PQC: Personalized Query Classification

Bin Cao
The Hong Kong University of Science & Technology
Hong Kong,China
caobin@cse.ust.hk

Jian-Tao Sun
Microsoft Research Asia
49 Zhichun Road
Beijing, China
jtsun@microsoft.com

Evan Wei Xiang
The Hong Kong University of Science & Technology
Hong Kong,China
wxiang@cse.ust.hk

Derek Hao Hu
The Hong Kong University of Science & Technology
Hong Kong,China
derekhh@cse.ust.hk

Qiang Yang
The Hong Kong University of Science & Technology
Hong Kong,China
qyang@cse.ust.hk

Zheng Chen
Microsoft Research Asia
49 Zhichun Road
Beijing, China
zhengc@microsoft.com

ABSTRACT

Query classification (QC) is a task that aims to classify Web queries into topical categories. Since queries are usually short in length and ambiguous, the same query may need to be classified to different categories according to different people's perspectives. In this paper, we propose the Personalized Query Classification (PQC) task and develop an algorithm based on user preference learning as a solution. Users' preferences that are hidden in clickthrough logs are quite helpful for search engines to improve their understandings of users' queries. We propose to connect query classification with users' preference learning from clickthrough logs for PQC. To tackle the sparseness problem in clickthrough logs, we propose a collaborative ranking model to leverage similar users' information. Experiments on a real world clickthrough log data show that our proposed PQC algorithm can gain significant improvement compared with general QC as well as natural baselines. Our method can be applied to a wide range of applications including personalized search and online advertising.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering, Search process; H.3.4 [**Systems and Software**]: Performance evaluation,performance measures

General Terms

Algorithms, Experimentation, Performance

Keywords

Personalized Query Classification, Clickthrough log, Collaborative ranking

CIKM'09, November 2–6, 2009, Hong Kong, China.

1. INTRODUCTION

With the exponentially increasing amount of information available on the Internet, Web search has become an indispensable tool for Web users to gain their desired information. Typically, Web users submit a short query consisting of a few words to search engines and expect to satisfy their information need. However, because these queries are usually short and ambiguous, how to interpret them is not trivial. This problem has become a major research issue in IR community. In this work, we refer to the problem of classifying queries to a set of topical categories as the query classification problem, or QC for short. The importance of QC is emphasized by many services provided by Web search. A direct application of QC is to provide better search result pages for users with interests of different categories. Search result pages can be grouped according to the categories predicted by a QC algorithm to give better user experiences. Online advertising services can be enhanced by QC results to promote different products more accurately.

Since Web queries are usually short and ambiguous, one same query may belong to different categories for different people. This difficulty is illustrated in Table 1, which shows the labeling results from three labelers in a query classification competition (ACM KDDCUP'05). In the table, the three human labelers, denoted as L1, L2 and L3, respectively, gave each query at most five labels in a ranked order. The average precision and F1 score values of each labeler are evaluated against the labeling results of the other two labelers. The average values among the three labelers are around 0.50, which indicates that the categories of the same queries are different for different people. From this example, we can see that it is difficult for humans and search engines to predict to which categories the query belongs for a particular user.

Due to the difficulty in solving the QC problem, search engines often treat queries as if they belong to all probable popular categories, in order to make the provided services fit different needs. For example, the search results are often reorganized to make returned Web pages diverse in topics (as shown in Figure 1). In online advertising, for example, when a user submits a query "apple", the advertisements related to different interpretations of the term are returned, such as "iPod", "Macintosh" and "apple (fruit)". In many search services, no matter whether the user is a farmer or a

programmer, there is no difference in the types of advertisements provided. This will cause many of the search results to be irrelevant and the delivered advertisements not to be targeted.

In order to optimize the quality of search services, it would be desirable to provide *personalized* query classification services according to the interests of each individual user. An effective way to improve personalized query understanding is to ask a user to explicitly specify which documents are relevant to the user's information need. However, previous studies have shown that the vast majority of users are reluctant to provide explicit feedback which can further reveal their preferences [5]. Therefore we need to rely on the *implicit feedback* information such as clickthrough data. Previous studies [20, 16, 15, 22] have shown that it is possible to learn users' interests over Web pages from clickthrough data for personalized search. However, it is still unclear how to achieve *personalized query classification*. One difficulty is that how to connect a user's understanding of a query to his clickthrough log. Another difficulty is that clickthrough data for a single user can be rather sparse, resulting in insufficient training data to learn a user's preference for personalized query classification. In this paper, we investigate these problems by proposing a unified model that can integrate search queries' categories and users' relevance feedbacks collaboratively. Experimental results on a real search engine's clickthrough log demonstrate that our approach is effective in tackling the personalized query classification problem.

