Query Expansion Using Path-Constrained Random Walks

Jianfeng Gao
Microsoft Research, Redmond
Washington 98052, USA
jfgao@microsoft.com

Gu Xu
Microsoft Bing, Bellevue
Washington 98004, USA
guxu@microsoft.com

Jinxi Xu
Microsoft Bing, Bellevue
Washington 98004, USA
jinxixu@microsoft.com

ABSTRACT

This paper exploits Web search logs for query expansion (QE) by presenting a new QE method based on path-constrained random walks (PCRW), where the search logs are represented as a labeled, directed graph, and the probability of picking an expansion term for an input query is computed by a learned combination of constrained random walks on the graph. The method is shown to be *generic* in that it covers most of the popular QE models as special cases and *flexible* in that it provides a principled mathematical framework in which a wide variety of information useful for QE can be incorporated in a unified way. Evaluation is performed on the Web document ranking task using a real-world data set. Results show that the PCRW-based method is very effective for the expansion of rare queries, i.e., low-frequency queries that are unseen in search logs, and that it outperforms significantly other state-of-the-art QE methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.2.6 [**Artificial Intelligence**]: *Learning*

General Terms

Algorithms, Experimentation

Keywords

Search Log, Query Expansion, Random Walk, Path Ranking Algorithm, Web Search

1. INTRODUCTION

Term mismatch is one of the fundamental challenges in Web search, where a query and its relevant documents are often composed using different vocabularies and language styles. Query expansion (QE) is an effective strategy to address the challenge. It expands a query issued by a user with additional related terms, called *expansion terms*, so that more relevant documents can be retrieved.

QE is a long-standing research topic in information retrieval (IR). The methods based on automatic relevance feedback (e.g., explicit feedback and pseudo relevance feedback (PRF)) have been proved to be useful for improving the performance of IR on TREC datasets [10, 11, 34, 36, 36, 45, 48]. However, these methods cannot be applied directly to a commercial Web search engine because the relevant documents are not always available and generating pseudo-relevant documents requires multi-phase retrieval, which is prohibitively expensive.

Recent studies demonstrate the success of exploiting search logs (i.e., clickthrough data) for QE [7, 14, 15, 19, 22, 41, 42]. These methods, called *log-based QE*, also derive expansion terms for a query from its (pseudo-)relevant document set. But, different from the methods based on automatic relevance feedback, the relevant document set is identified by user clicks recorded in search logs. For example, the set of (pseudo-)relevant documents of an input query can be formed by including the documents which have been previously clicked for the query or its similar queries [1, 7, 46]. Most state-of-the-art log-based QE methods use a global model that is pre-computed from search logs [14, 19]. The model captures the correlation between query terms and documents terms and can be used to generate expansion terms for the input query on the fly. Despite the effectiveness of log-based QE methods, they suffer from two problems. First is data sparseness. A large portion of queries have very few or no click in search logs, as stated by Zipf's law. Second is the ambiguity of search intent. For example, a term correlation model may fail to distinguish the search intent of the query term "book" in "school book" from that in "hotel booking". Although the problem can be partially alleviated by using the improved correlation models based on phrases and concepts [19], there are plenty of cases where the search intent can only be identified correctly via global context. For example, the query "why 6 bottles in one wrap" is about package, and the intent of the query "acme baked bread" is to look for the bakery in CA. In such cases, a (pseudo-)relevant document set of an input query, if available, is more likely to preserve the original search intent than any pre-computed global correlation model.

In this paper we address the two problems by proposing a new log-based QE method based on path-constrained random walks [32]. We represent the search logs, consisting of billions of clicked query-document pairs, as a labeled, directed graph, where there are three types of nodes, representing respectively queries, documents, and words (i.e., candidate expansion terms), and the edges between nodes are labeled by relations. An example graph, which will be described in detail in Section 3.1, is shown in Figure 1. For each path in the graph that links the input query Q to a candidate expansion term w, there is a path type π, defined by a sequence of edge labels. Each path type can be viewed as a particular process of generating w from Q, and the generation probability $P(w|Q,\pi)$ is computed by random walks along the paths that instantiate the path

SIGIR'13, 28 July – 1 August 2013, Dublin, Ireland.

type π, as known as *path-constrained random walks* (PCRW). Many log-based QE models proposed previously can be formulated in the framework of PCRW by defining particular path types. For example, the method based on relevance feedback, where the pseudo-relevant documents D are defined as the ones that have clicks for the input query Q or its similar queries Q', can be presented using the following path type

$$\langle similar\_Q2Q', click\_Q'2D, generate\_D2w \rangle.$$

This is a three-step random walk. The first step retrieves similar queries by a random walk on edges labeled by the relation $similar\_Q2Q'$. The second follows any edges labeled by $click\_Q'2D$. The third follows any edges labeled by $generate\_D2w$. These relations are summarized in Table 1, and will be described in detail in Section 3.

We will show that PCRW provides a generic and flexible modeling framework in that it not only covers most of the popular log-based QE models as special cases, but also allows us to devise new QE models that can potentially use much richer information than that of previous models. For example, we can define a rich set of walk behaviors that support a variety of labeled edges where different information can be used at different stages of the walk. Some examples will be presented in Section 3.2.

Moreover, because different QE methods often rely on different sources and are potentially complimentary, it is desirable to combine them to address data sparseness and help disambiguate search intent. For example, while the automatic feedback methods using (pseudo-)relevant documents are good to retain search intent but suffer from data sparseness especially for rare queries, the methods based on global term correlation models can be applied equally well to both common and rare queries but, due to the limited context information it captures, may lead to an unexpected shift of search intent. We will show that PCRW provides a flexible mathematical framework in which different QE features, specified by path types π, can be incorporated in a unified way. Formally, in the PCRW-based QE method the probability of picking w for a given Q, $P(w|Q)$, is computed by a learned combination of path-constrained random walks on the graph, i.e., $P(w|Q) = \sum_{\pi \in B} \lambda_\pi P(w|Q, \pi)$, where $\lambda_\pi{}'s$ are the combination weights learned on training data.

Our experiments (Section 4) show that the use of PCRW not only makes QE robust to data sparseness but also help disambiguate search intents, leading to a significant improvement over previous state-of-the-art QE methods.

In the rest of the paper, Section 2 reviews briefly PCRW. Section 3 describes in detail the PCRW-based QE method. Sections 4 and 5 present experiments and related work, respectively. The paper is concluded in Section 6.

2. PRELIMINARIES

This section briefly reviews the path-constrained random walk model. Readers are referred to [32] for a more detailed treatment. The model used in this study is a variant of the one described in [32, 33]. Modifications are made for the QE task.

Consider a directed, labeled graph $G = (C, T)$ where $T \subseteq C \times R \times C$ is the set of labeled edges (also known as *triples*) (c, r, c'). Each triple represents an instance $r(c, c')$ of the relation $r \in R$. For the QE task considered in this study it will be useful to introduce for each relation r a separate probabilistic model θ_r, which is used to assign a score to the edge. The score is the probability of reaching c' from c with a one-step random walk with edge type r, $P(c'|c, \theta_r)$. In Section 3 we will see how the use of relation-specific models allows us to build significantly more expressive QE models.

