z_i$ can be derived as:

$$\begin{aligned} P(q, z^r = z_i) &= P(t^{1:r-x+1}, t^{r+1:n}|t^{r-x+2:r}, z^r = z_i) P(t^{r-x+2:r}, z^r = z_i) \\ &= P(t^{1:r-x+1}, t^{r-x+2:r}, z^r = z_i) P(t^{r+1:n}|t^{r-x+2:r}, z^r = z_i) \\ &= P(t^{1:r}, z^r = z_i) P(t^{r+1:n}|t^{r-x+2:r}, z^r = z_i) \\ &= \alpha_r(i) \beta_r(i). \end{aligned} \tag{16}$$

$P(t^{1:r-x+1}|t^{r-x+2:r}, z^r = z_i)$ and $P(t^{r+1:n}|t^{r-x+2:r}, z^r = z_i)$ are independent, since $t^{r+1}$ depends at most $(x-1)$-preceding terms. $P(q)$ can also be computed by $\sum_i \alpha_r(i)\beta_r(i)$.

To derive the iterative updating formulas, we introduce two conditional probabilities. $P(z^r = z_i|q)$ is the probability that $z^r$ takes the value $z_i$ in the query $q$, which can be calculated as:

$$P(z^r = z_i|q) = \frac{\alpha_r(i)\beta_r(i)}{\sum_{z_j \in Z} \alpha_r(j)\beta_r(j)} = \frac{\alpha_r(i)\beta_r(i)}{P(q)}. \tag{17}$$

$P(z^r = z_i, z^{r+1} = z_j|q)$ is the joint probability of $z^r = z_i$ and $z^{r+1} = z_j$ in a given $q$, which can be calculated as:

$$P(z^r = z_i, z^{r+1} = z_j|q) = \frac{\alpha_r(i)P(z_j|z_i)P(t^{r+1}|z_j, t^{r-x+2:r})\beta_{r+1}(j)}{P(q)} \tag{18}$$

The next step is to aggregate the intermediate results and obtain the final updating formulas. The updating formulas of $P'(z_i)$ and $P'(z_j|z_i)$ are straightforward and they are given as follows:

$$P'(z_i) = \frac{\sum_{q \in Q} w(q)P(z^1 = z_i|q)}{\sum_{q \in Q} w(q)}, \tag{19}$$

$$P'(z_j|z_i) = \frac{\sum_{q \in Q} w(q) \sum_{r=1}^{n-1} P(z^r = z_i, z^{r+1} = z_j|q)}{\sum_{q \in Q} w(q) \sum_{r=1}^{n-1} P(z^r = z_i|q)}. \tag{20}$$

Recall that we design two strategies to capture the impact of the preceding terms $t^{r-x+1:r-1}$ on $t^r$. Accordingly, we have two different updating formulas for $P'(t_{l_I}|z_j, t_{l_1:l_{I-1}})$.

*N-gram Updating Formula*. For each $2 \leq I \leq x$, the updating formula of n-gram strategy is as follows:

$$P'_1(t_{l_I}|z_j, t_{l_1:l_{I-1}}) = \frac{\sum\limits_{q \in Q} w(q)\left(\sum\limits_{I \leq r \leq n} P(z^r = z_j|q) \prod\limits_{0 \leq k \leq I-1} \mathbf{I}_{\{t^{r-k} = t_{l_{I-k}}\}}\right)}{\sum\limits_{q \in Q} w(q)\left(\sum\limits_{I \leq r \leq n} P(z^r = z_j|q) \prod\limits_{1 \leq k \leq I-1} \mathbf{I}_{\{t^{r-k} = t_{l_{I-k}}\}}\right)}. \tag{21}$$

As previously mentioned, data sparsity may result in zero frequency of the $I$-gram pattern $t_{l_1:l_I}$. We employ the estimated universal topic distribution in Section 4.1 to smooth the above estimation:

$$\tilde{P}'_1(t_{l_I}|z_j, t_{l_1:l_{I-1}}) = \frac{\sum\limits_{q \in Q} w(q)\left(\sum\limits_{I \leq r \leq n} P(z^r = z_j|q) \prod\limits_{0 \leq k \leq I-1} \mathbf{I}_{\{t^{r-k} = t_{l_{I-k}}\}}\right) + \mu_1 P(t_{l_I}|z_j)}{\sum\limits_{q \in Q} w(q)\left(\sum\limits_{I \leq r \leq n} P(z^r = z_j|q) \prod\limits_{1 \leq k \leq I-1} \mathbf{I}_{\{t^{r-k} = t_{l_{I-k}}\}}\right) + \mu_1}, \tag{22}$$

where $\mu_1$ is the Dirichlet prior parameter.

*Skip-bigram Updating Formula*. For each $2 \leq I \leq x$, the updating formula of skip-bigram strategy is as follows:

$$P'_2(t_{l_I}|z_j, t_{l_1:l_{I-1}}) = \sum_{1 \leq p \leq I-1} \lambda_p \tilde{P}'(t_{l_I}|z_j, t_{l_{I-p}}), \tag{23}$$

where $\tilde{P}'(t_{l_I}|z_j, t_{l_{I-p}})$ is calculated as follows:

$$\tilde{P}'(t_{l_I}|z_j, t_{l_{I-p}}) = \frac{\sum_{q\in Q} w(q)\left(\sum_{I\le r\le n} \mathbf{I}_{\{t^r=t_{l_I}\}}\mathbf{I}_{\{t^{r-p}=t_{l_{I-p}}\}}P(z^r=z_j|q)\right) + \mu_1 P(t_{l_I}|z_j)}{\sum_{q\in Q} w(q)\left(\sum_{I\le r\le n} \mathbf{I}_{\{t^{r-p}=t_{l_{I-p}}\}}P(z^r=z_j|q)\right) + \mu_1}. \tag{24}$$

After one iteration, new parameters, namely, $P'(t_{l_I}|z_j, t_{l_{I-p}})$, $P'(z_j|z_i)$, and $P'(z_i)$, will be used in the next iteration. This iterative procedure will stop when a pre-defined condition is satisfied.

## 5. PRACTICALITY EXTENSION

To serve the purpose of query reformulation better, we investigate some extensions of the above graphical model for widening its suitability for different scenarios and applications.

### 5.1. Normalized Scoring

In some application scenarios, the candidate queries for the same input query may have different lengths. For example, some of them are generated by substitution operation on one term, while some others are generated by addition operation with one new term. To make the queries with different lengths comparable, normalization should be considered.

Suppose we want to compare two queries $q : t^{1:m}$ and $q' : t^{1:n}$, where $m < n$. Our strategy is to compute a normalized score of the longer query, i.e., $q'$, with the normalization factor $m$, i.e., the length of the shorter query $q$. Let $NQS(q'; m)$ denote the normalized score of $q'$ with factor $m$. This score is calculated as the average score of $n - m + 1$ term sequences of length $m$ embedding in $q'$:

$$NQS(q'; m) = \frac{1}{n-m+1}\sum_{b=1}^{n-m+1}\sum_{z^r\in Z, b\le r\le b+m-1} P(t^{b:b+m-1}, z^{b:b+m-1}; q'), \tag{25}$$

where $P(t^{b:b+m-1}, z^{b:b+m-1}; q')$ is the joint probability of the term sequence $t^{b:b+m-1}$ and the topic sequence $z^{b:b+m-1}$ given that $t^{b:b+m-1}$ is embedded in the query $q'$. $P(t^{b:b+m-1}, z^{b:b+m-1}; q')$ is calculated as:

$$P(t^{b:b+m-1}, z^{b:b+m-1}; q') = P(z^b|q')P(t^b|z^b)\prod_{r=b+1}^{b+m-1} P(z^r|z^{r-1})P(t^r|z^r, t^{r-x+1:r-1}). \tag{26}$$

Note that the preceding term sequence $t^{r-x+1:r-1}$ at most covers the term $t^b$ on the left.

For an original query $q : t^{1:m}$, its reformulated query $q'_s$ generated by substitution has the same length as $q$ and its score $QS(q'_s)$ is calculated with the marginal probability as given in Equation 1. The reformulated query $q'_a$ generated by addition is scored with:

$$QS(q'_a) = NQS(q'_a; m). \tag{27}$$

Thus, for the candidates $q'_s$'s and $q'_a$'s of $q$ generated from different operations, we are able to evaluate their quality and perform comparison.

### 5.2. Personalized Scoring and Interest Profile Generation

Different users have different personal preferences which are revealed by the fact that their queries concentrate on different topics. For example, teenagers usually issue

queries related to basketball, computer game, and so on. This personal interest is encapsulated in a topic-based profile in this paper. Then, the profile is taken into account in the evaluation of the candidate queries for the corresponding user.

Let $\Pi^u = \{\pi_1^u, \pi_2^u, \cdots, \pi_{|Z|}^u\}$ denote the topic-based interest profile of the user $u$, where $\pi_i^u = P(z_i|u)$ is the probability that the user $u$ prefers the topic $z_i$. Considering the probability constraint, we have $\sum_i \pi_i^u = 1$. Taking the interest profile into consideration, the satisfaction score of a query $q$ for $u$ is denoted by $QS(q;u)$, which is calculated as:

$$QS(q;u) = P(t^{1:n};\Pi^u) = \sum_{z^r \in Z, 1 \le r \le n} P(t^{1:n}, z^{1:n}; \pi_{z^1}^u), \tag{28}$$

where $P(t^{1:n}, z^{1:n}; \pi_{z^1}^u)$ is the joint probability of a term sequence $t^{1:n}$ and a topic sequence $z^{1:n}$ given that the user's preference on $z^1$ is $\pi_{z^1}^u$. In Equation (2), the probability of the first topic, i.e., $P(z^1)$, is pre-estimated with Equation (19). The same probability is employed in scoring the candidate queries for different users without considering the interest of the individuals. By revising Equation (2), $P(t^{1:n}, z^{1:n}; \pi_{z^1}^u)$ is calculated as in Equation (29):

$$P(t^{1:n}, z^{1:n}; \pi_{z^1}^u) = \pi_{z^1}^u P(t^1|z^1) \prod_{2 \le r \le n} P(z^r|z^{r-1}) P(t^r|z^r, t^{r-x+1:r-1}). \tag{29}$$

Note that the preference of $u$ on different topics is captured with $z^1$ in the beginning of the topic sequence, then the transition between the neighboring topics encodes this information among the entire topic sequence since the transition probability between the same topic is much higher than that of different topics. Thus, the preference score of $u$ on the query $q$ can be regarded as the probability that $q$ is issued by $u$ with his preference on different topics as $\Pi^u$.

The topic-based interest profile $\Pi^u$ of $u$ is automatically derived from the history queries of $u$. Specifically, we maximize the following log likelihood function:

$$\ell(\Pi^u; Q^u) = \sum_{q \in Q^u} \big( \log \sum_{z^r \in Z, 1 \le r \le |q|} P(t^{1:|q|}, z^{1:|q|}; \Pi^u) \big), \tag{30}$$

where $Q^u$ contains the history queries issued by $u$. Each query in $Q^u$ is associated with a time stamp indicating when it was issued. Therefore, the same query may appear multiple times with different time stamps.

We adopt a similar approach as for the training in Section 4 to estimate the interest profiles. Taking the interest profile of $u$ into account, we have:

$$\alpha_1(i;\Pi^u) = P(z_i|u) P(t^1|z^1 = z_i), \tag{31}$$

and

$$P(z^1 = z_i|q;\Pi^u) = \frac{\alpha_1(i;\Pi^u)\beta_1(i)}{P(q;\Pi^u)}. \tag{32}$$

The iterative updating formula for estimating $P(z_i|u)$ is as follows:

$$P'(z_i|u) = \frac{\sum_{q \in Q^u} k(q) P(z^1 = z_i|q;\Pi^u)}{\sum_{q \in Q^u} k(q)}, \tag{33}$$

where $k(q)$ is a Gaussian kernel measuring the freshness of query $q$:

$$k(q) = \exp\big(-\frac{d^2}{2\sigma^2}\big), \tag{34}$$

where $d$ indicates the age of $q$ measured by days, $\sigma$ controls the balance between older and fresher queries.