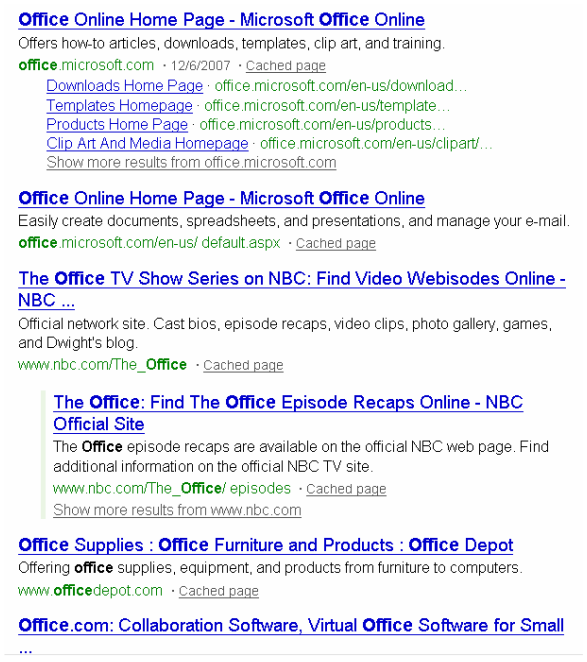

Figure 1: An example result page from a commercial search engine with query "office". The returned Web pages contain different topics ranging from Microsoft software product Office to TV show and common office products.

The rest of this paper is organized as follows. In Section 2, we will first survey the related work. In Section 3, we will introduce our overall model for PQC as well as the algorithm for learning user preferences. Then in Section 4, we will present our experimental results on real world datasets. Section 5 concludes this paper and also provides the future work we plan to investigate on this problem.

Table 1: The score of each labeler evaluated based on scores of the other two labelers as ground truth on the KDDCUP'05 dataset.

	L1	L2	L3	Avg
F1	0.538	0.477	0.512	0.509
Precision	0.501	0.613	0.463	0.526

2. RELATED WORK

2.1 Query Classification

Query classification (QC) is the task of classifying search queries into categories. It was used as the task of ACM KDDCUP in 2005 [14], an annual competition hosted by ACM KDD conference. In that year, 37 teams over the world ran their algorithms to classify 800,000 queries and submitted their results to compete. Since then, QC has received more and more attention in both academic and industry communities. Most existing QC research focused on finding new features associated with search queries to tackle the sparseness problem, or utilizing various Web resources to obtain additional training data. In [17], the winning team of ACM KDDCUP'05 proposed a novel method to classify queries to target categories without labeled data. With their method, a mapping between an intermediate taxonomy (e.g., Open Directory Project) and the target taxonomy is first constructed. Then the training data from the intermediate taxonomy are extracted to train a classification model for the target taxonomy. In [18], the authors improved their previous method by building a bridging classifier on an intermediate taxonomy and then use it to map user queries to the target categories via the above intermediate taxonomy. In [2], researchers proposed to find selection preferences between queries and query categories in the query log data to improve query classification. They also trained classifiers for QC with different features and then used these classifiers to vote for final results. To our best knowledge, no previous work has ever addressed the issue of personalized query classification (PQC). Recently, Cao et al. in [4] considered the QC problem enhanced using context information. They use linear chain conditional random field (CRF) model with context-aware features to utilize session information. However, their method only works when there is sufficient session information. They also ignored the long term history for individuals.

2.2 Personalized Search

Personalized search has become an active research topic recently due to the fact that different people often search for different things with the same query. Sugiyama et al. proposed a user profiling method for personalized search [21]. They constructed user profiles with a collaborative filtering model based on a user's browsing history in one day. Lee et al. proposed two types of features for personalized search: features constructed from user-click behavior and from the anchor-link distribution, respectively [13]. Dou et al. proposed an evaluation method for personalized search using

large scale clickthrough logs, based on which they found that both long-term and short-term contexts are very important in improving search performance for profile-based personalized search algorithms [9]. Tan et al. considered long-term search history for personalized search where a language model was used [23]. Recently, Chirita et al. proposed a personalized query enrichment method by accessing the desktop data of users [6]. In this paper, we address a different aspect of the personalization problem of query understanding, which is more general since many other personalized services, such as query suggestion or advertising, can be constructed based on PQC.