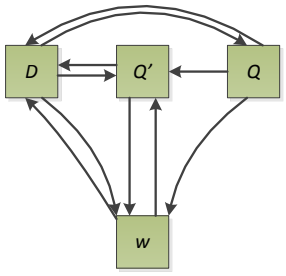

Figure 1. Search logs as a graph

A *path type* in G is a sequence $\pi = \langle r_1, \ldots, r_m \rangle$. An instance of the path type is sequence of nodes $c_0, \ldots, c_m$ such that $r_i(c_{i-1}, c_i)$. Each path type specifies a real-value feature. For a given node pair (s, t), where s is source node and t is target node, the value of the feature π is $P(t|s, \pi)$, i.e., the probability of reaching t from s by a random walk that instantiates the path type, also known as a *path-constrained random walk*. Specifically, suppose that the random walk has just reached c_i by traversing edges labeled $r_1, \ldots, r_i$ with $Q = c_0$. Then c_{i+1} is drawn at random, according to $\theta_{r_{i+1}}$, from all nodes reachable by edges labeled r_{i+1}. A path type π is *active* for pair (s, t) if $P(t|s, \pi) > 0$.

Let $B = \{\perp, \pi_1, \ldots, \pi_n\}$ be the set of all path types of length no greater than l that occur in the graph together with the dummy type $\perp$, which represents the bias feature. It is convenient to set $P(t|s, \perp) = 1$ for any nodes s, t. The score of whether target node t is related to source node s is given by

$$P(t|s) = \sum_{\pi \in B} \lambda_\pi P(t|s, \pi), \quad (1)$$

where λ_π is the weight of feature π. The model parameters to be learned are the vector $\boldsymbol{\lambda} = \langle \lambda_\pi \rangle_{\pi \in B}$. The construction of B and the estimation of $\boldsymbol{\lambda}$ are application specific. For the QE task source node is the input query to be expanded Q and target node is a candidate expansion term w. Thus, Equation (1) gives the probability of whether w is a good expansion term of Q. This is the QE model we will describe in detail in Section 3.

3. PCRW-BASED QE MODEL

3.1 Search Logs as a Graph

The search logs used in this study consist of a list of query-document pairs, also known as *clickthrough data*. Each pair contains a query and a document which has one or more user clicks for the query. We represent the search logs as a graph $G = (C, T)$, as shown in Figure 1. We define three types of nodes to represent respectively queries, documents, and words that occur in queries and documents. While a query in the search logs, denoted by Q', always has clicked document(s), an input query to be expanded, denoted by Q, could be a new, low-frequency query without clicked documents. Such a query is called a *rare query* in this paper. Q and Q' are treated as different nodes in G.

ID	Relation r	Scoring function
1	$similar\_Q2Q'$	cosine similarity between the term vectors of Q and Q', where term weights are assigned using the BM25 function.
2	$translate\_Q2Q'$	$\log \prod_{q' \in Q'} \sum_{q \in Q} P_{tm}(q'\|q) \frac{tf(q;Q)}{\|Q\|}$
3	$click\_Q2D$	$\log P(D\|Q) = \log \frac{click(Q,D)}{\sum_{D_i \in \mathbf{D}} click(Q,D_i)}$
4	$click\_D2Q$	$\log P(Q\|D) = \log \frac{click(Q,D)}{\sum_{Q_i \in \mathbf{Q}} click(Q_i,D)}$
5	$generate\_Q2w$	$\log\left((1-\alpha)\frac{tf(w;Q)}{\|Q\|} + \alpha\frac{cf(w)}{\|C\|}\right)$
6	$translate\_Q2w$	$\log \sum_{q \in Q} P_{tm}(w\|q) \frac{tf(q;Q)}{\|Q\|}$
7	$generate\_Q'2w$	$\log\left((1-\alpha)\frac{tf(w;Q')}{\|Q'\|} + \alpha\frac{cf(w)}{\|C\|}\right)$
8	$translate\_Q'2w$	$\log \sum_{q' \in Q'} P_{tm}(w\|q') \frac{tf(q';Q')}{\|Q'\|}$
9	$click\_Q'2D$	$\log P(D\|Q') = \log \frac{click(Q',D)}{\sum_{D_i \in \mathbf{D}} click(Q',D_i)}$
10	$generate\_D2w$	$\log\left((1-\beta)\frac{tf(w;D)}{\|D\|} + \beta\frac{cf(w)}{\|C\|}\right)$
11	$translate\_D2w$	$\log \sum_{w_i \in D} P_{tm}(w\|w_i) \frac{tf(w_i;D)}{\|D\|}$
12	$click\_D2Q'$	$\log P(Q'\|D) = \log \frac{click(Q',D)}{\sum_{Q_i' \in \mathbf{Q}} click(Q_i',D)}$
13	$generate\_w2D$	$\log P(D\|w) = \log \frac{P_{lm}(w\|D)P(D)}{\sum_{D_i \in \mathbf{D}} P_{lm}(w\|D_i)P(D_i)}$, where $P_{lm}(w\|D) = (1-\beta)\frac{tf(w;D)}{\|D\|} + \beta\frac{cf(w)}{\|C\|}$ and $P(D) = \frac{\sum_{Q \in \mathbf{Q}} click(Q,D)}{N}$
14	$generate\_w2Q'$	$\log P(Q'\|w) = \log \frac{P_{lm}(w\|Q')P(Q')}{\sum_{Q_i' \in \mathbf{Q}} P_{lm}(w\|Q_i)P(Q)}$, where $P_{lm}(w\|Q) = (1-\alpha)\frac{tf(w;Q)}{\|Q\|} + \alpha\frac{cf(w)}{\|C\|}$ and $P(Q) = \frac{\sum_{D \in \mathbf{D}} click(Q,D)}{N}$

Table 1: Relations and their scoring functions used in the graph in Figure 1. Here, $tf(q;Q)$ is the number of times term q occurs in query Q, and $|Q|$ is the length of query Q. $tf(w;D)$ is the number of times term w occurs in D, and $|D|$ is the length of document D. The $cf(w)$ and $|C|$ values are analogously defined on the collection level, where the collection consists of all the documents in search logs. $P_{tm}(.)$ is word translation probability assigned by a translation model trained on query-title pairs derived from clickthrough data. $P_{tm}(q'|q)$ in #2 is also assigned by the same query-title translation model based on the assumption that a good expansion term q' is likely to occur in the titles of the clicked documents [19]. $click(Q',D)$ is the number of times document D is clicked for Q' in search logs. In #13, **D** is the full set of documents in search logs. **Q** in #12 and #14 is the full set of queries in search logs. N in #13 and #14 is the total number of clicks in search logs, i.e., $N = \sum_{Q \in \mathbf{Q}} \sum_{D \in \mathbf{D}} click(Q,D)$. Finally, α and β are model hyperparameters that control smoothing for query and document language models, respectively.

