Context-Aware Query Recommendation by Learning High-Order Relation in Query Logs

Xiaohui Yan, Jiafeng Guo, Xueqi Cheng
Institute of Computing Technology, CAS
Beijing, China 100190
{yanxiaohui, guojiafeng}@software.ict.ac.cn, cxq@ict.ac.cn

ABSTRACT

Query recommendation has been widely used in modern search engines. Recently, several context-aware methods have been proposed to improve the accuracy of recommendation by mining query sequence patterns from query sessions. However, the existing methods usually do not address the ambiguity of queries explicitly and often suffer from the sparsity of the training data. In this paper, we propose a novel context-aware query recommendation approach by modeling the high-order relation between queries and clicks in query log, which captures users' latent search intents. Empirical experiment results demonstrate that our approach outperforms the baseline methods in providing high quality recommendations for ambiguous queries.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation*

General Terms

Algorithms, Experimentation, Theory

Keywords

Context-Aware, High-Order Model, Search Intent

1. INTRODUCTION

Tradition query recommendation approaches often find similar queries to suggest for each other[2, 1]. However, since queries are usually very short and ambiguous[6], a same query may convey totally different search intents in different cases. For example, the query "saturn" may either refer to a planet name or a car manufacturer. To only suggest similar queries based on the current query may not work well in such cases.

Recently, some context-aware approaches have been proposed to address the problem of recommendation for ambiguous queries[5, 3]. The basic idea of these methods is to leverage the query sequence patterns from query sessions [1]. That is, if a user queried "solar system" recently, it would be naturally recommend "planet saturn" or "uranus" to him when he is searching "saturn".

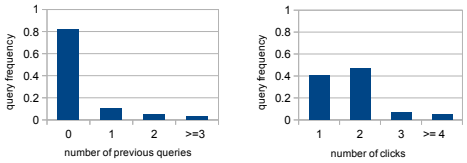

Figure 1: Query frequency with (a) different number of previous queries in the same session, (b) different number of clicks in query logs

However, these context-aware methods based on query sequence patterns matching have two major weaknesses. Firstly, they do not address the ambiguity of queries explicitly. Queries are usually viewed as the basic units in previous approaches without modeling their underlying search intents. Secondly, they mainly rely on the query sequence patterns existing in query logs, hence may suffer from the sparsity of data. We examined one month query logs from a commercial search engine. As Figure 1(a) shows, the number of query sequence patterns drops dramatically as their length increasing. Even worse, about 80% queries come from single-query sessions which means no previous queries existed.

On the contrary, we find that the click-through from a query often provides rich information that can clarify user's search intent for ambiguous queries. For example, when a user submits a query "saturn" , the search engine may return pages about both the Saturn planet and Saturn cars. Once the user clicking a URL in result pages about planet, it's clear that he cares about Saturn planet. Compared to the previous queries, the click-through from the current query always plays more direct effect on user's next search behavior Besides, we also find the click-through information is often more abundant than query sequence patterns in query logs, as Figure 1(b) shows, more than half of queries has one or more clicks, which may alleviate the sparsity problem.

Motivated by the above observations, we propose a probabilistic context-aware query recommendation approach by modeling the high-order relations between the current query, its click and the next query for recommendation. In our model, the ambiguity of queries can be resolved by intro-

[1]A query session is regard as a sequence of queries and clicks, performed by a user within a short period of time (e.g. 30 minutes) in a search process

CIKM'11, October 24–28, 2011, Glasgow, Scotland, UK.

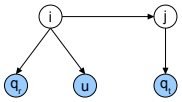

Figure 2: Graphical structure for a RUT triple $\langle q_r, u, q_t \rangle$.

ducing the latent factors over queries and their clicks. Once we have learned the fine-grained search intents underlying a query and its click-through, we can generate more accurate context-aware recommendation.

2. PROBLEM SETTING

To facilitate the discussion, we will first define some terminologies, and then give a formulation of our problem.

Definition 1. if a query q_t closely follows q_r in a query session, we denote q_r as a *refer query* (of q_t), and q_t as a *target query* (of q_r). All of the refer queries in query logs form the *refer query set* Q_r. Correspondingly, we define the *target query set* Q_t as the set of all target queries in query logs.