## 6. CANDIDATE PATTERN GENERATION

As investigated in previous works [Bing et al. 2011; Kraft and Zien 2004; Dang and Croft 2010; Craswell et al. 2013], external resources such as social tagging data and anchor text data are helpful in query reformulation. In this paper, we investigate a social tagging data resource, namely, Delicious bookmark, to generate addition and substitution patterns which are employed as supplements to the patterns generated from query log data.

### 6.1. Social-Tagging-Based Candidate Pattern Generation

*6.1.1. Addition Pattern.* A piece of bookmark, denoted by $\mathcal{B}_{lu} = \{t_1, t_2, \cdots\}$, is a set of tags which are given by a user $u$ to summarize the content of a URL $l$, such as {tutorial, java} and {news, cnn}. We mine the tag pairs that are frequently used together for generating addition patterns. For example, the term "tutorial" may be a good addition candidate for a query "java programming" to generate a reformulated query "java programming tutorial". Compared with the terms in queries, the assigned tags normally follow the natural word order less strictly. Therefore, syntagmatic and paradigmatic relations [Jacquemin 1999; Rapp 2002] are not reliable in social tagging scenario. We propose a mutual-information-based method to mine addition patterns from social tagging data.

On all $\mathcal{B}_{lu}$'s, the mutual information of two tags, $t_1$ and $t_2$, can be calculated as in Equation (35):

$$I_a(t_1, t_2) = \sum_{X_{t_1}, X_{t_2} \in \{0,1\}} P(X_{t_1}, X_{t_2}) \log \frac{P(X_{t_1}, X_{t_2})}{P(X_{t_1})P(X_{t_2})}, \tag{35}$$

where $X_{t_i}$ is a binary random variable indicating whether the term $t_i$ appears in a particular bookmark $\mathcal{B}_{lu}$. For example, $P(X_{t_1} = 1, X_{t_2} = 1)$ is the proportion of the bookmarks that contain $t_1$ and $t_2$ simultaneously. Because commonly used terms may be used to tag a lot of URLs such as "free" and "good". To cope with this problem, we adopt the normalized form of mutual information as the addition pattern mining score:

$$add(t_1, t_2) = \frac{I_a(t_1, t_2)}{\big[H(t_1) + H(t_2)\big]/2}, \tag{36}$$

where $H(t_i)$ is the entropy of $t_i$'s distribution among all the bookmarks, and calculated as in Equation (37):

$$H(t_i) = -\sum_{X_{t_i} \in \{0,1\}} P(X_{t_i}) \log P(X_{t_i}). \tag{37}$$

For a query $q: \ t^{1:n}$, we generate a set of addition candidates for each $t^r$ according to $add(t^r, t_i)$, which is denoted by $A(t^r)$. To generate candidate queries, we enumerate four positions for each candidate in $A(t^r)$, namely $t^{r-2}\_\_t^{r-1}$, $t^{r-1}\_\_t^{r}$, $t^{r}\_\_t^{r+1}$, and $t^{r+1}\_\_t^{r+2}$, where $t^i\_\_t^j$ denotes the position between $t^i$ and $t^j$, since the tags normally do not follow the *natural word order*.

*6.1.2. Substitution Pattern.* Our substitution candidate term extraction method is based on the observation that different users may use tags with similar meaning to describe the same resource. To capture this kind of co-occurrence, we aggregate all bookmarks given by different users to the same URL together, and compose a set of distinct tags for each URL, which is denoted by $\mathcal{T}_l$. We adopt the similar method as adopted for addition pattern generation to generate substitution patterns.

On all $\mathcal{T}_l$'s, the mutual information of two tags, $t_1$ and $t_2$, can be calculated as in Equation (38):

$$I_s(t_1,t_2) = \sum_{Y_{t_1},Y_{t_2}\in\{0,1\}} P(Y_{t_1},Y_{t_2}) \log \frac{P(Y_{t_1},Y_{t_2})}{P(Y_{t_1})P(Y_{t_2})}, \tag{38}$$

where $Y_{t_i}$ indicates whether the term $t_i$ appears in a particular tag set $\mathcal{T}_l$ of the URL $l$. For example, $P(Y_{t_1}=1, Y_{t_2}=1)$ is the proportion of tag sets that contain $t_1$ and $t_2$ simultaneously. And then, the normalized form of $I_s(t_1,t_2)$ is adopted as the substitution pattern mining score $substitute(t_1,t_2)$. For each query term $t^r$, we can generate a ranked list of its candidate substitution terms according to $substitute(t^r,t_i)$ and a particular number of top ranked terms are retained as preliminary substitution candidates of $t^r$.

The preliminary substitution candidates obtained above may suffer from noise. The first kind of noise is mainly caused by noun phrases. For example, "north" is found as the top 1 candidate for "carolina", and "united" is the top 1 candidate for "states". These are obviously not good candidates for term substitution. The main reason is that the phrases, i.e. "north carolina" and "united states", are used by many users to tag URLs. Thus, the $NMI$ values of two terms in the same phrase are large. Another kind of noise is caused by the frequently used terms. For example, "free" is found as one of the candidates for many terms such as "music", "movie", "film", etc. By manually examining the data, we find that users maintain a large amount of bookmarks for these free resources on the Web. We propose a method to filter out these undesirable substitution patterns.

For each tag $t$, we first construct its pseudo-context in the bookmark environment, denoted by $C(t)$, which is composed of all tags that co-occur with $t$ in at least one $\mathcal{B}_{lu}$. $cnt\big(t_i|C(t)\big)$ denotes the frequency of $t_i$ in the pseudo-context of $t$. It is equivalent to the co-occurrence frequency of $t$ and $t_i$ in all $\mathcal{B}_{lu}$'s. Then, the pseudo-context document $\mathcal{D}_t$ of the tag $t$ is defined as:

$$\mathcal{D}_t \equiv (w_1^t, w_2^t, \cdots, w_T^t), \tag{39}$$

where $w_i^t$ is the weight of $t_i$ in $\mathcal{D}_t$, which can be computed as:

$$w_i^t = \frac{cnt\big(t_i|C(t)\big)}{\sum_{t_j} cnt\big(t_j|C(t)\big)} \times \log \frac{|\mathbb{D}_T|}{|\{\mathcal{D}_{t_k}|t_i\in\mathcal{D}_{t_k}\}|}, \tag{40}$$

where $\mathbb{D}_T$ denotes the set of all pseudo-context documents, and $T$ is the vocabulary of the bookmark data. Then, we calculate a refined cosine similarity between a pair of pseudo-context documents:

$$sim(\mathcal{D}_{t_i},\mathcal{D}_{t_j}) = \frac{\sum_{k=1}^{|T|}(w_k^{t_i}\times w_k^{t_j})\big(1-\gamma(t_i,t_j|t_k)\big)}{|\mathcal{D}_{t_i}||\mathcal{D}_{t_j}|}, \tag{41}$$

where $\gamma(t_i,t_j|t_k)$ is the relation factor between term $t_i$ and $t_j$ given $t_k$, which is calculated as:

$$\gamma(t_i,t_j|t_k) = \frac{cnt\big(t_i,t_j|C(t_k)\big)}{\min\big\{cnt\big(t_i|C(t_k)\big), cnt\big(t_j|C(t_k)\big)\big\}}, \tag{42}$$

where $cnt\big(t_i,t_j|C(t_k)\big)$ is the number of $\mathcal{B}_{lu}$'s that contain $t_i$, $t_j$ and $t_k$ simultaneously. The design of $\gamma(t_i,t_j|t_k)$ is based on the following observation: one URL is seldom tagged by the same user with two terms that have very similar meaning such as "film" and "movie", "car" and "automobile". According to Equation 42, the

value of $\gamma$("car","automobile"|"repair") tends to be very small, while the value of $\gamma$("carolina","north"|"lottery") tends to be large. Consequently, the contribution of the term "repair" to the similarity between $\mathcal{D}_{car}$ and $\mathcal{D}_{automobile}$ is not penalized, and the contribution of the term "lottery" to the similarity between $\mathcal{D}_{carolina}$ and $\mathcal{D}_{north}$ is decreased. Therefore, the similarity, calculated with Equation 41, between the real substitution pairs such as "car" and "automobile" is not penalized by the factor $\gamma$, while the similarity between the noisy substitution pairs such as "north" and "carolina" is penalized by the factor $\gamma$. Furthermore, the refined cosine similarity can also filter out the terms which are frequently used such as "free" and "howto", because their contexts are quite diverse, and cannot have a high similarity with other specific terms' contexts.

### 6.2. Query-Log-Based Candidate Pattern Generation

Generally, Web search users issue queries with terms in *natural word order*. For example, "java tutorial" is issued much more frequently than "tutorial java". Therefore, the syntagmatic and paradigmatic relations [Jacquemin 1999; Rapp 2002] can be explored among the query term sequences for generating addition and substitution patterns.

*6.2.1. Addition Pattern.* We define left context and right context to capture the syntagmatic relations of a term in the query log data. Let $L(t)$ and $R(t)$ denote the sets of terms that occur on the left and right of $t$, respectively:

$$L(t) = \{t' | \exists q \in Q,\ \mathbf{I}_{\langle t't \rangle}(q) = 1\}, \tag{43}$$

$$R(t) = \{t' | \exists q \in Q,\ \mathbf{I}_{\langle tt' \rangle}(q) = 1\}, \tag{44}$$

where $\mathbf{I}_{\langle t't \rangle}(q)$ indicates whether the sequence $t't$ appears in $q$. Note that the above contexts are collected after the stop words and function words are removed from the queries to avoid their negative effect on context generation. The smoothed probability of each term in a context is calculated as follows:

$$P(t'; L(t)) = \frac{w(t'; L(t)) + \mu P(t')}{\sum_{t_i} w(t_i; L(t)) + \mu}, \tag{45}$$

$$P(t'; R(t)) = \frac{w(t'; R(t)) + \mu P(t')}{\sum_{t_i} w(t_i; R(t)) + \mu}, \tag{46}$$

where $w(t'; \cdot)$ is the weight of $t'$ in a context, $P(t')$ is a global unigram model estimated with the query log data. In the query log data, the issued queries are different from the effectiveness perspective. The more effective a query is, the more reliable its term context is. Therefore, $w(t'; \cdot)$ should consider the effectiveness of the queries where $t'$ originates. Specifically, if a query results in some clicks, its term contexts are more reliable than those of the queries that result in no click. If a query is the last query of a session and results in at least one click, its term contexts are more reliable. Let $\mathbf{I}_{click}(q)$ denote an indicator that indicates whether a click is resulted in by $q$, $\mathbf{I}_{end}(q)$ denote an indicator that indicates whether $q$ is the end of a session. $w(t'; L(t))$ is calculated as:

$$w(t'; L(t)) = \sum_{q} \{\mathbf{I}_{\langle t't \rangle}(q) + \mathbf{I}_{\langle t't \rangle}(q)\mathbf{I}_{click}(q) + \mathbf{I}_{\langle t't \rangle}(q)\mathbf{I}_{click}(q)\mathbf{I}_{end}(q)\}. \tag{47}$$

For terms $t^r$ and $t^{r+1}$, we aim at finding reliable candidate term $t'$ for generating addition reformulation $t^r t' t^{r+1}$. We first select top $M$ candidate terms, which can be the set of all terms with $w > 0$, from each of $L(t^{r+1})$ and $R(t^r)$ according to the probabilities of $P(\cdot; L(t^{r+1}))$ and $P(\cdot; R(t^r))$, respectively. Let $A(t^r t^{r+1})$ denote the set of merged candidates from both contexts. The score of each $t' \in A(t^r t^{r+1})$ is calculated as follows:

$$\mathcal{S}(t'; t^r t^{r+1}) = P(t'; R(t^r)) P(t'; L(t^{r+1})). \tag{48}$$

Then, the top ranked terms according to $\mathcal{S}(t'; t^r t^{r+1})$ are selected as addition candidates to be added between $t^r$ and $t^{r+1}$. For a query $q: t^{1:n}$, we have $n+1$ positions to generate addition candidates.