Each edge in the graph is labeled by a relation r, and is scored using a relation-specific model θ_r. The edge score is the probability of reaching target node t from source node s with a one-step random walk with edge type r, $P(t|s,\theta_r)$. The set of relations r and their corresponding scoring functions $score_{\theta_r}(s \to t)$, which are used in this study, are summarized in Table 1. To ensure that edge score is a probability, $P(t|s,\theta_r)$ is computed via softmax as

$$P(t|s,\theta_r) = \frac{\exp\left(score_{\theta_r}(s \to t)\right)}{\sum_{t_i} \exp\left(score_{\theta_r}(s \to t_i)\right)} \quad (2)$$

Note that in the original PCRW model [32] there is no θ_r, and the edge score is computed by

$$P(t|s,r) = \frac{\mathrm{I}(r(s,t))}{\sum_{t'} \mathrm{I}(r(s,t'))}$$

where $\mathrm{I}(r(s,t))$ is an indicator function that takes value 1 if there exists an edge with type r that connects s to t. Introducing θ_r allows us to easily incorporate well-established models that have been developed for QE and document ranking models in the IR community. The scoring functions in Table 1 lie in four categories. The first is the functions for the $similar\_*$ relation (e.g., #1), and is based on the BM25 model [39]. The second, including functions for the relations of $generate\_*$ (e.g., #5), uses unigram language models with Bayesian smoothing using Dirichlet priors [49]. The third, including functions for $click\_*$ (e.g., #3), uses a click model [13]. The last category, including functions for $translation\_*$ (e.g., #6), uses translation models [5, 17, 19], where, if clickthrough data is available for model training, the word translation probabilities P_{tm} are estimated on query-document pairs by assuming that a query is parallel to the documents clicked on for that query [17].

ID	Path type π (Comments)
TM1	$\langle translate\_Q2w \rangle$ (w is generated using clickthrough-based translation model from Q as in [19])
TM2	$\langle generate\_Q2w, generate\_w2D, generate\_D2w \rangle$ (variant of **TM1** where translation model is trained via 2-step random walks on word-document graph, as in [31])
TM3	$\langle generate\_Q2w, generate\_w2D, generate\_D2w, generate\_w2D, generate\_D2w \rangle$ (variant of **TM2** where 4-step random walks are used)
TM4	$\langle generate\_Q2w, generate\_w2Q', generate\_Q'2w \rangle$ (variant of **TM2** where random walks are performed on word-query graph)
TM5	$\langle generate\_Q2w, generate\_w2Q', generate\_Q'2w, generate\_w2Q', generate\_Q'2w \rangle$ (variant of **TM4** where 4-step random walks are used)
SQ1	$\langle similar\_Q2Q', generate\_Q'2w \rangle$ (w is generated from similar queries Q' of Q, where query similarity is based on BM25)
SQ2	$\langle translate\_Q2Q', generate\_Q'2w \rangle$ (variant of **SQ1** where query similarity is based on clickthrough-based translation model)
SQ3	$\langle similar\_Q2Q', click\_Q'2D, click\_D2Q', generate\_Q'2w \rangle$ (variant of **SQ1** where similar query set is enriched by 2-step random walks on query-document graph)
SQ4	$\langle similar\_Q2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, click\_D2Q', generate\_Q'2w \rangle$ (variant of **SQ3** where 4-step random walks are used)
SQ5	$\langle translate\_Q2Q', click\_Q'2D, click\_D2Q', generate\_Q'2w \rangle$ (variant of **SQ2** where similar query set is enriched by 2-step random walks on query-document graph)
SQ6	$\langle translate\_Q2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, click\_D2Q', generate\_Q'2w \rangle$ (variant of **SQ5** where 4-step random walks are used)
RD1	$\langle similar\_Q2Q', click\_Q'2D, generate\_D2w \rangle$ (w is generated from pseudo-relevant documents D clicked for similar queries Q' of Q)
RD2	$\langle translate\_Q2Q', click\_Q'2D, generate\_D2w \rangle$ (variant of **RD1** where query similarity is computed via translation model)
RD3	$\langle similar\_Q2Q', click\_Q'2D, translate\_D2w \rangle$ (variant of **RD1** where w is generated from D using translation model)
RD4	$\langle similar\_Q2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, generate\_D2w \rangle$ (variant of **RD1** where set of D is enriched by 2-step random walks on query-document graph)
RD5	$\langle similar\_Q2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, generate\_D2w \rangle$ (variant of **RD1** where 4-step random walks are used)
RD6	$\langle translate\_Q2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, generate\_D2w \rangle$ (variant of **RD2** where set of D is enriched by 2-step random walks on query-document graph)
RD7	$\langle translate\_Q2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, click\_D2Q', click\_Q'2D, generate\_D2w \rangle$ (variant of **RD6** where 4-step random walks are used)
RD8	$\langle click\_Q2D, generate\_D2w \rangle$ (w is generated from relevant documents D clicked for query Q)
RD9	$\langle click\_Q2D, click\_D2Q, click\_Q2D, generate\_D2w \rangle$ (variant of **RD8** where the set of D is enriched by 2-step random walks on query-document graph)
RD10	$\langle click\_Q2D, click\_D2Q, click\_Q2D, click\_D2Q, click\_Q2D, generate\_D2w \rangle$ (variant of **RD9** where 4-step random walks are used)

Table 2: Some examples of path types, each used as a feature in the PCRW model for QE.

3.2 Feature as Path Type

Given a graph, any path type π that starts with the input query node Q and ends with a word node w defines a real-value feature, which can be viewed as a QE model (or QE feature). The feature value is the probability of picking w as an expansion term $P(w|Q,\pi)$ by PCRWs of type π. In what follows we illustrate the capability of the PCRW model using examples in Table 2. We focus our discussion on three categories of QE features: (1) **TM** features, which perform QE using translation models (i.e., the corresponding path types are specified by IDs from **TM1** to **TM5** in Table 2), (2) **SQ** features, which perform QE using similar queries (i.e., **SQ1** to **SQ6**), and (3) **RD** features, which perform QE using (pseudo-)relevant documents (i.e., **RD1** to **RD10**).

Many log-based QE methods use clickthrough-based translation models where term correlations are pre-computed using query-document pairs extracted from clickthrough data [14, 19, 22]. Compared to the methods that are based on thesauri either compiled manually [38] or derived from document collections [28], such log-based methods are superior in that the translation models explicitly capture the correlation between query terms and document terms. One example is the word translation model described in [19], which can be encoded by the path type **TM1**, $\langle translation\_Q2w \rangle$. In case there is not (enough) clickthrough data for model training, Lafferty and Zhai [31] present a method using Markov chains, where the translation probability between two words is computed by random walks on a document-word graph. The method can be encoded by the path types of **TM2** and **TM3** in Table 2.

Rare queries present a big challenge for Web search [45]. The expansion of a rare query Q is often performed by adding terms from common queries Q' which are similar to Q [45]. The PCRW model achieves this by a random walk that instantiates the path type **SQ1**, $\langle similar\_Q2Q', generate\_Q'2w \rangle$. [6, 21, 35] show that (more) similar queries can be retrieved by performing random walks on a query-document click graph. Thus, rare query expansion could be improved using a larger set of similar queries identified by repeatedly applying random walks following the edges with types $click\_Q2D$ and $click\_D2Q$. **SQ3** and **SQ4** in Table 2 are two examples of such improved models.

A set of relevant documents D of an input query Q that is seen in the search logs can be formed by collecting all the documents

that have clicks for that query. Thus, the relevance feedback QE method can be represented as e.g., **RD8**,

$$\langle click\_Q2D, generate\_D2w \rangle.$$

If the input query is a rare query, we can form the set of pseudo-relevant documents through its similar queries Q' that are in the search logs, e.g., **RD1**,

$$\langle similar\_Q2Q', click\_Q'2D, generate\_D2w \rangle.$$

To conquer the data sparseness problem, more pseudo-relevant documents can be retrieved by performing random walks on a query-document click graph, such as **RD4** and **RD5** in Table 2.