Definition 2. if a query q_t closely follows q_r in a query session, and a URL u is a click-through with q_r in this session, we call the triple $\langle q_r, u, q_t \rangle$ a *RUT* triple.

From a probabilistic view, the query recommendation task is to find top ranked q_ts with respect to the probability $P(q_t|q_r, u)$ for recommendation, given the contextual information $\langle q_r, u \rangle$. Because $P(q_r, u)$ is fixed when $\langle q_r, u \rangle$ is given, the condition probabilty is proportional to the joint one $P(q_r, u, q_t)$. Therefore, the remaining problem is how to estimate the joint probability $P(q_r, u, q_t)$. Now we summarize the objective of our model as follows:

Problem Statement 1. Given a set of RUT triples and their frequencies, the objective is to learn the probability $P(q_r, u, q_t)$, for $q_r \in Q_r$, $q_t \in Q_t$ and $u \in U$, where U is the URLs set in query logs.

3. OUR APPROACH

3.1 The High-Order Model

To estimate the probability $P(q_r, u, q_t)$, we propose a generative model which captures the high-order relations between user's current query, click and the next query. The graphical structure of our model is shown in Figure 2. It explains a query refinement process as follows. At first, a user has search intent i with probability $P(i)$. He submits query q_r with probability $P(q_r|i)$, and then clicks a URL in result pages with probability $P(u|i)$. Depending on his last search intent, he derives another related search intent j with probability $P(j|i)$, and formulate the next query q_t with probability $P(q_t|j)$. Here we name search intent i as *refer intent* (of search intent j), and search intent j as *target intent* (of search intent i) for convenience.

Based on the above generative model, we can write the joint distribution as

$$P(q_r, u, q_t) = \sum_{i,j} P(q_r|i)P(u|i)P(i,j)P(q_t|j). \quad (1)$$

Furthermore, we assume all the RUT triples follow an identical independent distribution. The likelihood of all the observed data then can be written as

$$l = \prod_{q_r,u,q_t} \left[\sum_{i,j} P(q_r|i)P(u|i)P(i,j)P(q_t|j) \right]^{n(q_r,u,q_t)}, \quad (2)$$

with parameters $\Theta = \{P(q_r|i), P(u|i), P(i,j), P(q_t|j)\}$.

3.2 Parameters Estimation

The parameters Θ can be learned by the Expectation Maximization (EM) algorithm[4]. By standard calculates one arrives at the following E-step and M-step, which will be repeated alternatively until convergence.

E-step:

$$P(i,j|q_r,u,q_t) = \frac{P(i,j)P(q_r|i)P(u|i)P(q_t|j)}{\sum_{i',j'} P(i,j)P(q_r|i')P(u|i')P(q_t|j')}, \quad (3)$$

M-step:

$$P(i,j) = \frac{\sum_{q_r,u,q_t} n(q_r,u,q_t)P(i,j|q_r,u,q_t)}{\sum_{q'_r,u',q'_t} n(q'_r,u',q'_t)}, \quad (4)$$

$$P(q_r|i) = \frac{\sum_{u,q_t,j} n(q_r,u,q_t)P(i,j|q_r,u,q_t)}{\sum_{q'_r,u',q'_t,j'} n(q'_r,u',q'_t)P(i,j'|q'_r,u',q'_t)}, \quad (5)$$

$$P(u|i) = \frac{\sum_{q_r,q_t,j} n(q_r,u,q_t)P(i,j|q_r,u,q_t)}{\sum_{q'_r,u',q'_t,j'} n(q'_r,u',q'_t)P(i,j'|q'_r,u',q'_t)}, \quad (6)$$

$$P(q_t|j) = \frac{\sum_{q_r,u,i} n(p_r,u,p_t)P(i,j|q_r,u,q_t)}{\sum_{q'_r,u',q'_t,i'} n(q'_r,u',q'_t)P(i',j|q'_r,u',q'_t)}. \quad (7)$$

3.3 Model Reduction

However, the EM algorithm in this problem may not be very efficient because of the extremely high dimensions of queries and URLs. In fact, we may not necessarily rely on the specific clicked URL u to identify the refer intent. We can leverage the topic or semantic category of the clicked URL u for the same purpose. Therefore, we map the URLs to their semantic categories c, whose dimensions are much lower than the URLs.