2.3 Implicit Feedback

Many efforts have been made to utilize the implicit feedback information of clickthrough logs to improve search performance. Shen et al. in [19] used previous queries and clickthrough information to reduce the ambiguity in queries to find the most relevant documents. Craswell and Szummer tried to solve the sparseness problem of clickthrough logs by using a random-walk model on a bipartite graph constructed by clicked <query, url> pairs [7]. Similar click-based models are also used in [25]. However, such click-based models may be noisy and biased (position bias) [12], causing suboptimal performance. To reduce the noise and bias in clickthrough log, there are some approaches proposed to leverage extra information such as page dwell time, users' behaviors after clicks, etc [1]. An alternative way to avoid position bias is to use the preference information contained in user clicks [11]. We notice that, such preference information can be naturally represented through a ranking-based model. For this reason, we propose a collaborative ranking model for user preference learning from clickthrough logs.

3. PQC: PERSONALIZED QUERY CLASSIFICATION

In this section, we first introduce the overall framework of PQC. Then we present our collaborative ranking model for learning user preferences. Finally, we discuss how to estimate users' short-term and long-term preferences, as well as the method of combining them to improve PQC.

3.1 Overall Model

The overall model can be illustrated by a graphical model, as shown in Figure 2. Assume that a user u wants to search for the information that belongs to a category c. A query q belonging to c is first generated and submitted to a search engine. The search engine may interpret q as being from a set of categories where the returned search result pages may fall into these candidate categories. If we view the query from user u's perspective, this query should belong to category c and the clicked pages should be generated from c as well. Take the query "apple" as an example, the returned pages may be from different categories such as "computer hardware", "OS", "MP3 player" or "fruit". If the clicked pages are about "iPod" in particular, then it indicates that the user may be searching for MP3 players.

Formally, QC aims to estimate $p(c|q)$, namely the conditional probability of searching information about the category c given the query q. PQC targets at the problem of estimating $p(c|q,u)$, where the user u is also considered. $p(c|q,u)$ is the probability that the query q belongs to the category c for the user u. Using the Bayes formula,

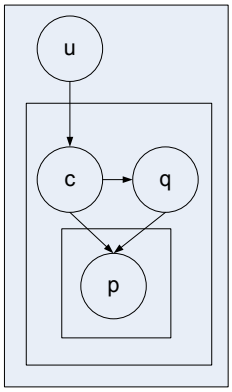

Figure 2: A graphical model representation for PQC.

$$\begin{aligned} p(c|q,u) &= \frac{p(c,q,u)}{p(q,u)} \\ &\propto p(c)p(q,u|c) \\ &= p(c)p(q|c)p(u|q,c) \end{aligned} \tag{1}$$

The first factor is the prior probability of the category c. The second factor is the probability of generating the query q from the category c. The third factor is the probability of occurrence of the user u conditioned on the category c and the query q. We can simplify this term by assuming that

$$p(u|q,c) = p(u|c).$$

This assumption means that user u has stable interests that are independent of the current query q. Therefore the condition on q is unnecessary. This assumption may hold in both long term and short-term perspectives. For example, from long term perspective, a computer scientist may have stable interests in computer science related information. From short-term perspective, any user aimed at searching certain information may have stable interests in the topics related to that information within the search session. Although there are counterexamples which violate this assumption, hereby we assume such an assumption holds most of the time.

Then the simplified equation becomes

$$p(c|q,u) \propto p(q|c)p(u|c)p(c) \tag{2}$$

This quantity is proportional to the probability that q is generated from c and the probability that u has interest in c.

Estimating $p(q|c)$ can be obtained from general QC. We use the method from [17], which has been shown to achieve promising classification accuracy. Then by Bayes rules,

$$p(q|c) = \frac{p(q,c)}{p(c)} = \frac{p(c|q)p(q)}{p(c)} \propto p(c|q)p(q) \tag{3}$$

To estimate $p(u|c)$, we also apply the Bayes rules

$$p(u|c) = \frac{p(u,c)}{p(c)} = \frac{p(c|u)p(u)}{p(c)} \propto p(c|u)p(u) \tag{4}$$

by which we convert the problem into estimating the user's preference on categories $p(c|u)$. Since the majority of users may be reluctant to provide explicit feedback information

on their search preferences [5]. Therefore, it is desirable if the user preferences can be learned *automatically* from the historical clickthrough logs. We will address this problem in the following sections.

3.2 User Preferences for PQC

An intuitive idea is that the historical queries submitted by a user can be used to help learn user preferences:

$$p(u|c) \approx p(q_1, q_2, \cdots, q_t|c)$$

The above equation shows that we can treat the problem of estimating user preferences as a query classification problem. The difference from the previous QC work is that we need to classify a group of queries collectively rather than an individual query.

The approach above can be used to learn the short-term user preferences within a short time period, e.g., within a search session. Suppose that a user wants to find some resources on "machine learning" and submits a query "machine learning tools". The next query may be "weka". Obviously, user preferences with only a query "weka" is not clear since the query itself is ambiguous. We can utilize the previous queries to help make better prediction.