Following previous work [12, 21, 31], in our experiments the random walks are implemented as matrix multiplication. As an example, we consider the task of retrieving similar queries by repeatedly applying random walks following $click\_Q2D$ and $click\_D2Q$. Let N be the number of query nodes in G and M be the number of document nodes. Let $\boldsymbol{A}$ be the $N \times M$ matrix with entries $\boldsymbol{A}_{Q,D} = P(D|Q)$, called query-document *transition matrix*, where the probability is calculated from clicks as in #3 in Table 1. Also, let $\boldsymbol{B}$ be the $M \times N$ matrix with entries $\boldsymbol{B}_{D,Q} = P(Q|D)$, where the probability is calculated from clicks as in #4 in Table 1. $\boldsymbol{A}$ and $\boldsymbol{B}$ are called *transition matrices*. It is easy to see that using $\boldsymbol{C} = \boldsymbol{AB}$ we can compute the probability of walking from an initial query Q_0 to any other query Q in $2k$ steps, and the corresponding probability, which is used to measure query-to-query similarity, is given by $P(Q|Q_0) = \boldsymbol{C}^k_{Q_0,Q}$. Because the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are sparse, the matrix product $\boldsymbol{C} = \boldsymbol{AB}$ can be computed efficiently. As k increases, $\boldsymbol{C}^k$ quickly becomes dense and the powers cannot be computed efficiently. However, as k increases, the search intent shifts from the initial query, as the probability quickly spreads out over all queries. Thus, in our experiments we limit k to 1 and 2.

3.3 QE as Path Ranking

For QE, we rewrite the PCRW model of Equation (1) as

$$P(w|Q) = \sum_{\pi \in B} \lambda_\pi P(w|Q, \pi), \tag{3}$$

which is weighted linear combination of path features π in B. Thus, the PCRW model performs QE by ranking a set of combined paths, each for one pair of Q and w (i.e., a candidate expansion term). This section presents the way B is constructed and the next two sections present the way parameters λ_π are estimated.

Given a graph, the total number of path types $|B|$ grows exponentially with the increase of path length. To make the computation feasible, in our experiments we set the maximum length to 7, and only consider a small set of relations that are highly selective, as shown in Table 1. Given a path type π, due to the large number of nodes in G, even with a length limit, the total number of paths that instantiate π could be extremely large. For example, since a word could translate to any other word based on a smoothed translation model, any node pair (Q', Q) would have a non-zero-score relation $translate\_Q2Q'$ (#2 in Table 1), thus making the transition matrix extremely dense. For efficiency, we keep the (multiplication of) transition matrices sparse by retaining only top-1000 (partial) paths after each step of random walk.

3.4 Training Data Generation

The training data used for the estimation of parameters λ_π in Equation (3) is denoted as $\{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x}_i$ is a vector of all the path features for the pair (Q_i, w_i). That is, the j-th component of $\mathbf{x}_i$ is $P(w_i|Q_i, \pi_j)$, and y_i is a Boolean variable indicating whether w_i is a good expansion term for Q_i. In our experiments D is generated using a method similar to [10], which will be described below.

Assume we have developed a relevance judgment set. The set consists of a set of queries. Each query is associated with a set of documents. Each query-document pair has a relevant label. The effectiveness of a document ranking model $Score(D, Q)$ can be evaluated on the set. We determine whether a word w is a good expansion for a query Q by examining whether expanding Q with w leads to a better document ranking result. Specifically, we use the following ranking model

$$Score(D, Q) = \alpha \log P(w|\theta_D) + \sum_{q \in Q} P(q|\theta_Q) \log P(q|\theta_D) \tag{4}$$

where w is the expansion term under consideration, α is its weight, q is a term in the original query Q, and θ_Q and θ_D are query and document models, respectively. The query model $P(q|\theta_Q)$ is estimated via MLE (maximum likelihood estimation) without smoothing as

$$P(q|\theta_Q) = \frac{tf(q; Q)}{|Q|} \tag{5}$$

where $tf(q; Q)$ is the number of times q occurs in Q, and $|Q|$ is the query length. The document model, e.g., $P(w|\theta_D)$, is estimated via MLE with Dirichlet smoothing as

$$P(w|\theta_D) = \frac{tf(w; D) + \mu P(w|C)}{|D| + \mu} \tag{6}$$

where $tf(w; D)$ is the number of times w occurs in D, $|D|$ is the document length, μ is the Dirichlet prior (set to 2000 in our experiments), and $P(w|C)$ is the probability of w on the collection C, estimated via MLE without smoothing.

Equation (4) can be viewed as a simplified form of QE with a single term. It is used to label whether w is a good expansion term for Q. To simplify the training data generation process, we assume that w acts on the query independently from other expansion terms, and each expansion term is added into Q with equal weight, i.e., $\alpha = 0.01$ or $\alpha = -0.01$.

The training data is generated as follows. For each query Q in the relevance judgment set, a set of candidate expansion terms $\{w_i\}$ is formed by collecting all terms that occur in the documents that are paired with Q but do not occur in Q. Then w_i is labeled as a good expansion term for Q if it improves the effectiveness of ranking document when $\alpha = 0.01$ and hurt the effectiveness when $\alpha = -0.01$. w_i is labeled as bad if it produces an opposite effect or produces similar effect when $\alpha = 0.01$ or $\alpha = -0.01$.

3.5 Parameter Estimation

Given training data $\{(\mathbf{x}_i, y_i)\}$, the model parameters $\boldsymbol{\lambda} = \langle \lambda_\pi \rangle_{\pi \in B}$ can be optimized by maximizing the following objective [32]

$$\mathcal{F}(\boldsymbol{\lambda}) = \sum_{(\mathbf{x}, y) \in \{(\mathbf{x}_i, y_i)\}} f(\mathbf{x}, y; \boldsymbol{\lambda}) - \alpha_1 \|\boldsymbol{\lambda}\|_1 - \alpha_2 \|\boldsymbol{\lambda}\|_2^2 \tag{7}$$

where α_1 and α_2 respectively control the strength of the L_1-regularization, which helps with structure selection, and L_2-regularization which helps prevent overfitting. $f(\mathbf{x}, y; \boldsymbol{\lambda})$ is the log-likelihood of the training sample $(\mathbf{x}, y)$, and is defined as

$$f(\mathbf{x}, y; \boldsymbol{\lambda}) = y \log P(\mathbf{x}, \boldsymbol{\lambda}) + (1 - y) \log(1 - P(\mathbf{x}, \boldsymbol{\lambda})) \quad (8)$$

and

$$P(\mathbf{x}, \boldsymbol{\lambda}) \equiv P(y = 1 | \mathbf{x}, \boldsymbol{\lambda}) = \frac{\exp(\boldsymbol{\lambda}^T \mathbf{x})}{1 + \exp(\boldsymbol{\lambda}^T \mathbf{x})} \quad (9)$$

is the model-predicted probability. In our experiments the maximization is performed using the OWL-QN algorithm [2], which is a special version of L-BFGS designed to deal with non-differentiable L_1 norm.

The PCRW-based model of Equation (3) assigns each path type a weight. Such a parameterization is called *one-weight-per-path-type*. An alternative way of parameterizing the model is *one-weight-per-edge-label* [11, 37]. [32] argue that the former is superior in that it takes into account the context in which a relation appears. In our experiments we compare these two parameterization options. Following [32], we use the same objective function and optimization procedure for the parameter estimation of the one-weight-per-edge-label model. Because the model can be seen as the combination of all the PCRWs with each path having its weight set to the product of all the edge weights along the path, we can calculate the gradient of edge weights by first calculating the gradient with respect to the paths, and then applying the chain rule of derivative.