Here we use the Open Directory Project[2] data for the URL-category mapping. In ODP, a URL might be assigned to more than one categories. Hence, a URL u corresponds to a vector $\vec{c}$ in the category space as follows

$$u \to \vec{c} = (w_{u,c_1}, w_{u,c_2}, ..., w_{u,c_n}),$$

where $w_{u,c}$ is the weight of each category for u defined by:

$$w_{u,c_i} = \begin{cases} 0 & \text{if u not belong to the ist category } c_i \\ 1/k & k \text{ is the number of non-zero elements in } \vec{c} \end{cases}.$$

Finally, we transform each RUT triple to *RCT* triples (Refer query, Category, Target query) as

$$\langle q_1, u, q_2 \rangle \to \begin{cases} \langle q_1, c_1, q_2 \rangle \\ ... \\ \langle q_1, c_n, q_2 \rangle \end{cases}.$$

The frequency of a RCT triple, denoted by $n(\cdot)$, can be calculated as the product of the weight and the frequency of

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

the corresponding RUT triple

$$n(q_1, c, q_2) = n(q_1, u, q_2) \times w_{u,c}. \quad (8)$$

We can then obtain a similar high-order model over the RCT triples and estimate the corresponding parameters with a much lower cost.

3.4 Query Recommendation Generation

With the above learned models, we now describe how to generate recommendations for a given context pair $\langle q_r, u\rangle$. Our approach is based on the assumption that the URL is conditionally independent on queries given the categories of a URL. In this way, $P(q_t|q, u)$ can be calculated as:

$$P(q_t|q_r, u) = \frac{\sum_{c\in C} P(u|c)P(q_r, c, q_t)}{P(q_r, u)} \quad (9)$$

By considering

$$P(u) = \frac{n(u)}{\sum_{u\in U} n(u)}, \quad (10)$$

$$P(c|u) = w_{u,c}, \quad (11)$$

we have

$$P(u, c) = P(c|u)P(u) = \frac{n(u)w_{u,c}}{\sum_{u'\in U} n(u)}, \quad (12)$$

$$P(u|c) = \frac{P(u, c)}{\sum_u P(u, c)} = \frac{n(u)w_{u,c}}{\sum_{u'\in U} n(u')w_{u,c}}, \quad (13)$$

if q_r and u are given, we combine (9) and (13) and result in

$$P(q_t|q_r, u) \propto g(q_t; q_r, u) = \sum_{c\in C} \frac{n(u)w_{u,c}P(q_r, c, q_t)}{\sum_{u'\in U} n(u')w_{u',c}} \quad (14)$$

Therefore, we can use $g(q_t; q_r, u)$ as a score function of $q_t \in Q_t$ for recommendation.

3.5 Model Training

In practice, the performance of the EM algorithm is highly related to two factors: the number of latent factors and initialized parameters. We propose a special clustering method to tackle these two problems together, so that we can further improve both the efficiency and effectiveness of the learning process.

Hidden Dimension Selection by clustering. We introduce this clustering method by taking clustering over q_ts as an example. Firstly, since the feature space $\{\langle q_r, c\rangle\}$ of q_t is sparse, we assume that if two target query q_{t_1} and q_{t_2} co-occurred with the same $\langle q_r, c\rangle$, they tend to have the same target intent. We extract all the q_t's from RCT triples as an initial cluster for each $\langle q_r, c\rangle$. The weight of each q_t for an initial cluster is set as the frequency $n(q_r, c, q_t)$ of the triple $\langle q_r, c, q_t\rangle$ in the query log.

Secondly, since many $\langle q_r, c\rangle$ pairs may have similar latent search intent, we utilize traditional clustering methods (e.g. hierarchical clustering) to merge the initial clusters based some similarity measure (e.g. cosine similarity). Instead of specifying the clusters number, a minimal similarity threshold min_s is used to control the merging steps. Once two clusters are merged, the weights of each q_t are accumulated.