*6.2.2. Substitution Pattern.* If the contexts of $t'$ are similar with the contexts of $t^r$, $t'$ is probably a good substitution candidate to generate a reformulated query $t^1 \cdots t^{r-1} t' t^{r+1} \cdots t^n$. To measure the similarity between two context probability distributions, we employ Jensen-Shannon divergence [Lin 1991] which is a symmetrized and smoothed version of the Kullback-Leibler divergence. It was evaluated as one of the best similarity measures between probability distributions in linguistic applications [Lee 1999] and it is defined as:

$$JSD(P,Q) = \frac{1}{2}[KLD(P||M) + KLD(Q||M)], \tag{49}$$

where $M = \frac{1}{2}(P+Q)$, $KLD()$ is the Kullback-Leibler divergence. When the base 2 logarithm is used in $KLD()$ calculation, we have $0 \leq JSD(P,Q) \leq 1$.

The normalized score of using $t'$ to substitute $t^r$ according to their left contexts is calculated as:

$$\mathcal{S}_L(t^r \rightarrow t') = \frac{JSD(P(\cdot; L(t^r)), P(\cdot; L(t')))}{\sum_t JSD(P(\cdot; L(t^r)), P(\cdot; L(t)))}. \tag{50}$$

Similarly, the score according to their right contexts is calculated as:

$$\mathcal{S}_R(t^r \rightarrow t') = \frac{JSD(P(\cdot; R(t^r)), P(\cdot; R(t')))}{\sum_t JSD(P(\cdot; R(t^r)), P(\cdot; R(t)))}. \tag{51}$$

If the left context of $t^r$ is more diverse, i.e., containing more terms, than the right context, $\mathcal{S}_L(t^r \rightarrow t')$ should attain a higher weight than $\mathcal{S}_R(t^r \rightarrow t')$. It is because the more diverse a context is, the more reliable the substitution score is. Therefore, we combine the above two substitution scores as follows:

$$\mathcal{S}(t^r \rightarrow t') = \frac{|L(t^r)|\mathcal{S}_L(t^r \rightarrow t') + |R(t^r)|\mathcal{S}_R(t^r \rightarrow t')}{|L(t^r)| + |R(t^r)|}. \tag{52}$$

According to this score, we obtain a particular number of top ranked substitution terms for each $t^r$ as preliminary substitution candidates.

As observed in previous works [Jones et al. 2006; Wang and Zhai 2008], purely context-based method may involve some noise in the preliminary substitution candidates. For example, "china mobile" and "china airline" are two popular queries. The term "china" increases the context similarity between "mobile" and "airline", which may result in undesirable substitution pattern ("mobile", "airline"). To filter out the possible noise, we calculate the $NMI(t^r, t')$ among the search sessions, where $NMI$ is defined the same as in previous section. Specifically, $X_{t^r}$ indicates whether the term $t^r$ appears in a particular search session. $P(X_{t^r} = 1, X_{t'} = 1)$ is the proportion of the sessions that contain $t^r$ and $t'$ simultaneously. The rationale is that if $t^r$ and $t'$ are a pair of substitution terms, the substitution operation between them should have been observed in history search sessions.

## 7. DATASETS AND EXPERIMENT SETUP

### 7.1. Datasets and Preprocessing

*7.1.1. Query Log.* The query log data used in our experiments is the AOL log [Pass et al. 2006], which is the largest publicly available query log and still regarded as a reliable experimental dataset in recent works [Feild and Allan 2013; Lucchese et al.

2011; Shokouhi 2013]. AOL query log spans three months from 1 March, 2006 to 31 May, 2006. The raw data is composed of queries and clicks recorded by a search engine. Since it contains a lot of noise, we first clean the raw data as done by most of the existing works. If a query contains non-alphabetical character, or it is a host navigation query such as "google.com" and "www.yahoo.com", it is removed. Then, the stop words are removed from the remaining queries. We adopt a combined method to detect user session boundaries via combining the advantages of both task-based session detection and time-gap-based session detection. First, two consecutive queries in the same session must share a pair of synonym terms. We employ the synonym relation in WordNet [Miller 1995] and the redirection relation in Wikipedia[4] for synonym term detection. A term is defined as one synonym term of its own. The effectiveness of using common terms in task-based session detection has been illustrated in some previous works [Jones and Klinkner 2008]. Second, the interval of two consecutive queries should be less than 10 minutes. This time interval is determined based on two observations on AOL query log [Lucchese et al. 2011]. One observation is that the average length of a time-gap session is about 15 minutes and the other observation is that there are an average of 2.57 queries per search task in AOL query log. Therefore, 10-minute interval of two consecutive queries is reasonable to detect task-based sessions. After sessions are detected, we remove those sessions without any clicked URL. In each session, we remove the queries which are in the end of the sessions and have no clicked URL. Finally, we obtain $7,031,642$ sessions, in which $1,282,165$ are multi-query sessions. There are $4,315,124$ unique queries and $673,073$ distinct terms.

The dataset is split into two subsets by the time stamp, namely, the history set and the testing set. The history set contains the first two months' log, and the testing set contains the third month's log. The pseudo-documents for latent topic detection are constructed with the history set as mentioned in Section 4. If a host involves less than 5 queries, it will be eliminated. Then, all pseudo-documents are ranked in descending order according to the number of distinct terms they have. In order to filter out those Web sites that are too general, we remove hosts from the top 0.1% of the pseudo-documents. Finally, 189,670 pseudo-documents are retained and fed into the latent topic detection algorithm as presented in Section 4.1. The number of preliminary substitution candidates is 100. The filtering $NMI$ threshold value used is 0.001 when filtering the noise in the preliminary candidates. The generation of substitution patterns is conducted offline and a list of candidates for each single term in the history query set is generated and stored. The number of addition candidate terms for each position in an input query is 50. For the generation of addition patterns, we first select top 200 weighted context terms for position $t^r\_\_t^{r+1}$ from each of $L(t^{r+1})$ and $R(t^r)$. Then, $\mathcal{S}(t'; t^r t^{r+1})$ is computed for those candidates. Finally, the top 50 for this position is retained.

*7.1.2. Bookmark Data.* Wetzker et al. [2008] provided a Delicious bookmark collection. In our experiment, we use a subset of this data collection spanning two years, namely, 2006 and 2007. In this period, 101,485,670 bookmarks were assigned to 39,232,530 distinct URLs by 891,479 users, including 2,523,190 distinct tags. To tackle the vocabulary gap between the AOL log and bookmark data, we filter out the tags that are not contained by the query log such as "socialsoftware", "visualstudio", etc. After this filtering, all bookmarks are utilized to generate the addition patterns and the number of candidate addition patterns for each term is 50. To tackle the problem of data sparsity in substitution pattern generation, we only consider the URLs which are tagged by at least 5 users. After this filtering, 1,910,547 URLs are collected and used in substi-

[4]http://en.wikipedia.org/

tution pattern generation. The number of preliminary substitution candidates is 100, and the similarity threshold value used is 0.19 when filtering the noise in the preliminary candidates. The generation of addition and substitution patterns from bookmark data is conducted offline since it does not need to know the input query in advance.

### 7.2. Experiment Setup

For conducting the comparison, we implement a query reformulation method known as context-based term association (CTA) method proposed in [Wang and Zhai 2008], and follow the advice of the authors on some implementation details. This method is based on an observation that terms with similar meaning tend to co-occur with the same or similar terms in the queries. For example, both "auto" and "car" often occur with "rental", "pricing", etc. Therefore, both candidate term generation and query scoring in CTA are conducted by exploiting the association of term contexts in the history queries. A contextual model is defined to capture context similarity, and a translation model is defined to score the quality of a candidate query. We generate the translation models for the top 100,000 terms in the history set. Since candidate generation from query log is not the focus of our comparison experiment, we also use our candidate generation method on query log for CTA.

We also conduct comparison with another existing method, namely, GenEdit(G), proposed in [Herdagdelen et al. 2010], which is referred to as GEG in this paper. To rank the candidates of an input query, GEG predicts a score for each input and candidate query pair. The score represents the quality of the candidate and it is defined as a revised Levenshtein distance between the input query and the candidate query. First, instead of characters, GEG takes each term as a basic unit in the cost calculation between two queries. Second, whenever there is evidence of semantic association between two terms, it should be "cheaper" to substitute these terms instead of deleting one and adding the other. For an unrelated term pair, a combination of addition and deletion should always be less costly than a substitution. The association between two terms is defined as the normalized PMI value measuring their co-occurrence in the successive queries of the same session. Any term pair that is not co-occurring in any session is assigned a PMI value of zero. In [Herdagdelen et al. 2010], the second dataset contains short queries collected from search engines, which shares similar characteristics as our dataset. GEG achieved the second best performance (1.85% lower than the best). Meanwhile, GEG does not require expensive features such as the number of documents retrieved by a search engine so that it can be reliably reproduced.

We conduct an automatic evaluation by utilizing the session information of query log, which was observed providing strong indication on users' satisfaction [Bing et al. 2011; Hassan et al. 2011; Song et al. 2012]. In a search session, when users feel unsatisfied with the results of the current query, they may reformulate the current query and search again. After obtaining satisfactory results, they stop searching. Therefore, an automatic evaluation can be conducted by investigating this behavior. We introduce the definitions of two types of query.

DEFINITION 1 (SATISFACTORY QUERY). *In a search session, the query which results in at least one URL clicked and is located in the end of the session is called a satisfactory query.*

DEFINITION 2 (UNSATISFACTORY QUERY). *The query which is located just ahead of the satisfactory query in the same search session is called an unsatisfactory query.*

Thus, given an unsatisfactory query as input, we expect that the corresponding satisfactory query can be generated by a query reformulation method. Note that for an unsatisfactory query in a particular search session, only the query following it in the

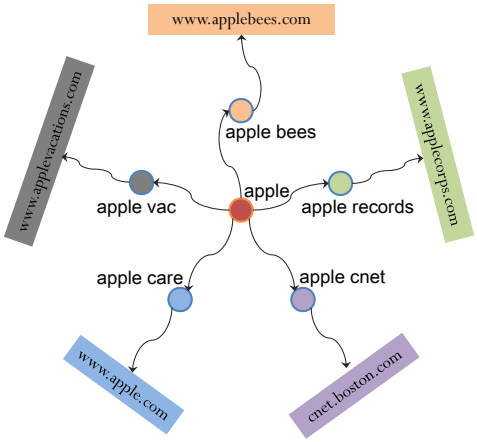


Fig. 2. Different search intentions of users with the same initial query "apple".

Table II. Number of different types of operations in the testing set.

| | Number of satisfactory query | Not included in the training set | |
|---|---|---|---|
| | | Number | Ratio |
| Substitution | 172,789 | 108,928 | 63% |
| Addition | 106,926 | 87,008 | 81% |
| Deletion | 34,329 | 15,333 | 45% |
| Other | 104,453 | 78,797 | 74% |
| Sum | 418,497 | 290,066 | 69% |

same session could be its satisfactory query. It is because the search intentions of different users may be different, even the unsatisfactory queries issued by them are the same in individual search sessions [Lucchese et al. 2011; Sadikov et al. 2010; Teevan et al. 2008]. Thus, the users may generate different satisfactory queries. One such example is given in Fig. 2. Different users issued the same query "apple" in the beginning. After they found the retrieved results are too general to meet their information need, they reformulated the query by adding one term. According to their own information need, five different satisfactory queries are generated. Therefore, in our evaluation, each search session serves as an individual testing case and its unsatisfactory query is employed as one input of the reformulation method, which attempts to generate the satisfactory query of this session. The findings in [Hassan et al. 2013] can also support the rationale of the above setting.