4. EXPERIMENTS

4.1 Dataset and Evaluation Method

In this study the effectiveness of a QE method is evaluated by issuing a set of queries which are expanded using the method to a search engine and then measuring the Web search performance. Better QE methods are supposed to lead to better Web search results using the correspondingly expanded query set.

Due to the characteristics of our QE methods, we cannot conduct experiments on standard test collections such as the TREC data because they do not contain related search logs we need. Therefore, following previous studies of log-based QE [e.g., 14, 19, 40], we used the proprietary datasets that have been developed for building a commercial search engine, and demonstrated the effectiveness of our methods by comparing them against several state-of-the-art QE methods that are originally developed using TREC data [45, 34, 36]. For comparison, we also reproduced on our datasets the results of several previous state-of-the-art log-based QE methods [14, 19, 41].

Our relevance judgment set consists of 12,000 rare queries in English. On average, each query is associated with 33 Web documents (URLs). Each query-document pair has a relevance label. The label is human generated and is on a 5-level relevance scale, 0 to 4, with 4 meaning document D is the most relevant to query Q and 0 meaning D is not relevant to Q.

The relevance judgment set is constructed as follows. First, the rare queries are sampled from one day of search engine logs. Adult, spam, and bot queries are all removed. To reflex a natural distribution of rare queries, we do not try to control the quality of these queries. We found that in comparison with common queries, rare queries are longer and contain more spelling errors. For example, in our rare query set, the average query length is 5 (vs. 3-word for a common query set), and there are around 20% misspelled queries (vs. 12% for a common query set). Second, for each query, we collect Web documents to be judged by issuing the query to several popular search engines (e.g., Google, Bing) and fetching top-10 retrieval results from each. Finally, the query-document pairs are judged by a group of well-trained assessors. In this study all the queries are preprocessed as follows. The text is white-space tokenized and lowercased, numbers are retained, and no stemming/inflection treatment is performed. Since all the document ranking and QE models tested in our experiments contain free parameters that must be estimated empirically on data, we used two-fold cross validation to report results: a set of results on one half of the relevance judgment set is obtained using the parameter settings optimized on the other half, and global retrieval results are combined from those of the two sets.

The search logs used in this study consist of approximately 3 billion query-document pairs sampled from the search logs of a commercial search engine. The Web document collection consists of around 730 million Web pages. In the retrieval experiments we use the index based on the content fields (i.e., body and title text) of each Web page.

The performance of Web search is evaluated by mean *Normalized Discounted Cumulative Gain* (NDCG) [26]. We report NDCG scores at truncation levels 1, 3, and 10. We also performed a significance test, i.e., a t-test with a significance level of 0.05. A significant difference should be read as significant at the 95% level.

4.2 System

We constructed the graph G using the search logs described in Section 4.1. G consists of 730 million document nodes D, 1.8 billion query nodes Q', and 100 million word nodes w. To represent the rare queries in the relevant judgment set, we extend G by generated 12,000 input query nodes Q, each for one rare query. The edges between nodes are labeled using the relations defined in Table 1. Since rare queries are unseen in search logs, the edges between Q and D, as in Figure 1, have a zero score, and all path types that include zero-score edges are inactive, such as **RD8**, **RD9** and **RD10** in Table 2.

QE is performed as follows. Given a trained PCRW model and the node of an input query Q, we perform random walks in G following all possible paths that instantiate the path types defined in B. We then generate a list of candidate expansion term nodes w together with their scores $P(w|Q)$, as computed by Equation (3). We sort all the predictions (Q, w) by the scores in descending order, and pick the top-n words that are not in the input query for QE. In our experiments we set $n = 10 \times |Q|$, where $|Q|$ is the length of the input query. The terms in the expanded query are weighted using the method described in [45]. The weights of the terms in the original query are set to 2, and the weight of a new term is set to $1.0 - 0.9 \times i/n$, where i the rank of the term in the sorted list of top-n candidates.

We use the unigram language model with Dirichlet smoothing to perform document ranking [49]. The model is defined as the second term on the right-hand-side of Equation (4).

4.3 Main Results

Table 3 summarizes the main results using different QE methods, evaluated on the relevance judgment set described in Section 4.1.

#	QE Methods	NDCG@1	NDCG@3	NDCG@10
1	**NoQE**	0.2648	0.2985	0.3905
2	**LCA (PRF)**	0.2742^{α}	0.3107^{α}	0.4075^{α}
3	**RM (PRF)**	0.2689^{α}	0.3077^{α}	0.4068^{α}
4	**LCE**	0.2695^{α}	0.3069^{α}	0.4098^{α}
5	**TC**	$0.2811^{\alpha\beta}$	$0.3198^{\alpha\beta}$	$0.4132^{\alpha\beta}$
6	**SMT**	$0.2803^{\alpha\beta}$	$0.3230^{\alpha\beta\gamma}$	$0.4199^{\alpha\beta\gamma}$
7	**TM**	$0.2837^{\alpha\beta}$	$0.3212^{\alpha\beta}$	$0.4183^{\alpha\beta\gamma}$
8	**PCRW**	$0.2959^{\alpha\beta\gamma}$	$0.3302^{\alpha\beta\gamma}$	$0.4265^{\alpha\beta\gamma}$

Table 3: Document ranking results using different QE methods. The superscripts α, β, and γ indicate statistically significant improvements ($p < 0.05$) over **NoQE**, **LCA**, and **TC**, respectively.

Row 1 in Table 3 (i.e., **NoQE**) is the baseline that uses the raw input queries without expansion. Rows 2 to 6 are the QE methods proposed previously. For fair comparison, the number of expansion terms for a query Q is set to $n = 10 \times |Q|$ for all QE methods.

LCA (Row 2) is local context analysis [45]. **RM** (Row 3) is relevance model [34]. LCA and RM are state-of-the-art PRF methods, developed respectively for the vector space and language modeling IR frameworks. **LCE** (Row 4) is latent concept expansion [36], which is a generalization of RM in that it explicitly models term dependencies for QE. Unfortunately, the generalization does not lead to any significant improvement in our experiments (Row 4 vs. Rows 2 and 3).

TC (Row 5) is the log-based QE method based on our implementation of the term correlation model [14]. We see that both **TC** and PRF methods improve the effectiveness of Web search significantly, and the log-based method outperforms significantly the RPF methods that do not use query logs. The results confirm the conclusion of [14].

SMT (Row 6) is a statistical machine translation (SMT) based QE system. Following Riezler et al. [41], the system is an implementation of a standard phrase-based SMT system with a set of features derived from a translation model and a language model, combined under the log-linear model framework [29]. To apply the system to QE, expansion terms of a query are taken from those terms in the 10-best translations of the query that have not been seen in the original query string. The results show that **SMT** is also effective (Row 6 vs. Row 1), outperforming significantly **TC** in NDCG at 3 and 10 (Row 6 vs. Row 5). This result is more or less consistent with what is reported in Riezler et al. [41], despite the difference in training data we used, (i.e., Riezler et al. used query-snippet pairs while we used query-title pairs).