We also clustering q_r's in the same way, by considering $\langle c, q_t\rangle$ as a single target intent as an initial cluster.

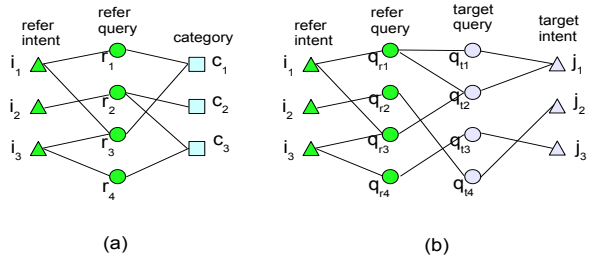

Figure 3: Example of (a) a $\langle i, q_r, c\rangle$ tripartite graph, (b) a $\langle i, q_r, q_t, j\rangle$ multipartite graph

Parameters Initialization. By utilizing the clustering results to initialize the parameters, we can significantly reduce the number of parameters.

$P(q_r|i)$ and $P(q_t|j)$ can be initialized directly from clustering results by normalizing the weights of queries in each cluster. However, $P(c|i)$ and $P(i, j)$ are not so easy to initialized from the clustering result. However, given all $P(q_r|i)$'s, $P(q_t|j)$'s and $n(q_r, c, q_t)$'s, we can grasp some instructive relation between c and i, and between intent i and intent j by putting them in multipartite graphs showing in Figure 3. Figure 3(a) is constructed by connecting a refer intent node i_k to a refer query node q_{r_k} if $P(q_{r_k}|i_k) > 0$, and connecting q_{r_k} to a category node c_k if $n(q_{r_k}, c_k) > 0$ in the training RCT triples. The idea is that we only need to initialize such $P(c|i)$s that there exists a path from c to i in the tripartite graph. For other $P(c|i)$s, even if we assign them non-zero values, the EM iterations will reduce them to zero at last. The proof is not difficult and we skip it due to the space limitation. After extracting all such kind of $P(c|i)$s, we randomly initialize them and then perform normalization for each intent i. The $P(i, j)$ initialization is almost the same, with a different multipartite graph showing in Figure 3(b). In which, q_{r_k} connect to q_{t_l} if $n(q_{r_k}, q_{t_l}) > 0$, and q_{t_k} if $P(q_{t_k}|j_k) > 0$.

4. EXPERIMENTAL RESULTS

4.1 Dataset and Baselines

Our experiments are based on the "Spring 2006 Data Asset" distributed by Microsoft Research[3]. We compared our result with three baselines: Co-occurrence (Co-occ), which recommends queries occurring after (may not adjacent) test query q most often in training query sessions; N-gram, which recommends queries closely following the query sequence most often; Context-Aware and Concept-Based method (CACB) [3]. The only difference of CACB from N-gram is that CACB clusters queries to "concepts" by their clicks, and then transform query sequences into concept sequences. To generate recommendation, a concept is represented by the most frequent query in it.

We compared these methods based on a collection of ambiguous queries. Here, we take a simple approach to extract possible ambiguous queries. By exploiting the URL-category data we extracted from ODP, we find queries with clicked URLs from multiple different categories are more likely to be ambiguous. We thus randomly extract 1000 query sessions, whose last query's clicked URL belongs to different

[3]http://research.microsoft.com/users/nickcr/wscd09/

Table 1: Examples for refer intents and target intents involve the query "saturn", but with two different means. The second row is top 5 queries in the intent, and the third row is categories corresponding to refer intent

i_{1935}	i_{19645}	j_{34564}	j_{29915}
gmc vehicles	**saturn**	**saturn**	solar system
saturn	neptune	mustang	planet mercury
volvo suv	uranus	ford	earth
honda	planet mercury	nissan	planets in order
chevrolet	venus	hyundai	**saturn**
Automotive	Astronomy		

Table 2: Query recommendation examples with different context

context: saturn → nineplanets.org			
Co-occ	N-gram	CACB	High-order
solar system	pontiac	toyota	uranus
uranus diameter	solar system	mitsubish	uranus diameter
uranus	saturn cars	honda	solar system
saturn cars	scion	hyundai	earth
saturn roadster	kia	solar system	astrology
context:honda ⇒ saturn → satrun.com			
Co-occ	N-gram	CACB	High-order
solar system	pontiac	toyota	saturn sky
uranus diameter	saturn cars	pontiac	saturn roadster
uranus	toyota	scion	saturn cars
saturn cars		gm com	pontiac
saturn roadster			mazda

categories, as our test data. Recommendation is conduct on the last query of each session. Finally, the recommendation results are manually assessed by three judges to decide whether the recommendation is meaningful or not.