### 7.3. Statistics of Satisfactory Queries

In the testing set of the query log data, there are in total 2,231,546 search sessions and 418,497 multi-query sessions. As defined above, the last queries in search sessions with at least one URL clicked are satisfactory queries. For obtaining the satisfactory query from the second last query in a search session, there are three major types of operations that the users can perform, namely, substitution, addition, and deletion. The numbers of different types of operations in the testing set are given in the second column of Table II. It can be observed that addition and substitution are more often performed by users to reformulate queries. Deletion is relatively less important. As indicated by the "Other" type, users also carry out more complicated operations by combining the above simple operations. We also calculated the number of satisfactory queries that are not included in the training set. From the last column, we can see

Table III. Length difference (length of satisfactory query - length of the corresponding unsatisfactory query).

| Length difference | $\leq$-5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | $\geq$5 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Satisfactory queries | 434 | 1,099 | 4,426 | 18,103 | 59,315 | 172,789 | 120,316 | 34,536 | 6,050 | 1,128 | 301 |
| Ratio | 0.10% | 0.26% | 1.06% | 4.33% | 14.17% | 41.29% | 28.75% | 8.25% | 1.45% | 0.27% | 0.07% |

Table IV. Statistics of new terms for satisfactory query by substitution.

| | Number of new terms | | | | | | |
|---|---|---|---|---|---|---|---|
| Satisfactory query length | 1 | 2 | 3 | 4 | 5 | $\geq$6 | Sum |
| 2 | 66,347 | 5,832 | 0 | 0 | 0 | 0 | 72,179 |
| 3 | 39,999 | 14,158 | 8,111 | 0 | 0 | 0 | 62,268 |
| 4 | 15,872 | 5,458 | 2,503 | 3,847 | 0 | 0 | 27,680 |
| 5 | 4,726 | 1,280 | 631 | 455 | 936 | 0 | 8,028 |
| 6 | 1,272 | 293 | 107 | 84 | 63 | 207 | 2,026 |
| $\geq$7 | 409 | 83 | 23 | 18 | 18 | 57 | 608 |
| Sum | 128,625 | 27,104 | 11,375 | 4,404 | 1,017 | 264 | 172,789 |

Table V. Statistics of new terms for satisfactory query by addition.

| | Number of new terms | | | | | | |
|---|---|---|---|---|---|---|---|
| Satisfactory query length | 1 | 2 | 3 | 4 | 5 | $\geq$6 | Sum |
| 2 | 21,023 | 0 | 0 | 0 | 0 | 0 | 21,023 |
| 3 | 32,864 | 7,267 | 0 | 0 | 0 | 0 | 40,131 |
| 4 | 15,865 | 11,011 | 1,332 | 0 | 0 | 0 | 28,208 |
| 5 | 5,457 | 4,809 | 1,668 | 250 | 0 | 0 | 12,184 |
| 6 | 1,331 | 1,478 | 718 | 287 | 47 | 0 | 3,861 |
| $\geq$7 | 457 | 475 | 310 | 167 | 71 | 39 | 1,519 |
| Sum | 76,997 | 25,040 | 4,028 | 704 | 118 | 39 | 106,926 |

that about 63%, 81% and 45% of the satisfactory queries generated by substitution, addition, and deletion respectively do not exist in the training set. Deletion operation is relatively easier for users and addition operation is more difficult. The reason is that the addition operation has much more candidates and it involves more difficulty for users to generate the corresponding satisfactory queries. In total, 69% of the satisfactory queries do not exist in the training set. It means that any query suggestion method cannot successfully recommend satisfactory queries for these 69% of the cases. We also calculate the length difference between the satisfactory queries and the unsatisfactory queries. The result is given in Table III. It can be observed that for more than 84% of the cases the length difference is no more than 1, which is generally coincident with the observation that more than 76% of times users only modify one of the terms in the queries [Song et al. 2012]. This provides the rationality for why most existing works [Bing et al. 2011; Wang and Zhai 2008] only introduce one new term in query reformulation.

Since we focus on the substitution and addition operations in this paper, some further analysis on the characteristics of the satisfactory queries obtained by these two operations is conducted. The number of new terms in the satisfactory queries obtained by substitution is given in Table IV. For the queries with length 2, most of them are generated by replacing one original term in the unsatisfactory queries. For longer queries, the conservative strategy of replacing one or two original terms is also preferred. In total, this strategy covers about 90% of the successful substitution operations performed by the users. Similar statistics for the satisfactory queries generated by addition operation is given in Table V. With respect to the addition operation, users normally add one or two new terms to make the unsatisfactory queries more specific to retrieve satisfactory results. In total, this strategy covers about 91% of the successful addition operations performed by the users. One interesting observation for the addition operation is that the shorter the unsatisfactory queries are, the more new

terms are needed to generate the satisfactory queries. This is coincident with the fact that the shorter the unsatisfactory queries are, the less specific the information need specified by them is.

## 8. EXPERIMENTAL RESULTS OF QUERY REFORMULATION

### 8.1. Reformulation Performance

*8.1.1. Performance Comparison of Single Operation.* Recall that we propose a general method for candidate query reformulation and an extension method for personalization. The evaluation of the personalization extension will be given later since this method requires an interest profile of a user and it is not applicable to arbitrary input queries. In this section, we first conduct the performance comparison among our general query reformulation method, GEG, and CTA. For our scoring method, we adopt three settings by varying the window size $x$ as discussed in Section 3. Specifically, we let $x = 2$ and $x = 3$ for both n-gram and skip-bigram strategies. When $x = 2$, both n-gram and skip-bigram become the same bigram model. Thus, we have three different variants of our scoring model, namely, Model1 with $x = 2$, Model2 with $x = 3$ for n-gram, and Model3 with $x = 3$ for skip-bigram. Model1 is similar to our previous work [Bing et al. 2011].

The performance comparison among the variants of our method, GEG, and CTA is conducted on scoring the candidate queries generated by substituting one term of the input queries. All the 128,625 sessions in the second column of Table IV are employed as testing cases for this experiment. Specifically, the second last queries, i.e., unsatisfactory queries, of these sessions are used as the input for a reformulation method, and the corresponding satisfactory queries are used as the gold standard results. For an input query $q$, we first generate a list of candidate terms for each term of $q$. Then, each candidate term is used to generate a candidate query. All candidate queries of $q$ are fed into one scoring method to evaluate the quality. Then, according to their scores, the top candidates are determined as the final output results for $q$.

We evaluate the performance of each method at top K candidate queries. Specifically, the recall value at K (Recall@K) is calculated for a method. If the gold standard result can be found in the top K, the reformulation of the input query is considered as successful. Recall@K is computed as the successful cases divided by all employed testing cases. For each method, at most 30 result queries are considered here. Mean Reciprocal Rank (MRR) is another possible metric to conduct evaluation for this type of single answer retrieval task [Voorhees 2009]. Actually, both MRR@K and Recall@K are monotone increasing functions of the number K. Therefore, both of them can show the relative advantages of different methods. Since we also want to show the rate of successful cases and Recall@K value happens to be equal to the success rate, we report the performance of different methods evaluated with Recall@K.

The performance of different methods for scoring the substitution operation on one term is given in Fig. 3. It can be observed that the scoring variants of our graphical model outperform GET and CTA significantly when K is small. The major reason for the difference in performance is that our graphical model scores a query taking into account the semantic consistency of the terms in the latent topic space, which cannot be captured by the comparison methods. Because all methods adopt the same set of substitution candidates and most of the input queries have less than 30 candidate queries, similar Recall@K values are achieved when K is larger, e.g, 30. Among the variants of our model, Model3 with skip-bigram language model achieves the best performance. It is because Model3 captures more context information than Model1 with bigram language model. Meanwhile, compared with Model2 with trigram language

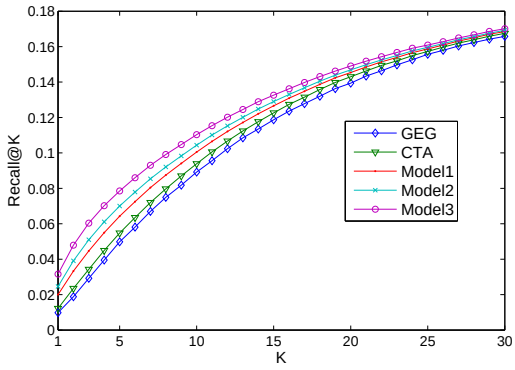


Fig. 3. Comparison of different methods for scoring the substitution operation on one term.

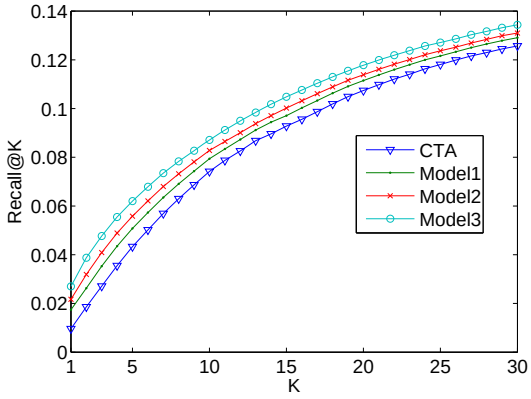


Fig. 4. Comparison of different methods for scoring the addition operation with one new term.

model, Model3 allows more flexible context and avoids the zero frequency issue of trigrams.

Similarly, we also conduct performance comparison between the variants of our method and CTA for scoring the addition operation with one new term added into the input queries. GEG is not compared since it employs a unit cost for a single addition operation and is not appropriate to differentiate the quality of addition candidates. All the 76,997 sessions in the second column of Table V are employed as testing cases for this experiment. We also use the query-log-based addition patterns for candidate query generation. The performance of four methods is given in Fig. 4. We obtained similar results as observed in the above substitution experiment. The scoring variants of our graphical model outperform CTA. Among the variants of our model, Model3 still achieves the best performance and Model2 can outperform Model1.

From the above experiments, we find that Model3 with skip-bigram language model is the most effective variant of our graphical model for both substitution reformulation operation and addition reformulation operation. In addition, we conduct statistical significance test for comparing the performance. Specifically, the improvements of Model3 over GEG, CTA, and Model1 are statistically significant at 95% confidence level or above in paired t-tests. For the test between Model3 and Model2, the $P$ value is 0.055.

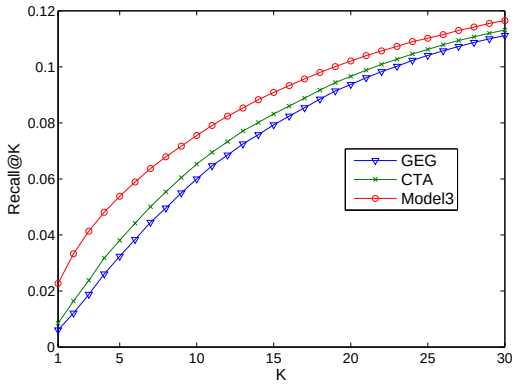


Fig. 5. Comparison of Model3, GEG, and CTA for scoring hybrid candidates from different operations.