TM (Row 7) is the QE method using a clickthrough-based translation model [17, 19]. **TM** and **TC** models are trained on the same clickthrough data that consists of 3 billion query-title pairs. The result that **TM** outperforms **TC** confirms the conclusion of [19] that a translation model trained using the EM algorithm [8, 16] is better than a correlation model estimated purely based on frequency counting as in **TC**.

TC, **SMT** and **TM**, considered as state-of-the-art QE methods, have been frequently used for comparison in related studies.

Row 8 is the PCRW-based method, described in Section 3. It outperforms significantly the baseline (Row 1) and the other QE methods we used for comparison (Rows 2 to 6). Since the PCRW model combines a wide variety of features, each encoded by a path type, it is instructive to investigate the QE performance of individual features and the impact of how these features are combined.

ID (path length)	NDCG@1	NDCG@3	NDCG@10
NoQE	0.2648	0.2985	0.3905
PCRW	0.2959	0.3302	0.4265
TM1 (1)	***0.2837***	***0.3212***	***0.4183***
TM2 (3)	0.2714	0.3101	0.3995
TM3 (5)	0.2705	0.3100	0.3989
TM4 (3)	0.2680	0.3050	0.4031
TM5 (5)	0.2688	0.3051	0.4030
SQ1 (2)	0.2768	0.3139	0.4084
SQ2 (2)	0.2793	0.3159	0.4127
SQ3 (4)	***0.2806***	***0.3164***	***0.4109***
SQ4 (6)	0.2793	0.3146	0.4103
SQ5 (4)	0.2796	0.3151	0.4107
SQ6 (6)	0.2778	0.3140	0.4102
RD1 (3)	0.2844	0.3200	0.4133
RD2 (3)	***0.2893***	***0.3242***	***0.4167***
RD3 (3)	0.2871	0.3245	0.4213
RD4 (5)	0.2873	0.3221	0.4163
RD5 (7)	0.2870	0.3234	0.4175
RD6 (5)	0.2841	0.3194	0.4122
RD7 (7)	0.2835	0.3176	0.4119

Table 4: Document ranking results using different QE features, each encoded by a path type whose ID is defined in Table 2. The IDs **NoQE**, **PCRW** and **TC** are defined in Table 3.

4.4 Individual Features

Recall that in Section 3.2 we group path types into three categories: (1) **TM** features, (2) **SQ** features, and (3) **RD** features. They generate expansion terms using different data sources, and thus are expected to be complimentary. Table 4 presents QE results using individual features, where the best feature in each category is in **bold** and *italic*. Comparing the results of individual features with that of the PCRW model reveals that combining features significantly improves the QE performance.

Figures 2 and 3 present respectively two example queries, where different QE features give complimentary expansion terms and the combined achieves the best result. The query "acme baked bread" in Figure 2 is issued to search for the homepage of a bakery company in Berkeley, CA. The expanded query based on clickthrough-based translation model (**TM1** in Table 2) leads to worse document ranking results than that of **NoQE** because the model generates expansion terms from query terms in a word-by-word fashion, e.g., generating "bakery" or "bread" from "baked". But, without knowing that the entire query refers to an entity (i.e., company), it cannot generate expansion terms relating to the properties of the entity (e.g., the location of the company). However, using features based on similar queries (**SQ1**) location names such as "san francisco" and "berkeley" are selected as expansion terms. It is also encouraging to see that its relevant document (www.acmebread.com) is ranked top in its pseudo-relevant document set obtained via similar queries (**RD1**).

The query "waterfall glass in dallax tx" in Figure 3 contain two common terms "waterfall" and "glass". The search intent suggested by the two terms when they occur in the same query is very different from that when only one of them occurs. As expected, the QE method using a word translation model (**TM1**) fails to improve the search performance. Neither do the similar queries retrieved via random walks (**SQ1** and **SQ3**) provide very useful expansion terms since most of the similar queries are simply different permutations

recipe 0.03239	company 0.00280	company 0.00280
recipes 0.01685	co 0.00203	markets 0.00199
bake 0.01675	berkeley 0.00203	groceries 0.00143
oven 0.00886	recipes 0.00628	coupons 0.00143
baking 0.00705	sf 0.00203	weekly 0.00143
cooks 0.00611	oven 0.00395	ad 0.00143
company 0.00598	bakery 0.00167	recipes 0.00825
html 0.00543	home 0.00280	pharmacy 0.00143
food 0.00451	recipe 0.00280	bakery 0.00143
set 0.00439	san 0.00280	grocery 0.00167
bakery 0.00432	francisco 0.00280	stores 0.00167
breads 0.00324	fresh 0.00280	homemade 0.00143
rec 0.00317	ferry 0.00203	oven 0.00395
search 0.00293	building 0.00203	baked 0.00729
make 0.00289	garlic 0.00203	yeast 0.00143
ff 0.00272	pudding 0.00280	breads 0.00239
home 0.002168	pita 0.00203	ehow 0.00333
markets 0.0018	mountain 0.00203	grandmothers 0.0017
honeybaked 0.0014	food 0.00167	recipe 0.00283
… …	… …	… …
(a)	(b)	(c)

acme bread; acme bread company; acme bread co; acme bread berkeley; acme recipes; acme bread sf; baked bread recipes; oven baked bread; acme bakery; home baked bread; oven baked bread recipe; acme bread san francisco; oven baked bread recipes; home baked bread recipes;
acme bread company san francisco; acme bread ferry building
… …

(d)

1. the acme bread company
   http://www.acmebread.com
2. acme markets groceries coupons weekly ad recipes and pharmacy
   http://www.acmemarkets.com
3. bakery acme markets grocery stores
   http://www.acmemarkets.com/departments/bakery.jsp
4. acme bread company
   http://www.ferrybuildingmarketplace.com/acme_bread_company.php
5. homemade oven baked yeast breads
   http://baking.about.com/od/yeastbreads/Yeast_Breads.htm
6. how to make home baked bread ehow com
   http://www.ehow.com/how_4794468_home-baked-bread.html
7. grandmothers oven baked bread recipe best recipes
   http://www.bestrecipes.com.au/recipe/...
8. how to bake bread bread recipes healthy breads
   http://bread-by-yia-yia.com
9. bread baking
   http://breadbaking.about.com
10. bread recipes allrecipes com
    http://allrecipes.com/Recipes/Bread

… …

(e)

Figure 2: QE results of $Q =$ **acme baked bread.** (a), (b) and (c) are top expansion terms and their scores $P(w|Q)$ generated using features **TM1**, **SQ1** and **RD1**, respectively; (d) are top similar queries generated using **SQ1**; (e) are top pseudo relevant documents generated using **RD1**. Features are defined in Table 2.

texas 0.05671	lounge 3.993E-05	windsor 7.85E-05
waterfalls 0.01893	house 3.424E-05	texas 6.19E-05
falls 0.01093	club 3.424E-05	apartments 4.69E-05
water 0.00592	uptown 2.802E-05	house 4.69E-05
city 0.00401	auto 2.031E-05	communities 4.69E-05
home 0.00362	stained 2.031E-05	real 2.39E-05
dfw 0.00267	bar 2.031E-05	estate 2.39E-05
fall 0.00258	blowing 2.031E-05	travel 2.39E-05
worth 0.00255	door 2.031E-05	hotels 2.10E-05
center 0.00238	oregon 2.031E-05	profile 2.03E-05
fort 0.00220	coldplay 2.031E-05	population 2.03E-05
company 0.00191	hiking 2.031E-05	maps 2.03E-05
wikipedia 0.00187	flat 2.031E-05	averages 2.03E-05
wiki 0.00186	tile 2.031E-05	homes 2.03E-05
park 0.00186	nightclub 2.031E-05	statistics 2.03E-05
houston 0.00185	… …	relocation 2.03E-05
county 0.00168		hospitals 2.03E-05
glasses 0.00158		restaurants 1.99E-05
north 0.00157		events 1.85E-05
… …		… …
(a)	(b)	(c)

glass dallas; dallas glass; glass dallas tx; waterfall glass; glass waterfall; dallas auto glass; glass house dallas; stained glass dallas; glass dallas club; glass club dallas; glass lounge dallas tx; the glass house dallas; glass bar dallas; dallas glass blowing; glass dallas uptown; dallas glass and door; dallas glass oregon; coldplay dallas tx; hiking dallas tx; glass lounge dallas
… …