However, this method has a major problem for preference learning. The sessions containing the current query may not reflect all search interests of a given user. Also, the method does not utilize the user's click history information. Additionally, the sessions of one user may be too limited to infer user preferences. These reasons provide the motivation for a collaborative ranking model, which we present next, to solve the PQC problem.

3.3 Collaborative Ranking Model for Learning User Preferences

Clickthrough data is often quite sparse, meaning that the information for inferring search interests of a particular user may be very limited [9]. Our key observation is that users who clicked the same pages for the same query tend to have similar interpretations for the query. Therefore, a personalized query classification algorithm can use the clickthrough data of many similar users to improve the query classification performance.

Therefore, to estimate $p(c|u)$, we introduce a collaborative ranking model to solve the sparseness problem. Consider the above example of the query "Apple". Even if a user who submitted this query did not search for results related to "mp3 players" before, his previous search profile may indicate that he is similar to some other users who have queried "mp3 players". In other words, we can exploit the correlations between users and their interested categories. Next, we will formalize this intuition using a latent factor model.

A major difficulty faced by click-based models as indicated by [12], is that user clicks are biased by the displayed position. Therefore, pairwise preferences are suggested instead of click information, which can avoid the problem of position bias. Besides, user preference can be naturally expressed as a ranking relation. For example, we may say that we prefer finding information on "computers" more than on "fruits", but we may find it difficult to attach a numerical score to indicate how much we prefer an individual category. Therefore, in this section, we propose a collaborative ranking model for user preferences learning.

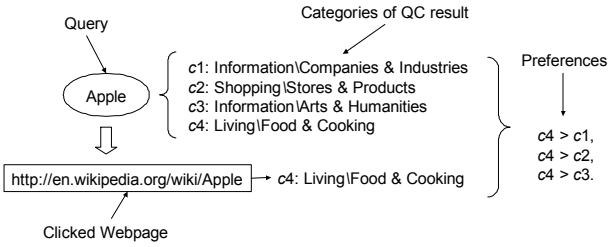

Figure 3: An example on preferences extraction.

3.3.1 Generating Pairwise Preferences

Since search queries are usually too short to specify the exact search intent, a general query classifier may classify a query q to several categories $C_q = \{c_1, c_2, \cdots, c_k\}$. However, for a particular user u, the query q may only refer to fewer categories. The preferences of the user u may be implicitly reflected by u's clickthrough logs. If the user u only clicked on Web pages of category c_i, we can generate the preference $c_i \succ c_j, j \neq i$ for user u, which states that c_i is more preferred than c_j.

More generally, for each clicked page, we can obtain its category information $C_p = \{c'_1, c'_2, \cdots, c'_k\}$. Then we can generate the preferences by the following formula

$$\{c_i \succ c_j | c_i \in C_q \cap C_p, c_j \in C_q - C_p\}$$

where $\cap$ and $-$ are the set operations of intersection and complement, respectively. An example can be found in Figure 3. In the given example, the query is "apple" and the related categories returned by QC including "Information \Companies" and "Living \Food", etc. Since the user only clicked the Web page about the fruit, we can generate the pairwise preference such as "Living \Food $\succ$ Information \Companies".

3.3.2 Learning User Preferences

To model the pairwise preferences, we adopt the likelihood cost function used in [3]. Let r_{ui} be the preference score for the user u and the category i. For the observed pairwise preference such that the category i is preferred to the category j, we use the Bradley-Terry model [3] to define its likelihood.

$$P(r_{ui} \succ r_{uj}) = \frac{1}{1 + e^{-(r_{ui} - r_{uj})}} = \varphi(r_{ui} - r_{uj})$$

where $\varphi(x) = \frac{1}{1+e^{-x}}$. Notation $r_{ui} \succ r_{uj}$ means that the category i is preferred to the category j for the user u. For each user u, we may have a set of pairwise preferences $< i, j >_u$.

Latent factor models are able to uncover the underlying factors that determine the generation of observed data. For example, PLSA [10] models how documents are generated from latent semantic topics. Considering the problem in a probabilistic framework, we model r_{ui} as the joint probability $p(u, c)$, which represents the probability of the event that a user u is interested in a category c. The probability will be high if they are both related to some latent factor f, which can be formulated as $p(u, c) = \sum_f p(u|f)p(c|f)p(f)$. These latent factors may represent topics or other forms of semantic clusters. We can further use matrix notations to

simplify the problem. Due to the equivalence between PLSA and nonnegative matrix factorization [8], the model can be expressed as

$$\mathbf{R} = \mathbf{U}\mathbf{C}^{\mathrm{T}},$$

where $\mathbf{R}$ is the matrix of relevance score with rows representing users and columns representing queries. The probabilistic perspective helps us to understand the interpretation of the model. However, it would be more convenient to use a matrix factorization based model which is easier to handle from computation perspective.