*8.1.2. Performance Comparison of Hybrid Operations.* In the above performance investigation of a single operation, it is assumed that we know the operation type that should be carried out for getting the satisfactory query. In this experiment, we relax this assumption and investigate the reformulation performance of hybrid operations on an input query. Specifically, it is possible that a user performs addition or substitution to reformulate a query. Therefore, our system should be able to rank the candidate query of good quality higher among the hybrid candidate queries generated by addition or substitution. Similar to the above experiment, we also restrict the operations to introduce one new term in this experiment. The testing cases include all substitution testing sessions in the second column of Table IV and the addition testing sessions in the second column of Table V excluding the 21,023 sessions whose unsatisfactory queries have length 1. Thus, we have 184,599 testing cases in total. In GEG, the substitution candidates with cost less than 1 are ranked higher than the addition candidates. Otherwise, they are ranked lower than the addition candidates. The addition candidates simply attain a ranking according to the scores of the corresponding addition patterns.

As shown in Fig. 5, our Model3 can outperform GEG and CTA. Paired t-test (with $P < 0.05$) shows that the performance of Model3 is significantly better. When K=1, Model3 achieves more than 100% improvement compared with GEG and CTA. The performance achieved by the hybrid operations is lower than that achieved by individual operations as given in Figs. 3 and 4. The reason is that the number of hybrid candidates is much larger than that of candidates from a single operation, which imposes more challenges on the scoring method.

*8.1.3. Result Analysis.* As shown in Table II, 37% satisfactory queries of the substitution testing cases and 19% satisfactory queries of addition testing cases are existing ones in our history query set. Therefore, we want to analyze the ratio of the successful cases whose corresponding gold standard results, i.e., satisfactory queries, are contained by the history query set. Such ratio for the experiment of substitution operation on one term with Model3 scoring is depicted in Fig. 6. In general, our framework generates many unobserved queries out of the history query set to meet the query reformulation need of the users. Specifically, more than 50% of the generated satisfactory queries for query length 2 are unobserved ones at K=10. Such ratios for query length 3, 4, and 5 are more than 70%, 80%, and 90%, respectively. One interesting observation is that the existing ratio of larger query length is smaller than that of smaller query length. It is because there are less number of candidates in the reformulation of shorter

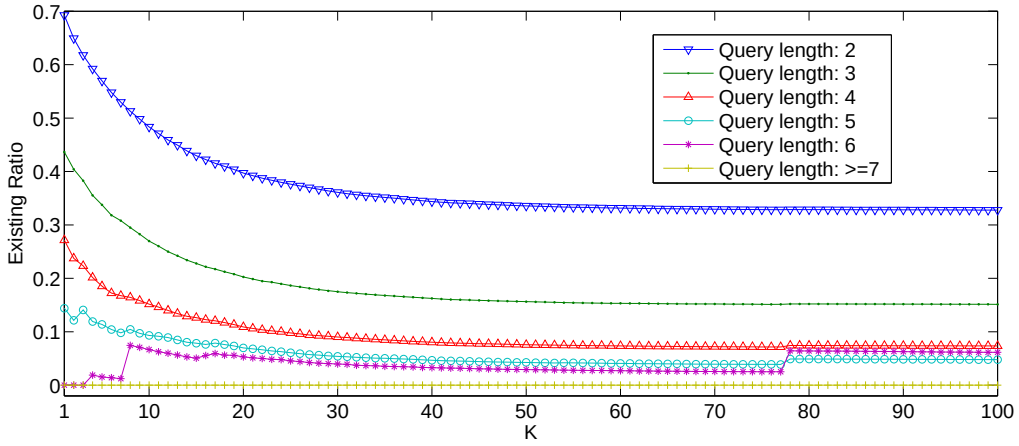


Fig. 6. The ratio of the successful cases whose generated satisfactory queries are existing in the history query set.

queries since the terms to be substituted are less. Consequently, it is more likely that the generated satisfactory query for the shorter input query exists in the history query set. Another observation is that the ratio of smaller K is greater than that of larger K. It means that if the generated satisfactory queries are existing ones, they tend to be ranked higher. This can be attributed to the fact that the scoring method considers the term context information in the history query set. Therefore, if the generated satisfactory queries are existing ones, there is more term context available for them to be ranked higher.

We also analyze the effect of query length on the performance of our framework. The results for different length in the experiment of substitution operation on one term with Model3 scoring are depicted in Fig. 7. For small K values, the performance on shorter queries is better than that on longer queries. When K values are large, e.g. greater than 80, the performance on longer queries is better. This phenomenon is due to two reasons. The first reason is that it is easier to rank shorter candidate queries. Therefore, the gold standard results of shorter input unsatisfactory queries are ranked higher and Recall@K values are consequently larger. The second reason is that longer input queries have larger number of candidate queries since there are more terms to be substituted in them. Thus, the chance of successfully reformulating these queries is higher, which is revealed by the larger Recall@K values for large K. We can observe that the performance on query length 3 surpasses the performance on query length 2 at K=27. The performance on query length 4 surpasses the performance on query length 2 at K=34, and surpasses the performance on query length 3 at K=45. Similar trend can be observed for the performance on query length 5 and 6.

### 8.2. Effectiveness of Candidates from Bookmark

Recall that we investigate a social tagging data resource, namely, Delicious bookmark, to generate addition and substitution patterns which are employed as supplements to the patterns generated from query log data. To investigate the efficacy of the candidates generated from social tagging data, we conduct two experiments, namely, substitution operation on one term and addition operation with one new term. Both experiments employ Model3 scoring and compare the performance of two candidate options, namely, query log candidate and hybrid candidate. The results of substitution and ad-

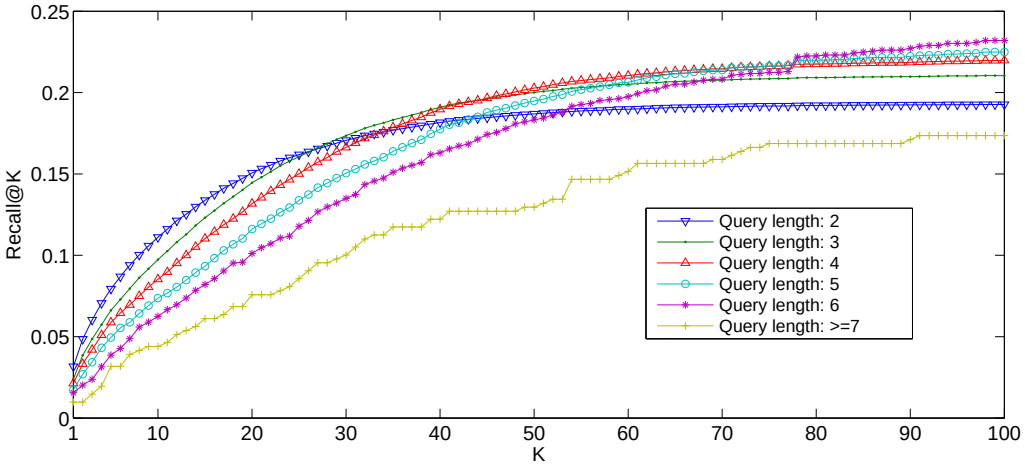


Fig. 7. The effect of input query length for substitution operation on one term with Model3 scoring.

dition experiments are depicted in Figs. 8 and 9, respectively. For both addition and substitution, the hybrid candidate option can achieve better results compared with the pure query log candidate option. The reason is that the hybrid candidate option provides some patterns that are not contained by the history query set. These patterns are useful and employed by the users to reformulate unsatisfactory queries in the testing set. Therefore, the hybrid candidate option can perform better consistently for both small and large K values. 18.2% (0.031/0.17) and 18.5% (0.025/0.135) improvements are achieved for substitution and addition operations respectively when K=30.

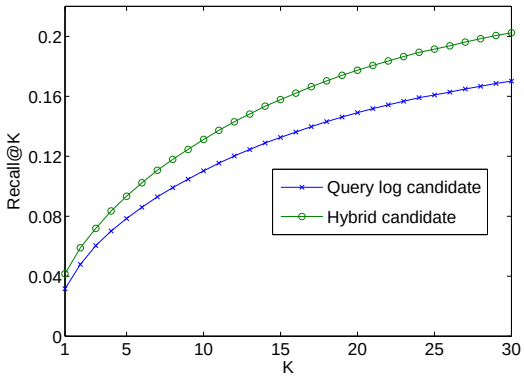


Fig. 8. Comparison of two candidate options, namely, query log candidate and hybrid candidate for substitution operation.

### 8.3. Unfamiliar Query Reformulation

We find that the query reformulation task becomes more challenging when there is very limited term context information for the input query available in the history query set. To investigate this problem in detail, we first define unfamiliar query more precisely as follows:

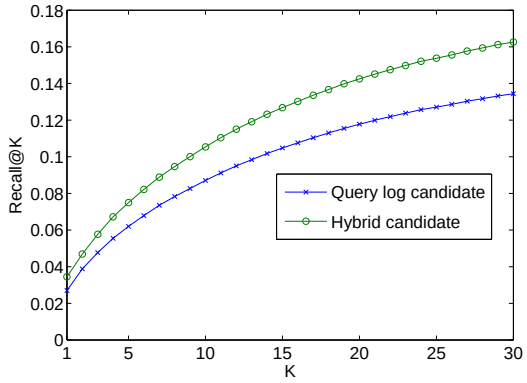


Fig. 9. Comparison of two candidate options, namely, query log candidate and hybrid candidate for addition operation.

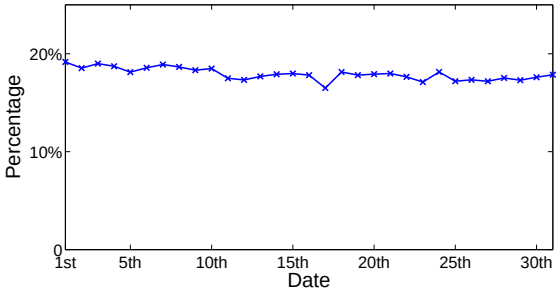


Fig. 10. Percentage of unfamiliar queries.

DEFINITION 3 (UNFAMILIAR QUERY). *Let* $(t_1, \cdots, t_n)$ *denote a particular query* $q$ *with length greater than 1, i.e.* $n > 1$. *Let* $(t_i, \cdots, t_j)$ *denote a sub-sequence of* $q$, *where* $1 \leq i < j \leq n$. *If none of* $(t_i, \cdots, t_j)$*'s has ever been observed before as a whole query or a sub-sequence of any query,* $q$ *is regarded as an unfamiliar query.*

We conduct an investigation on the percentage of unfamiliar queries in the query log used in our experiments. We started with using queries of the first 2 months as the history queries on hand. At the beginning, the queries in the first day of the third month were used to simulate the newly issued queries, and we calculated the percentage of unfamiliar queries in that day. Then, these queries were added into history query set so that they were treated as history queries when we proceeded to the second day of the third month. The same statistical procedure was repeated for all the subsequent days. The statistical information is shown in Fig. 10. We can see that the percentage of unfamiliar queries does not change significantly when the amount of history log accumulates. Thus, it shows that a certain percentage of unfamiliar queries will always be issued by users to convey their information need on up-to-date things. Consequently, it poses more challenges in generating candidates for those unfamiliar queries due to the lack of context information. Moreover, it also causes difficulty in scoring the candidate queries.