(d)

1. welcome to the city of dallas texas city web portal
   http://dallascityhall.com/
2. dallas wikipedia the free encyclopedia
   http://en.wikipedia.org/wiki/Dallas
3. dallas hotels restaurants events and things to do dallas cvb
   http://www.visitdallas.com
4. dallas tx apartments glass house by windsor windsor communities
   http://www.windsorcommunities.com/apartments/dallas/glasshouse
5. dallas texas tx profile population maps real estate …
   http://www.city-data.com/city/Dallas-Texas.html
6. dallas city guide hotels restaurants nightlife attractions real estate
   http://www.dallas.com
7. glass lounge uptown dallas tx
   http://www.yelp.com/biz/glass-lounge-dallas
8. glass uptown uptown dallas tx
   http://www.yelp.com/biz/glass-uptown-dallas
9. the dallas glass club dallas texas dgc home
   http://dallasglassclub.org/
10. glass uptown lounge website coming soon
    http://glassuptown.com/

… …

(e)

Figure 3: QE results of $Q =$ **waterfall glass in dallas tx.** (a), (b) and (c) are top expansion terms their scores $P(w|Q)$ generated using features **TM1**, **SQ1** and **RD1**, respectively; (d) are top similar queries generated using **SQ1**; (e) are top pseudo relevant documents generated using **RD1**. Features are defined in Table 2.

of the same set of terms. Fortunately, as shown in Figure 3 (e), these permuted queries lead to a set of clicked documents (**RD1**) from which effective expansion terms are generated.

Results in Table 4 reveal the effectiveness of individual features, some of which have not been studied previously. **TM1**, which is also reported in Row 7 in Table 1, is the best among all **TM** features. Although the translation probabilities in **TM2**, **TM3**, **TM4** and **TM5** are estimated via random walks, rather than on query-document pairs as in **TM1**, these models still improve the baseline **NoQE**, although not as effective as **TM1**, thus providing an alternative way of obtaining translation models without training data.

Parameterization	NDCG@1	NDCG@3	NDCG@10
untrained (λ_π=1)	0.2910	0.3263	0.4233
one-weight-per-edge-label	0.2924	0.3277	0.4249
one-weight-per-path-type	0.2959	0.3302	0.4265

Table 5: Comparison with different parameterization methods.

Results of the **SQ** features suggest that (1) it is more effective to retrieve *semantically* similar queries for QE, and this can be achieved either by applying a translation model (**SQ2** vs. **SQ1**) or by applying random walks on query-document graph (**SQ3** vs. **SQ1**); and (2) taking a 2-step random walk is useful but taking longer steps is not (e.g., **SQ3** vs. **SQ1** and **SQ4**).

Results of the **RD** features show that (1) it is more effective to use a translation model to retrieve similar queries (**RD2** vs. **RD1**) or to generate expansion terms from pseudo-relevant documents (**RD3** vs. **RD1**); and (2) random walks cannot significantly improve the quality of the pseudo-relevant document set (e.g., **RD6** and **RD7** vs. **RD2**).

4.5 Impact of Parameter Estimation

Table 5 compares the PCRW model parameterized using one-weight-per-path-type with two baselines. We see that (1) the trained models outperform the untrained model; and (2) one-weight-per-path-type is slightly better than one-weight-per-edge-label, but the difference is not statistically significant, except for NDCG score at 1, indicating that capturing context information between relations in a path is useful, although the impact on QE is marginal.

5. RELATED WORK

Our work is a significant extension to the random walk models described in [11] in two aspects. First, while we use a PCRW model, [11] uses a more traditional Markov chain model, similar to [23, 37, 43], where random walks are not *constrained* by path types and their models are parameterized as one-weight-per-edge-label. As discussed in Section 3.5, paths in G provide more useful features for QE than edges since the former captures more context information. Although it is difficult for us to perform a direct comparison between our model and the model in [11] due to the dramatically different data sources that these two models are based on respectively, the result in Table 5 suggests that given the same search logs as thesauri, our model is likely to perform better. Second, while G in our model is constructed on search logs, G in [11] is constructed on thesauri that are compiled manually or derived from document collections. Our design decision of using search logs rather than pre-complied thesauri is motivated by those studies that show that log-based QE methods [e.g., 15, 19, 22, 42] often lead to a superior performance to the QE methods that use human-compiled thesauri [e.g., 24, 38] largely due to the fact that models trained on search logs explicitly capture the correlation between query terms and document terms, thus bridging the lexical gap between them more effectively. On the other hand, Web scale thesauri such as ConceptNet and Wikipedia have recently been explored for QE, leading to some promising results [30, 47]. The graph representation and the PCRW-based inference we proposed provide a flexible framework to incorporate such new thesauri. We leave it to future work.

In addition to QE, random walk models have also been applied on other Web search applications, such image search [12], query suggestion [6, 35], query translation for cross-lingual IR [9], click model smoothing [21], and email search [37]. Our method bears some resemblance to all these previous works.

Previous studies on QE can be roughly grouped into two categories: the automatic relevance feedback methods [10, 11, 34, 36, 40, 45, 48] developed mainly on TREC data and the log-based methods [14, 15, 19, 22, 41, 42] where the correlation between query terms and document terms is learned from clickthrough data. Most of the features used in our PCRW model, as in Table 2, are inspired by these QE models.

Search logs have been proved to be a valuable data source for many Web search tasks. In addition to QE, they have also been used for document ranking [1, 20, 27], query processing and spelling correction [18, 25], user query clustering [3, 4, 44], etc.

6. CONCLUSIONS

This paper exploits search logs for QE for Web search ranking. We present a QE method based on path-constrained random walks, where the search logs are represented as a labeled, directed graph, and the probability of selecting an expansion term for an input query is computed by a learned combination of constrained random walks on the graph. We show that our method is generic and flexible in that it not only represents most of popular QE models as features, but also allows us to easily devise new features, which can potentially use much richer information than previous QE models, by defining path types with a rich set of walk behaviors. The PCRW model also provides a principled mathematical framework in which different QE models, i.e., defined as path types or features, can be incorporated in a unified way, thus making it less susceptible to the sparseness issue of clickthrough data and ambiguous search intent of user queries. The evaluation on a real-world data set shows that the PCRW-based method significantly outperforms other state-of-the-art QE methods.

One area in future work is to adapt the PCRW-based method for Web document ranking directly. For example, we might model the relevance score of a query Q and a document D as the probability, computed by a learned combination of path-constrained random walks from Q to D, where different document ranking models can be incorporated as path types. In addition to clickthrough data, we need to incorporate other data source to construct G, such as link graphs and the category structure of Web documents.