Unifying the Bradley-Terry model and the matrix factorization model, we can obtain the log-likelihood function as

$$l = \sum_{u} \sum_{<i,j>_u} \log \varphi(\mathbf{u}_u \mathbf{C}^{\mathrm{T}} \mathbf{d}_{i \succ j}^{\mathrm{T}})$$

where $\mathbf{d}_{i \succ j}$ is a vector with element i being 1 and j being -1, otherwise being 0. $\mathbf{u}_u$ is the uth row of the matrix $\mathbf{U}$.

For the matrices $\mathbf{U}$ and $\mathbf{C}$, we also assign them prior distributions with the following form

$$p(c_{ij}) \sim \mathcal{N}(0, \sigma_c^2), \quad p(u_{ij}) \sim \mathcal{N}(0, \sigma_u^2) \tag{5}$$

where u_{ij} and c_{ij} are the elements of $\mathbf{U}$ and $\mathbf{C}$. σ_u and σ_c are parameters controlling the confidence of prior. The prior distributions have the same effect as regularization terms in maximum margin learning [24], which can prevent overfitting. Then, we obtain the posterior distribution as our objective function,

$$l = \sum_{u} \sum_{<i,j>_u} \log \varphi(\mathbf{u}_u \mathbf{C}^{\mathrm{T}} \mathbf{d}_{i \succ j}^{\mathrm{T}}) + \log(p(\mathbf{U})) + \log(p(\mathbf{C}))$$

Gradient based optimization algorithms can be used to optimize the above objective function. The gradients of the cost function with respect to $\mathbf{U}$ and $\mathbf{C}$ are shown in Equation (6) and (7).

$$\frac{\partial l}{\partial \mathbf{u}_u} = \sum_{<i,j>_u} \frac{\varphi'(\mathbf{u}_u \mathbf{C}^{\mathrm{T}} \mathbf{d}_{i \succ j}^{\mathrm{T}})}{\varphi(\mathbf{u}_u \mathbf{C}^{\mathrm{T}} \mathbf{d}_{i \succ j}^{\mathrm{T}})} \mathbf{d}_{i \succ j} \mathbf{C} - \mathbf{u}_u / \sigma_q^2 \tag{6}$$

$$\frac{\partial l}{\partial \mathbf{C}} = \sum_{i} \sum_{<i,j>_u} \frac{\varphi'(\mathbf{u}_u \mathbf{C}^{\mathrm{T}} \mathbf{d}_{i \succ j}^{\mathrm{T}})}{\varphi(\mathbf{u}_u \mathbf{C}^{\mathrm{T}} \mathbf{d}_{i \succ j}^{\mathrm{T}})} \mathbf{d}_{i \succ j}^{\mathrm{T}} \mathbf{u}_u - \mathbf{C} / \sigma_u^2 \tag{7}$$

where $\varphi'(x) = \frac{e^{-x}}{(1+e^{-x})^2}$.

When $-x$ is large, e^{-x} may overflow. In order to avoid this issue, the following equivalent equation is used.

$$\varphi'(x) = \frac{1}{(1+e^{x})(1+e^{-x})}$$

When $|x|$ is large, $\varphi'(x)$ is equal to 0.

There are different gradient-based algorithms available for such a non-constraint optimization problem. In this paper, we use the gradient-descent algorithm to optimize the objective function. For each iteration, the complexity of calculating the gradient is in the order of N, where N is the number of preference pairs in the training data. It is independent of the size of the matrix. Therefore, our algorithm can handle large-scale data with millions of users. After obtaining the preference score matrix, we can normalize the matrix R with respect to each user to get $p(c|u)$.

3.4 Combining Long-Term Preferences and Short-Term Preferences

Users typically have long-term preferences as well as short-term preferences. For example, a computer scientist usually submits many queries related to computer science (long-term preference) while he may also submit some queries not related to computer science sometimes (short-term preference). We show how to convert preferences learning as a query classification problem in Section 3.2. We can use the method with queries in a session to learn the short-term preference. In Section 3.3 we proposed a collaborative ranking model, which can be used for long-term preferences learning. In this section, we will discuss how to combine these two kinds of preferences to do personalized query classification. A simple, yet effective, method is to directly add the two preference models together as

$$p(q|c) = (1-\alpha) p_s(q|c) + \alpha p_l(q|c) \tag{8}$$

where α is an importance weight for short-term preferences. A user session, for example, can be regarded as a kind of short-term preferences. One problem of the session based methods is that they are unable to deal with *very short sessions* well, since very short sessions do not provide much contextual information. When there are very limited data available, it is hard to infer the real interest of users. This problem can be solved by using the long-term preferences, which is verified in our experimental section.