We conduct an experiment of substitution operation on one term of the unfamiliar queries in the second column of Table IV. In this experiment, we compare three combinations of candidate generation and query scoring, namely, CTA with query log

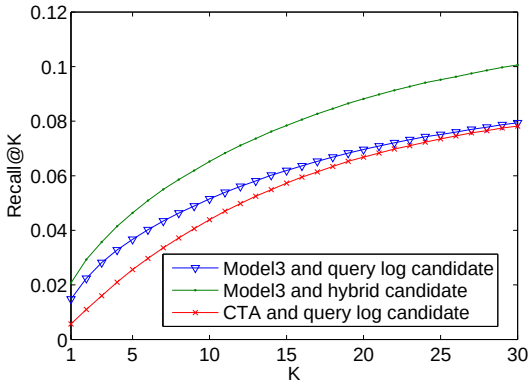


Fig. 11. Performance of substitution operation on one term of the unfamiliar queries.

candidate, Model3 with query log candidate, and Model3 with hybrid candidate. The result is depicted in Fig. 11. Referring to the results reported in Figs. 3 and 8, the Recall@K values of all three methods on unfamiliar queries are significantly lower, which shows that it is very challenging to reformulate such queries. Comparing the results of the three methods in Fig. 11, there are two important observations. First, the hybrid candidate can improve the performance for unfamiliar query reformulation more significantly. Precisely, with K=30, the improvement ratio of Model3 with hybrid candidate over Model3 with query log candidate is about 25% (0.02/0.08). While in Fig. 8, the improvement ratio is about 18%. It is because it is more difficult to figure out the candidate terms to reformulate the unfamiliar queries purely based on history query log. Second, Model3 with query log candidate outperforms CTA with query log candidate more significantly for small K. The reason is that CTA mines the history queries to obtain the context hints to differentiate the quality of different candidate queries. There is no or very little related information available for the unfamiliar input queries. Our graphical model can detect and maintain consistency of semantic meaning when it scores a candidate query, which is achieved with the sequence of hidden nodes representing the latent topics of the corresponding terms. This characteristic enables the graphical model to conduct reliable scoring for the candidate queries of an unfamiliar query.

### 8.4. Personalized Scoring

As mentioned that even the unsatisfactory queries are the same in two different search sessions, the users may generate different satisfactory queries since the search intention of different sessions may be different. Our personalized scoring method can generate more tailor-made ranking of the candidate queries for the same unsatisfactory query from different users' search sessions.

Among the 386,952 users appearing in the history set. Fraction of users that issued different numbers of queries in the history set is given in Fig. 12, which generally follows Zipf's law. There are 9,720 users each of whom issued more than 100 queries. We call them active users. In total, these active users issued 28% of the queries in the history set. We generate interest profiles for the active users since their history queries are abundant for deriving their interest. The value of $\sigma$ in Gaussian kernel was set to be 20. Among 418,497 multi-query sessions in the testing set, 99,168 are issued by those active users. 31,259 testing cases in the second column of Table IV are issued

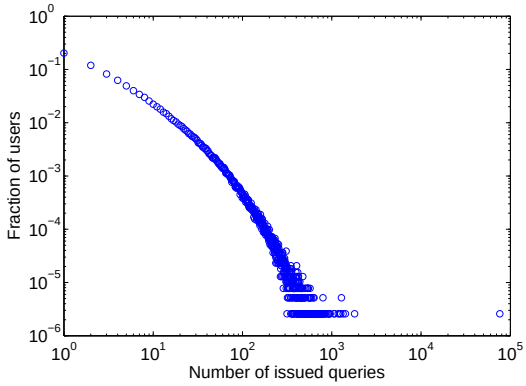


Fig. 12. Fraction of users that issued different numbers of queries in the history set.

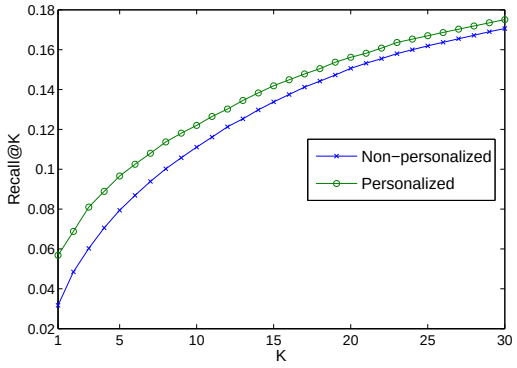


Fig. 13. Comparison of personalized and non-personalized scoring method.

by the active users. We employ all of them to conduct an experiment of substitution operation on one term.

The comparison between personalized and non-personalized results with Model3 scoring is depicted in Fig. 13. The personalized scoring results can outperform the non-personalized results. Especially, when K is small, the improvement is more significant. One reason is that the tailor-made ranking of the candidate queries captures the interest of the user with his or her interest profile. For the example given in Fig 2, the tailor-made rankings form different users impose the space to achieve better performance. Another reason is that, as shown in previous research [Teevan et al. 2007], these active users normally revisit some previous searching topics and each user has his or her own preferred keyword vocabulary. The topic-based interest profile is generated with the history queries that reveal one user's keyword vocabulary. Therefore, the personalized scoring can favor the candidate queries that meet the user's vocabulary well.

### 8.5. Impact of Topic Number

The number of topics used in the latent semantic detection is an important parameter of our framework. To investigate its impact on the performance, we enumerate the number of topics from 10 to 100 with step interval 10. For each topic number, the sub-

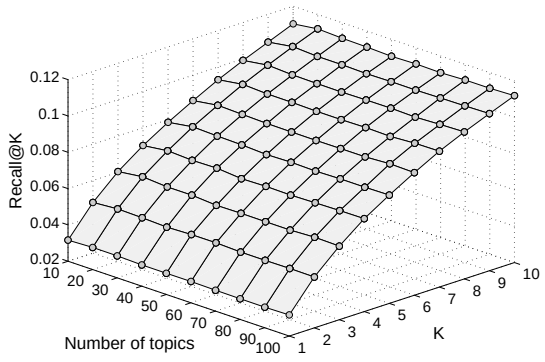


Fig. 14. The performance with different number of topics.

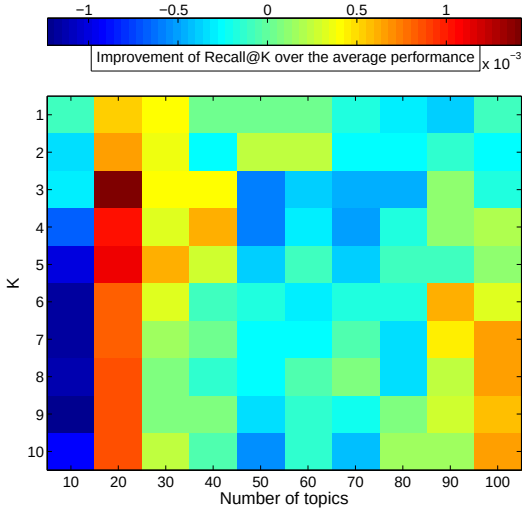


Fig. 15. The performance improvements of different number of topics over the average performance.

stitution experiment on one term is carried out with Model3 scoring. The performance results are depicted in Fig. 14 and the improvements of different topic numbers over the average performance are depicted in Fig. 15. The results show that our framework is not sensitive to the number of topics. The topic number 20 is slightly better than the others. In Fig. 15, two regions are worthy of more attention, namely, the region near the upper left, and the region in the lower right corner. These two regions achieve relatively better results than other nearby topic numbers at the corresponding K values. In general, we can draw the conclusion that smaller topic numbers can perform better at smaller K values, and larger topic numbers can perform better at larger K values. These two impacts together illustrate why the topic numbers 20, 30, 90, and 100 achieve slightly better Recall@10 values.

Table VI. Web search quality measured by NDCG.

| Candidate Rank | Scoring method | | | Unsatisfactory query |
|---|---|---|---|---|
| | GEG | CTA | Model3 | |
| 1 | 0.456 | 0.512 | 0.619 | |
| 2 | 0.352 | 0.413 | 0.532 | |
| 3 | 0.284 | 0.328 | 0.480 | 0.202 |
| 4 | 0.245 | 0.274 | 0.438 | |
| 5 | 0.192 | 0.240 | 0.377 | |

## 9. WEB SEARCH QUALITY AND CASE STUDY

### 9.1. Evaluation of Web Search Quality

We conduct another experiment to examine the quality of the candidate queries from the perspective of meeting the information need of the user. Specifically, we manually evaluate the quality of the Web search results of candidate queries. 100 sessions from the experiment of hybrid operations are randomly selected for this experiment. The gold standard search results for each session are prepared as follows. We collect the top 10 results of the satisfactory query of a session from Google. After that, two human annotators are asked to annotate these 10 results. To better restore the information need of the original user, we show all information of a session to the annotators, including the queries as well as the clicks of them. The annotators first discuss the possible information need of the user and then they mark the results with binary relevance score. Note that the exact clicked URLs resulted in by the satisfactory queries are not available in the AOL log, since only host names are given. Even the exact clicked URLs are available, the corresponding pages may not exist on the live Web now making it impossible to re-retrieve them by the candidate queries. The top 10 results are also collected from Google for each of the top 5 candidate queries of a session. Given the gold relevant results obtained above for each session, the normalized discounted cumulative gain (NDCG) is calculated for each candidate query. The average NDCG value is calculated for the candidate queries at each rank position, i.e. from rank 1 to rank 5. The results are shown in Table VI. The average NDCG value of the unsatisfactory queries of the testing sessions is also given in the last column. In almost all cases, these three methods can outperform the unsatisfactory query, which shows that query reformulation can help users get better search results. Paired t-tests show that Model3 is significantly better than GEG and CTA with $P < 0.01$. One interesting observation is that although the success rate (i.e., Recall value) in query reformulation is not high (referring to Figs. 3, 4, and 5), the candidate queries from Model3 are able to retrieve results with encouraging quality. Specifically, the best candidate queries of Model3 achieve an NDCG value of 0.619.

### 9.2. Case study

In this subsection, we show some case studies of candidate pattern from social tagging data, i.e., Delicious bookmark.

*9.2.1. Addition Patterns.* The candidate addition terms for 12 selected terms are given in Table VII. For each term, the top 5 candidates are given along with their Normalized Mutual Information ($NMI$) scores. In general, there are three kinds of relations between the terms in an addition pattern. The first kind refers to the fact that the terms in an addition pattern are context terms in phrases, such as “purifier” for “air”, “card” for “birthday”, “used” for “car”, etc. The second kind refers to the fact that one term in an addition pattern is a hypernym or hyponym of the other term, such as “email” for “mail”, “property” for “house”, “euromillions” for “lotto” etc. The last kind refers to the fact that the terms in an addition pattern share very similar properties, such as “tablet” for “pc”, “aim” for “msn”, etc.

Table VII. Addition patterns from social tagging data.

| air | | birthday | | car | |
|---|---|---|---|---|---|
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| purifier | 0.082 | party | 0.063 | insurance | 0.064 |
| purifiers | 0.055 | card | 0.036 | used | 0.050 |
| conditioning | 0.045 | invitations | 0.032 | rental | 0.047 |
| force | 0.037 | greeting | 0.028 | cheap | 0.028 |
| filters | 0.035 | cake | 0.023 | loan | 0.025 |
| **cd** | | **children** | | **health** | |
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| burning | 0.047 | parenting | 0.061 | nutrition | 0.065 |
| burn | 0.038 | nile | 0.021 | diet | 0.055 |
| iso | 0.040 | std | 0.020 | exercise | 0.048 |
| cover | 0.024 | adhd | 0.019 | medicine | 0.043 |
| bestselling | 0.019 | rheumatoid | 0.016 | food | 0.031 |
| **house** | | **lotto** | | **mail** | |
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| prefab | 0.015 | millions | 0.036 | email | 0.062 |
| property | 0.014 | winning | 0.032 | spam | 0.041 |
| rent | 0.013 | euromillions | 0.027 | postfix | 0.033 |
| plans | 0.012 | genauere | 0.024 | webmail | 0.028 |
| estate | 0.011 | jackpot | 0.023 | imap | 0.026 |
| **msn** | | **music** | | **pc** | |
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| messenger | 0.109 | audio | 0.053 | pocket | 0.042 |
| im | 0.046 | radio | 0.031 | plasma | 0.033 |
| techy | 0.026 | guitar | 0.016 | tablet | 0.032 |
| chat | 0.024 | download | 0.015 | samsung | 0.031 |
| aim | 0.035 | itunes | 0.015 | nano | 0.028 |

*9.2.2. Substitution Patterns.* The candidate substitution terms for 12 selected terms are given in Table VIII. In general, there are four kinds of relations between the terms in a substitution pattern. The first kind refers to the fact that the candidates' meaning is identical with input terms' meaning such as "automotive" and "auto" for "car", "kids" for "children", "lottery" for "lotto", etc. The second kind is that the candidate terms' meaning is related to the input term's meaning, such as "dvd" for "cd", "fitness" for "health", "estate" for "house", "messenger" for "msn". etc. The third kind refers to the fact that the candidate terms have a similar property of the input terms such as "icq" and "aim" for "msn". The last kind refers to other types of strongly associated relation. In the last case, although the candidate terms do not have similar meaning or property of the inputs, the relationship between them is quite obvious. For example, "powerball" is a kind of lottery ticket, "rock" is a genre of popular music. Interestingly, two non-English terms are suggested for "music", namely, "musica" from Spanish and "musik" from Swedish, and both are correct.