Precision and recall are used to evaluate the quality of the results of these methods quantitatively. Here, precision is defined as the average ratio of meaningful ones over the all recommendations generated by a method for a test query. While recall is defined as the average ratio of meaningful recommendations found by a method over all meaningful ones in the training data. Nevertheless, it's impossible to collect all the meaningful recommendations in such a large scale data. Instead, we use the TREC pooling method to construct a pool of meaningful recommendations collecting from the top 5 recommendations of all these methods. Recall metric of each method is then calculated base on this pool.

4.2 Experiments Results

Qualitative Evaluations. In Table 1, we show two refer intents and two target intents learned by the high-order model. It's clear that i_{1935} and j_{34564} talk about cars, which is consistent with the ODP category "Automotive", while i_{19645} and j_{29915} are about planet. Furthermore, Table 2 shows recommendations of all the methods under different contexts, we can see that the High-order model can provide more accurate results than others in these cases.

Quantitative Evaluations. Figure 4 shows the precision and recall across the top 5 positions for each method. In precision, the High-order method outperforms N-gram and CACB. All of the three methods can outperform the Co-occ method significantly. It shows that for ambiguous queries, the context-aware manner can capture users' search intent more accurately than the context-free ones. In recall, our High-order method significantly outperforms the other methods. The sequence pattern based methods performs

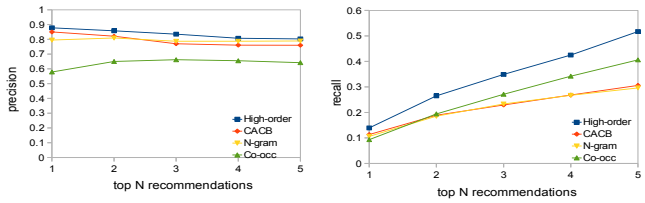

Figure 4: (a) Precision and (b) Recall of query recommendation across the top 5 positions over different methods

worst, due to the sparsity of the sequence patterns in query logs. While our approach can clearly benefit from the rich click-through information.

5. CONCLUSIONS

We address the problem of context-aware query recommendation. Unlike the existing approaches which leverage query sequence patterns in query sessions, we use the click-through of the given query as the major clue of users' search intents to provide context-aware recommendation. We proposed a probabilistic model by learning the High-order relations between the current query, its click-though and the next query. Compared with existing methods, our approach achieves both better precision and recall on recommendations for ambiguous queries.

6. ACKNOWLEDGMENTS

This research work was funded by the National High-tech R&D Program of China under grant No. 2010AA012500, and the National Natural Science Foundation of China under Grant No. 61003166 and Grant No. 60933005.

7. REFERENCES

[1] R. Baeza-Yates, C. Hurtado, and M. Mendoza. Query recommendation using query logs in search engines. In W. Lindner, M. Mesiti, C. Türker, Y. Tzitzikas, and A. Vakali, editors, *Current Trends in Database Technology - EDBT 2004 Workshops*, volume 3268 of *Lecture Notes in Computer Science*, pages 395–397. Springer Berlin / Heidelberg, 2005.

[2] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, pages 407–416, New York, NY, USA, 2000. ACM.

[3] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click-through and session data. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 875–883, New York, NY, USA, 2008. ACM.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.

[5] B. M. Fonseca, P. B. Golgher, E. S. de Moura, and N. Ziviani. Using association rules to discover search engines related queries. In *Proceedings of the First Conference on Latin American Web Congress*, pages 66–, Washington, DC, USA, 2003. IEEE Computer Society.

[6] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 162–168, New York, NY, USA, 2001. ACM.