With the learned user preferences, we can classify the queries into categories from a user's perspective. The classification process can be achieved by using a Bayes rule given in Equation 2.

4. EXPERIMENTS

In this section, we will demonstrate the effectiveness of our proposed PQC method through a series of experiments. We first introduce our experimental setup, including some basic descriptions of the dataset and the evaluation metric we will use in our experimental settings. Next we will show the result of our proposed collaborative ranking model on the dataset to demonstrate the effectiveness in learning the user preferences. Finally, we evaluate the learned user preferences through improvement of query classification performance.

4.1 Experimental Setup

4.1.1 Experimental Data

In order to validate the effectiveness of our proposed solution to PQC, we have carried out extensive experiments on a real world clickthrough log dataset obtained from a commercial search engine. This clickthrough log dataset is recorded from November 1, 2007 to November 15, 2007. We randomly sampled 10,000 users for our experiments. There are 22,696 queries and 51,366 urls involved in the dataset. Figure 4 shows the histogram of the number of queries submitted by the users in the log. We found that in the log, there are only 22 users whose historical query lengths are longer than 50. These users are not shown in the figure for better display of the results. In this dataset, most users have fewer than 10 historical queries, which supports our claim that short sessions are the majority from which little contextual information can be obtained. The average number of

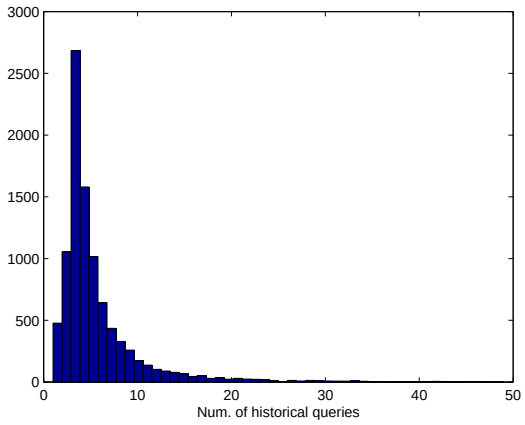

Figure 4: Histogram of number of historical queries for users.

historical queries amongst all users is 5.3. This average number of queries is not large since the time-span of the original clickthrough log is not long; users would not issue too many queries during such a short time period. We plan to investigate the experiment on long time-span data in our future work. Although the time-span is not quite long, we will see our proposed method still have significant performance gain compared to the baselines.

4.1.2 Evaluation

Performing evaluation for large-scale PQC results is a non-trivial task, since it is impossible to ask users to label their own queries during their searching behaviors. Therefore, we resort to implicit feedbacks based on the clickthrough logs, to reflect important information about users' real information needs. In this section, we propose an unsupervised evaluation method based on clickthrough logs. The evaluation can be done in an automatic way without manual labeling.

Our basic idea is as follows. We observe that the categories of user-clicked pages reflect the user's real intention after the user submits a query. Thus, PQC results should be consistent with the category label of the clicked pages. As a result, we can check the consistency of the prediction from PQC and user's clicking behavior. For example, if the PQC algorithm classifies a query q for the user u into a category c, but the user's clicks show what he/she wanted is a Web page that is classified into a category c' different from c, then the PQC system must have predicted wrongly. This rule can be generalized for consistency measurement, by defining a new metric $hit@k$, as follows:

$$hit@k = \frac{|C_q^k \cap C_p|}{|C_q|} \tag{9}$$

where C_q^k is the top k categories predicted by the algorithm for the query q, while C_q, without superscripts, is all the categories the PQC algorithm predicts for the query q. C_p is the categories obtained from the user's clicked pages p, which is regarded as the ground truth results of PQC. The value of $hit@k$ will be high if the PQC results are more consistent with the categories of the clickthrough logs.

For the classification labels, we use the categories defined in the ACM KDDCUP'05 data set[1]. There are a total of 67 categories that form a hierarchy. Table 2 shows several examples of queries and their categories in the test data of ACM KDDCUP'05 competition. As we can see, for each query, it may belong to more than one category. Thus, as we mentioned before, it is important to disambiguate between these candidate categories and select the "true labels" amongst them.