## 10. RELATED WORK

Query log data has been applied as an important resource in improving the performance of search engines and other related topics [Baeza-Yates and Tiberi 2007; Agichtein et al. 2006; Cui et al. 2002; Wen et al. 2002; Joachims 2002; Radlinski and Joachims 2005; White and Morris 2007; Bing et al. 2014; Zhao et al. 2006]. Web query reformulation has been a hot topic among those research for the past decade. Besides the three major types of reformulation operations, namely, substitution, addition, and deletion [Bing et al. 2011; Jones et al. 2006; Wang and Zhai 2008; Jones and Fain 2003; Balasubramanian et al. 2010], researchers also investigated some other fine-grained types in the query log [Huang and Efthimiadis 2009], including stemming [Peng et al. 2007], spelling correction [Cucerzan and Brill 2004; Li et al. 2012; Duan et al. 2012;

Table VIII. Substitution patterns from social tagging data.

| air | | birthday | | car | |
|---|---|---|---|---|---|
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| apollo | 0.135 | birthdays | 0.085 | cars | 0.319 |
| airline | 0.127 | greeting | 0.061 | auto | 0.260 |
| airlines | 0.113 | ecards | 0.059 | automotive | 0.176 |
| flights | 0.105 | anniversary | 0.053 | automobile | 0.132 |
| flight | 0.100 | ecard | 0.043 | vehicle | 0.082 |
| cd | | children | | health | |
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| dvd | 0.130 | kids | 0.207 | medical | 0.135 |
| cds | 0.081 | parenting | 0.136 | fitness | 0.131 |
| iso | 0.077 | child | 0.080 | medicine | 0.126 |
| burning | 0.071 | baby | 0.072 | nutrition | 0.124 |
| cdrom | 0.064 | parents | 0.053 | diet | 0.103 |
| house | | lotto | | mail | |
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| houses | 0.161 | lottery | 0.321 | email | 0.267 |
| housing | 0.122 | lotteries | 0.127 | webmail | 0.099 |
| realestate | 0.104 | loto | 0.085 | smtp | 0.096 |
| estate | 0.079 | powerball | 0.081 | gmail | 0.091 |
| property | 0.063 | lotterie | 0.042 | spam | 0.090 |
| msn | | music | | pc | |
| *cand.* | *NMI* | *cand.* | *NMI* | *cand.* | *NMI* |
| messenger | 0.219 | audio | 0.135 | computer | 0.092 |
| im | 0.115 | musica | 0.118 | computers | 0.088 |
| icq | 0.105 | musik | 0.087 | hardware | 0.074 |
| aim | 0.103 | bands | 0.076 | windows | 0.069 |
| chat | 0.074 | rock | 0.060 | xp | 0.060 |

Duan and Hsu 2011], abbreviation expansion [Wei et al. 2008], syntactic reformulation with syntactic operators [Duan et al. 2011], etc. In [Guo et al. 2008], a Conditional Random Field model is proposed to conduct some fine-grained reformulation operations. Chirita et al. [2007] exploited user's personal information repository such as text documents, emails and cached Web pages to implicitly personalize the query expansion for the user. Sadikov et al. [2010] conducted an investigation on clustering the reformulations of the same original query. The clusters provide a summary of the possible diverse interests of the users who issue the same original query. Their method can also be used to achieve a better personalized query reformulation. Some works investigated the effectiveness of using anchor text in query reformulation [Dang and Croft 2010; Kraft and Zien 2004]. They find that anchor text is also an effective resource. Wikipedia, as an important linguistic repository, was also utilized in query expansion and encouraging results were reported in the task of blog post retrieval [Weerkamp et al. 2012]. Query completion [White and Marchionini 2007; Shokouhi 2013; Bar-Yossef and Kraus 2011] is another hot topic in query assistance service, which completes the queries when the users typing in the search box and can be regarded as a special type of query expansion.

Query difficulty (or quality) prediction has also been investigated in recent years. The existing works fall into two broad categories: pre-retrieval prediction [He and Ounis 2006; He et al. 2008; Kim et al. 2013; Zhao et al. 2008] and post-retrieval prediction [Cronen-Townsend et al. 2002; Hauff et al. 2008]. Pre-retrieval predictors generally make use of evidence based on term distribution statistics such as the inverse document frequency, inverse collection term frequency, simplified clarity score [He and Ounis 2006], association rule mining from history query together with external resources such as Wikipedia categories [Kim et al. 2013]. On the other hand, post-retrieval predictors make use of the retrieval results from a particular retrieval al-

gorithm to provide evidence for the prediction. Both kinds of predictions need some external resources, therefore, they cannot fulfill the task of query scoring investigated in this paper.

Another research direction for helping users issue more effective queries is query suggestion. To conduct query suggestion, a commonly used method is mining the click graph [Deng et al. 2009; Mei et al. 2008]. In [Mei et al. 2008], a sub-bipartite graph of queries and URLs is generated for a particular input query. Then, the hitting time of each node is computed by performing a random walk on the subgraph. After that, the query nodes which have the top $n$ earliest hitting time are used as the suggestions. Deng et al. [2009] constructed a query-to-query graph instead of the bipartite graph, and the random walk is conducted on the entire graph. They also proposed an entropy-based method to calculate the weight of the edges. Cao et al. [2008b] investigated context aware query suggestion by mining query log. They used a clustering method on the bipartite graph to summarize queries into concepts. Then, user sessions are employed to mine the concept patterns, which can help to provide more precise suggestions. Query flow graph [Boldi et al. 2008, 2009] is another frequently used data structure in query suggestion, which constructs a query-to-query graph based on session information. Anagnostopoulos et al. [2010] formulated the task of query recommendation as finding shortcut edges in the query flow graph so as to nudge the reformulation paths of users. The richness of query log information has significant impact on the performance of the above methods. Implicit user-feedback-based method [Song and He 2010] and template-based method [Szpektor et al. 2011] are proposed to tackle the problem caused by lacking of enough query log. In semantic query suggestion, researchers aim at recommending related entities from knowledge bases such as DBpedia to the users according to the input queries [Meij et al. 2009].

Instead of employing click through or session information, several query similarity functions [Alfonseca et al. 2009b,a; Herdagdelen et al. 2010; Xu and Xu 2011] are proposed by leveraging term appearance in large Web corpus, temporal patterns of queries, taxonomic relation, variant of edit distance, and n-gram space model. These methods can measure the similarity of two queries and they have been applied in both query suggestion and reformulation in some way. Besides the above types of query suggestion and reformulation methods in which the end users are not involved, Hollink and Vries proposed an interactive way [Hollink and de Vries 2011] which can provide the users some assistant feedback to help them generate more effective queries. The feedback is mined from query log together with a semantic resource, namely DBpedia.

## 11. CONCLUSIONS AND FUTURE WORK

In this paper, we present a framework for performing query reformulation, and this framework mainly has the following contributions. First, the graphical model can detect and maintain consistency of semantic meaning when scoring a candidate query. It is also capable of conducting reliable scoring for the candidates of an unfamiliar query. Second, the graphical model is able to capture the term context in the history query by skip-bigram and n-gram language models. Third, the graphical model is able to take the users' history search interests into consideration when it conducts query reformulation for different users. Fourth, a method is presented to mine addition and substitution patterns from social tagging data. The experimental results demonstrate that the proposed model can conduct more reliable candidate query assessment. Furthermore, social tagging data can provide useful candidate patterns, which are not observed in history queries.

Currently, phrases are separated into single words in our framework in both latent topic detection and term dependency estimation. One future direction is to enhance our model from single word level to phrase level. Some models such as topical n-grams

model [Wang et al. 2007] and phrase translation model [Gao et al. 2010] have attested the efficacy of the phrase level solution. In this work, the topic-based interest profile is derived from the history queries of a user. However, as discussed in the paper, some users may not have abundant history queries. One possible direction to solve this problem is to investigate other types of personal information, such as Web browsing, browser bookmark, etc.

## REFERENCES

AGICHTEIN, E., BRILL, E., AND DUMAIS, S. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '06. 19–26.

ALFONSECA, E., CIARAMITA, M., AND HALL, K. 2009a. Gazpacho and summer rash: Lexical relationships from temporal patterns of web search queries. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*. EMNLP '09. 1046–1055.

ALFONSECA, E., HALL, K., AND HARTMANN, S. 2009b. Large-scale computation of distributional similarities for queries. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. NAACL-Short '09. 29–32.

ANAGNOSTOPOULOS, A., BECCHETTI, L., CASTILLO, C., AND GIONIS, A. 2010. An optimization framework for query recommendation. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*. WSDM '10. 161–170.

BAEZA-YATES, R. AND TIBERI, A. 2007. Extracting semantic relations from query logs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD '07. 76–85.

BALASUBRAMANIAN, N., KUMARAN, G., AND CARVALHO, V. R. 2010. Exploring reductions for long web queries. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '10. 571–578.

BAR-YOSSEF, Z. AND KRAUS, N. 2011. Context-sensitive query auto-completion. In *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. 107–116.

BAUM, L. E., PETRIE, T., SOULES, G., AND WEISS, N. 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Math. Statist. 41,* 1, 164–171.

BENDERSKY, M., METZLER, D., AND CROFT, W. B. 2011. Parameterized concept weighting in verbose queries. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. SIGIR '11. 605–614.

BILLERBECK, B., SCHOLER, F., WILLIAMS, H. E., AND ZOBEL, J. 2003. Query expansion using associated queries. In *Proceedings of the twelfth international conference on Information and knowledge management*. CIKM '03. 2–9.

BING, L., LAM, W., JAMEEL, S., AND LU, C. 2014. Website community mining from query logs with two-phase clustering. In *Computational Linguistics and Intelligent Text Processing - Proceedings of 15th International Conference (Part II)*. CICLing '14. 201–212.