7. REFERENCES

[1] Agichtein, E., Brill, E., and Dumais, S. 2006. Improving web search ranking by incorporating user behavior information. In *SIGIR*, pp. 19-26.

[2] Andrew, G., and Gao, J. 2007. Scalable training of L1-regularized log-linear models. In *ICML*.

[3] Baeza-Yates, R. 2007. Graphs from search engine queries. In Jan van Leeuwen et al. (Eds.): *SOFSEN 2007*, LNCS 4362, pp. 1-8.

[4] Baeza-Yates, R., and Tiberi, A. 2007. Extracting semantic relations from query logs. In *SIGKDD*, pp. 76-85.

[5] Berger, A., and Lafferty, J. 1999. Information retrieval as statistical translation. In *SIGIR*, pp. 222-229.

[6] Boldi, P., Bonchi, F., Castillo, C., Donato, D., and Vigna, S. 2009. Query suggestions using query-flow graphs. In *WSDM'09*.

[7] Broder, A., Ciccolo, P., Gabrilovich, E., Josifovski, V., Metzler, D., Riedel, L., and Yuan, J. 2009. Online expansion of rare queries for sponsored search. In *WWW 2009*.

[8] Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2): 263-311.

[9] Cao, G., Gao, J., Nie, J-Y., and Bai, J. 2007. Extending query translation to cross-language query expansion with Markov chain models. In *CIKM'07*.

[10] Cao, G., Nie, J-Y., Gao, J., and Robertson, S. 2008. Selecting good expansion terms for pseudo-relevance feedback. In *SIGIR*, pp. 289-305.

[11] Collins-Thompson, K., and Callan, J. 2005. Query expansion using random walk models. In *CIKM'05*.

[12] Craswell, N., and Szummer, M. 2007. Random walks on the click graph. In *SIGIR'07*, pp. 239-246.

[13] Craswell, N. Zoeter, O., Taylor, M. J., and Ramsey, B. 2008. An experimental comparison of click position-bias models. In WSDM 2008, pp. 87-94.

[14] Cui, H., Wen, J-R., Nie, J-Y. and Ma, W-Y. 2002. Probabilistic query expansion using query logs. In *WWW*, pp. 325-332.

[15] Cui, H., Wen, J-R., Nie, J-Y. and Ma, W-Y. 2003. Query expansion by mining user log. *IEEE Trans on Knowledge and Data Engineering.* Vol. 15, No. 4. pp. 1-11.

[16] Dempster, A., Laird, N., and Rubin, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39: 1-38.

[17] Gao, J., He, X., and Nie, J-Y. 2010. Clickthrough-based translation models for web search: from word models to phrase models. In *CIKM*, pp. 1139-1148.

[18] Gao, J., Li, X., Micol, D., Quirk, C., and Sun, X. 2010. A large scale ranker-based system for query spelling correction. In *COLING*, pp. 358-366.

[19] Gao, J., Nie, J-Y. 2012. Towards concept-based translation models using search logs for query expansion. In *CIKM*.

[20] Gao, J., Toutanova, K., and Yih, W-T. 2011. Clickthrough-based latent semantic models for web search. In *SIGIR*, pp. 675-684.

[21] Gao, J., Yuan, W., Li, X., Deng, K., and Nie, J-Y. 2009. Smoothing clickthrough data for web search ranking. In *SIGIR*, pp. 355-362.

[22] Gao, J., Xie, S., He, X., and Ali, A. 2012. Learning lexicon models from search logs for query expansion. In *EMNLP*.

[23] Haveliwala, T. H. 2002. Topic-sensitive pagerank. In *WWW*, pp. 517-526.

[24] Hovy, E., Gerber, L., Hermjakob, U. Junk, M., and Lin, C-Y. 2000. Question answering in webclopedia. In *TREC 9*.

[25] Huang, J., Gao, J., Miao, J., Li, X., Wang, K., and Behr, F. 2010. Exploring web scale language models for search query processing. In *WWW*, pp. 451-460.

[26] Jarvelin, K. and Kekalainen, J. 2000. IR evaluation methods for retrieving highly relevant documents. In *SIGIR*, pp. 41-48

[27] Joachims, T. 2002. Optimizing search engines using clickthrough data. In *SIGKDD*, pp. 133-142.

[28] Jing, Y., and Croft., B. 1994. An association thesaurus for information retrieval. In *RIAO*, pp. 146-160.

[29] Koehn, P., Och, F., and Marcu, D. 2003. Statistical phrase-based translation. In *HLT/NAACL*, pp. 127-133.

[30] Kotov, A., and Zhai, C. 2012. Tapping into knowledge base for concept feedback: leveraging conceptnet to improve search results for difficult queries. In *WSDM*.

[31] Lafferty, J., and Zhai, C. 2001. Document language models, query models, and risk minimization for information retrieval. In *SIGIR'01*, pp. 111-119.

[32] Lao, N., and Cohen, W. W. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 81:53-67.

[33] Lao, N., Subramanya, A., Pereira, F., Cohen, W. W. 2012. Reading the web with learned syntactic-semantic inference rules. In *EMNLP-CoNLL-2012*, pp. 1017-1026.

[34] Lavrenko, V., and Croft, B. 2001. Relevance-based language models. In *SIGIR*, pp. 120-128.

[35] Mei, Q., Zhou, D., and Church, K. 2008. Query suggestion using hitting time. In *CIKM'08*.

[36] Metzler, D., and Croft, B. 2007. Latent concept expansion using markov random fields. In *SIGIR'07*, pp. 311-318.

[37] Minkov, E., Cohen, W. W., and Ng, A. Y. 2006. Contextual search and name disambiguation in email using graphs. In *SIGIR'06*.

[38] Prager, J., Chu-Carroll, J., and Czuba, K. 2001. Use of Wordnet hypernyms for answering what is questions. In *TREC 10*.

[39] Robertson, S., and Zaragoza, H. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, Vol. 3, No. 4 (2009) 333-389.

[40] Rocchio, J. 1971. Relevance feedback in information retrieval. In *The SMART retrieval system: experiments in automatic document processing*, pp. 313-323, Prentice-Hall Inc.

[41] Riezler, S., Liu, Y. and Vasserman, A. 2008. Translating queries into snippets for improving query expansion. In *COLING 2008*. 737-744.

[42] Riezler, S., and Liu, Y. 2010. Query rewriting using monolingual statistical machine translation. *Computational Linguistics*, 36(3): 569-582.

[43] Toutanova, K., Manning, C. D., and Ng, A. Y. 2004. Learning random walk models for inducing word dependency distributions. In *ICML*.

[44] Wen, J., Nie, J-Y., and Zhang, H. 2002. Query clustering using user logs. *ACM TOIS*, 20(1): 59-81.

[45] Xu, J., and Croft, B. 1996. Query expansion using local and global document analysis. In *SIGIR*.

[46] Xu, J., and Xu, G. 2011. Learning similarity function for rare queries. In *WSDM'11*.

[47] Xu, Y. Jones, G.J.F., and Wang, B. 2009. Query dependent pseudo-relevance feedback based on Wikipedia. In *SIGIR*.

[48] Zhai, C., and Lafferty, J. 2001a. Model-based feedback in the kl-divergence retrieval model. In *CIKM*, pp. 403-410.

[49] Zhai, C., and Lafferty, J. 2001b. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR*, pp. 334-342.