Table 2: Example Queries and Their Categories

Queries	Categories
FIFA 2006	Sports/Soccer Sports/Schedules & Tickets Entertainment/Games & Toys
cheesecake recipes	Living/Food & Cooking Information/Arts & Humanities
friendships poem	Information/Arts & Humanities Living/Dating & Relationships
apple	Information/Local & Regional Information/Companies & Industries Shopping/Stores & Products Living/Food & Cooking Information/Arts & Humanities
office	Information/Arts & Humanities Entertainment/Humor & Fun Entertainment/TV Shopping/Buying & Researching Computers/Software

4.1.3 Obtain PQC Labels

In order to obtain the categories of clicked pages, we need to build a classifier to classify Web pages into the same label space as the queries. We crawled the contents of Web pages that have been clicked. Since we do not have labeled data to train the classifier, we use *bridging classifier* introduced in [18] to construct the classification model.

We import the supervision information from a third party knowledge base to leverage the connection between a clicked page p and the target label C^T in the following manner:

$$P(C^T|p) \propto \sum_{C_j^I \in C^I} P(C^T|C_j^I)P(p|C_j^I)P(C_j^I) \tag{10}$$

where C_j^I is some intermediate category in the third party knowledge base (in our case, Wikipedia is used as the knowledge base). We adopt a uniform distribution for the class prior $P(C_j^I)$. The likelihood $P(C^T|C_j^I)$ and $P(p|C_j^I)$ can be estimated with the statistics $P(w_i|C_j^I)$ provided by the knowledge base

$$P(d|C_j^I) = \prod_{w_i \in d} P(w_i|C_j^I). \tag{11}$$

We selected the top 100 relevant articles from Wikipedia to form C^I. Since the articles in Wikipedia are organized in a network structure, we added the neighboring articles to the corresponding C_j^I. We use mutual information as the measure for relevance.

We use the KDDCUP'05 data set to test our bridging classifier implemented with Wikipedia. The comparison is

[1] http://www.sigkdd.org/kdd2005/kddcup.html

Table 3: Performance on ACM KDDCUP'05 Test Bed

	Precision@5	Recall@5	F1@5
Wikipedia	**32.9%**	**48.9%**	**39.0%**
KDDCUP	32.1%	48.8%	38.3%

shown in Table 3, and the baseline performance is the winning solution of KDDCUP'05 competition [17]. From the table, we can observe that our implementation achieved comparable results.

We split the data into training and test subsets according to the following method. First, we set a threshold value for the length of historical queries. The records for each user within this length are treated as the training data and future queries are treated to be test data for evaluation. For example, suppose that we set the threshold value to 5. All of the first 5 queries that are submitted by a user are used as the training data, while the remaining queries are used as the test data. For users who submitted no more than five queries, we simply used all their queries for training and do not include their queries in the testing phase. The QC algorithm in [17] will assign a query to each category with a probability, which is $p(c|q)$. We use the top five categories as the query category list. Since the QC algorithm does not consider the user information, therefore the ranking of the five categories are the same for all users.

For our PQC method, we learn user preferences and calculate $p(c|u,q)$ to re-rank the top five categories. For both the QC and PQC results, we can evaluate them by the $hit@k$ metric introduced above. It is easy to see that, the higher the value of $hit@k$ is, the better the performance of the algorithm will be.

4.2 Effects of Personalization on QC

Table 4 summarizes our experimental results of PQC and compares it with the basic QC approach, using the $hit@k$ metric introduced above. From the table, we can observe that the $hit@1$ value is improved significantly by applying the PQC approach. Recall that the $hit@1$ value measures whether the top predicted category reflects the user's clicking behavior. For $hit@2$ and $hit@3$ the improvements are also significant. It can also be observed that with the increase of k, the amount of improvement we can achieve drops. Such a phenomenon is reasonable, since the more categories we predict, the more possible it would be to include the category of the clicked URL into the predicted category set of the queries. However, since in practical applications, the top ranked category is the most important output, such experimental result indicates personalization is a very promising way to achieve better query classification performance. We also can compare our proposed PQC model with a memory-based approach that does not use collaboration. The method is one special case when we choose $\alpha = 1$ in Equation 8.

4.2.1 *Effect of History Length*

Figure 5 shows the change of $hit@k$ when the number of historical queries used for training varies. To conduct this experiment, we use the records that are after the tenth queries of users' as test data. Then we fix the test data and change the number of previous queries used as training data.

Table 4: Comparison between QC and PQC. The QC algorithm that achieves comparable results as the solution of ACM KDDCUP'05 championship team. The method Mem is a memory-based approach without collaboration.