BING, L., LAM, W., AND WONG, T.-L. 2011. Using query log and social tagging to refine queries based on latent topics. In *Proceedings of the 20th ACM international*

*conference on Information and knowledge management*. CIKM '11. 583–592.
BLEI, D. M., NG, A. Y., AND JORDAN, M. I. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res. 3*, 993–1022.
BOLDI, P., BONCHI, F., CASTILLO, C., DONATO, D., GIONIS, A., AND VIGNA, S. 2008. The query-flow graph: Model and applications. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. CIKM '08. 609–618.
BOLDI, P., BONCHI, F., CASTILLO, C., DONATO, D., AND VIGNA, S. 2009. Query suggestions using query-flow graphs. In *Proceedings of the 2009 Workshop on Web Search Click Data*. WSCD '09. 56–63.
CAI, Y. AND LI, Q. 2010. Personalized search by tag-based user profile and resource profile in collaborative tagging systems. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. CIKM '10. 969–978.
CAO, G., NIE, J.-Y., GAO, J., AND ROBERTSON, S. 2008a. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '08. 243–250.
CAO, H., JIANG, D., PEI, J., HE, Q., LIAO, Z., CHEN, E., AND LI, H. 2008b. 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. 875–883.
CHIRITA, P. A., FIRAN, C. S., AND NEJDL, W. 2007. 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. 7–14.
COLLINS-THOMPSON, K. AND CALLAN, J. 2005. Query expansion using random walk models. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. CIKM '05. 704–711.
CRASWELL, N., BILLERBECK, B., FETTERLY, D., AND NAJORK, M. 2013. Robust query rewriting using anchor data. In *Proceedings of the sixth ACM international conference on Web search and data mining*. WSDM '13. 335–344.
CRONEN-TOWNSEND, S., ZHOU, Y., AND CROFT, W. B. 2002. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '02. 299–306.
CROUCH, C. J. AND YANG, B. 1992. Experiments in automatic statistical thesaurus construction. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '92. 77–88.
CUCERZAN, S. AND BRILL, E. 2004. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. EMNLP'04. 293–300.
CUI, H., WEN, J.-R., NIE, J.-Y., AND MA, W.-Y. 2002. Probabilistic query expansion using query logs. In *Proceedings of the 11th International Conference on World Wide Web*. WWW '02. 325–332.
DANG, V. AND CROFT, B. W. 2010. Query reformulation using anchor text. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*. WSDM '10. 41–50.
DENG, H., KING, I., AND LYU, M. R. 2009. 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. 339–346.
DUAN, H. AND HSU, B.-J. P. 2011. Online spelling correction for query completion. In *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. 117–126.
DUAN, H., LI, R., AND ZHAI, C. 2011. Automatic query reformulation with syntactic

operators to alleviate search difficulty. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. CIKM '11. 2037–2040.

DUAN, H., LI, Y., ZHAI, C., AND ROTH, D. 2012. A discriminative model for query spelling correction with latent structural svm. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. EMNLP-CoNLL '12. 1511–1521.

FEILD, H. AND ALLAN, J. 2013. Task-aware query recommendation. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '13. 83–92.

GAO, J., HE, X., AND NIE, J.-Y. 2010. Clickthrough-based translation models for web search: From word models to phrase models. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. CIKM '10. 1139–1148.

GAO, J., XU, G., AND XU, J. 2013. Query expansion using path-constrained random walks. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '13. 563–572.

GUO, J., CHENG, X., XU, G., AND SHEN, H. 2010. A structured approach to query recommendation with social annotation data. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. CIKM '10. 619–628.

GUO, J., XU, G., LI, H., AND CHENG, X. 2008. A unified and discriminative model for query refinement. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. 379–386.

HASSAN, A., SHI, X., CRASWELL, N., AND RAMSEY, B. 2013. Beyond clicks: Query reformulation as a predictor of search satisfaction. In *Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management*. CIKM '13. 2019–2028.

HASSAN, A., SONG, Y., AND HE, L.-W. 2011. A task level metric for measuring web search satisfaction and its application on improving relevance estimation. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. CIKM '11. 125–134.

HAUFF, C., MURDOCK, V., AND BAEZA-YATES, R. 2008. Improved query difficulty prediction for the web. In *Proceedings of the 17th ACM conference on Information and knowledge management*. CIKM '08. 439–448.

HE, B. AND OUNIS, I. 2006. Query performance prediction. *Inf. Syst. 31*, 585–594.

HE, B. AND OUNIS, I. 2007. Combining fields for query expansion and adaptive query expansion. *Inf. Process. Manage. 43*, 1294–1307.

HE, J., LARSON, M., AND DE RIJKE, M. 2008. Using coherence-based measures to predict query difficulty. In *Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval*. ECIR'08. 689–694.

HE, Q., JIANG, D., LIAO, Z., HOI, S. C. H., CHANG, K., LIM, E.-P., AND LI, H. 2009. Web query recommendation via sequential query prediction. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*. ICDE '09. 1443–1454.

HERDAGDELEN, A., CIARAMITA, M., MAHLER, D., HOLMQVIST, M., HALL, K., RIEZLER, S., AND ALFONSECA, E. 2010. Generalized syntactic and semantic models of query reformulation. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '10. 283–290.

HOLLINK, V. AND DE VRIES, A. 2011. Towards an automated query modification assistant. In *1st International Workshop on Usage Analysis and the Web of Data (USEWOD2011)*.

HUANG, J. AND EFTHIMIADIS, E. N. 2009. Analyzing and evaluating query reformulation strategies in web search logs. In *Proceedings of the 18th ACM Conference on*

*Information and Knowledge Management*. CIKM '09. 77–86.

HUANG, Q., SONG, D., AND RÜGER, S. 2008. Robust query-specific pseudo feedback document selection for query expansion. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*. ECIR'08. 547–554.

JACQUEMIN, C. 1999. Syntagmatic and paradigmatic representations of term variation. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. ACL '99. 341–348.

JOACHIMS, T. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. 133–142.

JOACHIMS, T., GRANKA, L., PAN, B., HEMBROOKE, H., RADLINSKI, F., AND GAY, G. 2007. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Trans. Inf. Syst. 25,* 2.

JONES, R. AND FAIN, D. C. 2003. Query word deletion prediction. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. SIGIR '03. 435–436.

JONES, R. AND KLINKNER, K. L. 2008. Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. CIKM '08. 699–708.

JONES, R., REY, B., MADANI, O., AND GREINER, W. 2006. Generating query substitutions. In *Proceedings of the 15th International Conference on World Wide Web*. WWW '06. 387–396.

KIM, Y., HASSAN, A., WHITE, R. W., AND WANG, Y.-M. 2013. Playing by the rules: mining query associations to predict search performance. In *Proceedings of the sixth ACM international conference on Web search and data mining*. WSDM '13. 133–142.

KRAFT, R. AND ZIEN, J. 2004. Mining anchor text for query refinement. In *Proceedings of the 13th International Conference on World Wide Web*. WWW '04. 666–674.

LEE, C.-J., CHEN, R.-C., KAO, S.-H., AND CHENG, P.-J. 2009. A term dependency-based approach for query terms ranking. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*. CIKM '09. 1267–1276.

LEE, L. 1999. Measures of distributional similarity. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. ACL '99. 25–32.

LEVINSON, S. E., RABINER, L. R., AND SONDHI, M. M. 1983. An introduction to the application of the theory of probabilistic functions of markov process to automatic speech recognition. *Bell System Technical Journal 62,* 4, 1035–1074.

LI, Y., DUAN, H., AND ZHAI, C. 2012. A generalized hidden markov model with discriminative training for query spelling correction. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '12. 611–620.

LIN, J. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory 37*, 145–151.

LUCCHESE, C., ORLANDO, S., PEREGO, R., SILVESTRI, F., AND TOLOMEI, G. 2011. Identifying task-based sessions in search engine query logs. In *Proceedings of the fourth ACM international conference on Web search and data mining*. WSDM '11. 277–286.

MEI, Q., ZHOU, D., AND CHURCH, K. 2008. Query suggestion using hitting time. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. CIKM '08. 469–478.

MEIJ, E., BRON, M., HOLLINK, L., HUURNINK, B., AND RIJKE, M. 2009. Learning semantic query suggestions. In *Proceedings of the 8th International Semantic Web Conference*. ISWC '09. 424–440.

MILLER, G. A. 1995. Wordnet: a lexical database for english. *Commun. ACM 38,* 11, 39–41.

OGILVIE, P., VOORHEES, E., AND CALLAN, J. 2009. On the number of terms used in automatic query expansion. *Inf. Retr. 12,* 6, 666–679.

PASS, G., CHOWDHURY, A., AND TORGESON, C. 2006. A picture of search. In *Proceedings of the 1st International Conference on Scalable Information Systems*. InfoScale '06.

PENG, F., AHMED, N., LI, X., AND LU, Y. 2007. Context sensitive stemming for web search. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '07. 639–646.

QIU, Y. AND FREI, H.-P. 1993. Concept based query expansion. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '93. 160–169.

RADLINSKI, F. AND JOACHIMS, T. 2005. Query chains: Learning to rank from implicit feedback. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. KDD '05. 239–248.

RAPP, R. 2002. The computation of word associations: comparing syntagmatic and paradigmatic approaches. In *Proceedings of the 19th international conference on Computational linguistics - Volume 1*. COLING '02. 1–7.

SADIKOV, E., MADHAVAN, J., WANG, L., AND HALEVY, A. 2010. Clustering query refinements by user intent. In *Proceedings of the 19th International Conference on World Wide Web*. WWW '10. 841–850.

SHOKOUHI, M. 2013. Learning to personalize query auto-completion. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '13. 103–112.

SONG, Y. AND HE, L.-W. 2010. Optimal rare query suggestion with implicit user feedback. In *Proceedings of the 19th International Conference on World Wide Web*. WWW '10. 901–910.

SONG, Y., ZHOU, D., AND HE, L.-W. 2012. Query suggestion by constructing term-transition graphs. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*. WSDM '12. 353–362.

SUN, R., ONG, C.-H., AND CHUA, T.-S. 2006. 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. 382–389.

SZPEKTOR, I., GIONIS, A., AND MAAREK, Y. 2011. Improving recommendation for long-tail queries via templates. In *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. 47–56.

TEEVAN, J., ADAR, E., JONES, R., AND POTTS, M. A. S. 2007. Information re-retrieval: repeat queries in yahoo's logs. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '07. 151–158.

TEEVAN, J., DUMAIS, S. T., AND LIEBLING, D. J. 2008. To personalize or not to personalize: modeling queries with variation in user intent. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '08. 163–170.

VOORHEES, E. 2009. The trec-8 question answering track report. In *Proceedings of the 8th Text REtrieval Conference*. TREC-8. 77–82.

WANG, X., MCCALLUM, A., AND WEI, X. 2007. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*. ICDM '07. 697–702.

WANG, X. AND ZHAI, C. 2008. Mining term association patterns from search logs

for effective query reformulation. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. CIKM '08. 479–488.

WEERKAMP, W., BALOG, K., AND DE RIJKE, M. 2012. Exploiting external collections for query expansion. *ACM Trans. Web 6,* 4, 18:1–18:29.

WEI, X., PENG, F., AND DUMOULIN, B. 2008. Analyzing web text association to disambiguate abbreviation in queries. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '08. 751–752.

WEN, J.-R., NIE, J.-Y., AND ZHANG, H. 2002. Query clustering using user logs. *ACM Trans. Inf. Syst. 20,* 1, 59–81.

WETZKER, R., ZIMMERMANN, C., AND BAUCKHAGE, C. 2008. Analyzing social bookmarking systems: A del.icio.us cookbook. In *Mining Social Data (MSoDa) Workshop Proceedings*. 26–30.

WHITE, R. W. AND MARCHIONINI, G. 2007. Examining the effectiveness of real-time query expansion. *Inf. Process. Manage. 43,* 3, 685–704.

WHITE, R. W. AND MORRIS, D. 2007. Investigating the querying and browsing behavior of advanced search engine users. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '07. 255–262.

XU, J. AND CROFT, W. B. 2000. Improving the effectiveness of information retrieval with local context analysis. *ACM Trans. Inf. Syst. 18,* 1, 79–112.

XU, J. AND XU, G. 2011. Learning similarity function for rare queries. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. WSDM '11. 615–624.

XUE, X. AND CROFT, W. B. 2013. Modeling reformulation using query distributions. *ACM Trans. Inf. Syst. 31,* 2, 6:1–6:34.

ZHAO, L. AND CALLAN, J. 2012. Automatic term mismatch diagnosis for selective query expansion. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '12. 515–524.

ZHAO, Q., HOI, S. C. H., LIU, T.-Y., BHOWMICK, S. S., LYU, M. R., AND MA, W.-Y. 2006. Time-dependent semantic similarity measure of queries using historical click-through data. In *Proceedings of the 15th international conference on World Wide Web*. WWW '06. 543–552.

ZHAO, Y., SCHOLER, F., AND TSEGAY, Y. 2008. Effective pre-retrieval query performance prediction using similarity and variability evidence. In *Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval*. ECIR'08. 52–64.