	QC	Mem	PQC	Improvement (to QC)
hit@1	0.08	0.10	**0.13**	62.5%
hit@2	0.21	0.24	**0.26**	28.8%
hit@3	0.33	0.35	**0.37**	12.1%
hit@4	0.44	0.44	**0.46**	4.6%

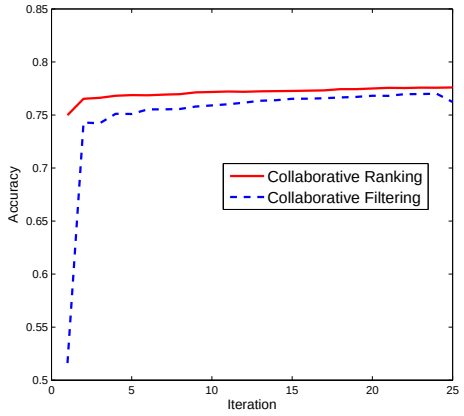

Figure 7: Compare collaborative ranking model with collaborative filtering.

As shown in Figure 5, it is reasonable that with more users' historical queries in our system, our prediction becomes more accurate. This is a clear trend for $hit@1$. For $hit@k$ when $k \neq 1$, there seems to be no improvement from a length of six to ten. However, the variance is greatly reduced. It is also surprising to see that even with only one previous query as training data, we can still significantly improve the accuracy of PQC results as compared to the QC result in Table 4. This indicates the effectiveness of using collaboration ranking for PQC.

4.2.2 *Effect of Long-Term and Short-Term*

Figure 6 shows the change of $hit@k$ when α varies. The trend reflects the effect of long-term preferences and short-term preferences. The figure shows the best performance is achieved when both long-term and short-term preferences are considered. For $hit@1$ and $hit@2$, we can find that long-term preferences are more useful than short-term preferences. This observation validates our previous assumption that clickthrough log in one session is not sufficient to infer users' preferences.

4.3 On User Preferences Prediction

Understanding the underlying user preferences is a key part for our PQC algorithm. We use an example to show that our PQC algorithm could make more reasonable predictions on query categories while taking the user preferences into consideration. Table 5 shows two users who are interested in two different categories of the query "video". The table shows the results produced by both the QC and our

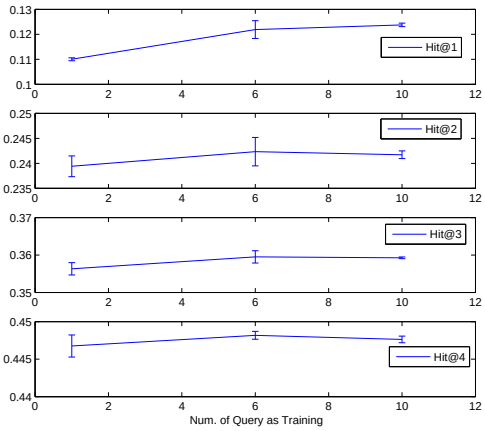

Figure 5: Performance with different length of training queries.

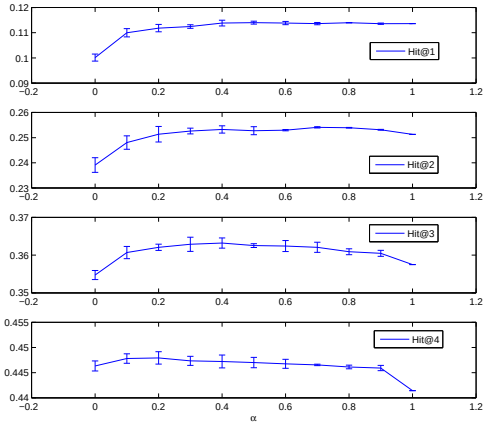

Figure 6: Performance with different α.

PQC approaches. It can be seen that, for the PQC outputs, categories involving "computers" categories are ranked higher for user A and categories involving "Entertainment" categories are ranked higher for user B. To evaluate the effectiveness of our model more precisely, we also calculated the accuracy of user preference prediction, which can be calculated on the held-out test data. Collaborative ranking is learned according to our algorithm in the previous section. Figure 7 shows the comparison between collaborative filtering method and our proposed collaborative ranking method on the accuracy of prediction. It can be seen that our collaborative ranking method could outperform the collaborative filtering method, especially when the iteration number is small.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we developed a personalized query classification model PQC which can significantly improve the accuracy of query classification. The PQC solution uses a collaborative ranking model for users' preference learning to leverage many similar users' preferences. Preference learning is based on the relevance feedback and collaborative information. We also proposed an evaluation method for PQC using clickthrough logs, based on which, we have evaluated our PQC system with ranking results from human subjects and clickthrough logs.

In the future, we plan to investigate other types of personalized query classification problems that are different from topical queries, such as functional output, like categorizing queries into commercial/non-commercial ones. Since users' interests may change with time, we can also take time into consideration. For example, for a user, more recent preferences may be more important. Besides, we also plan to apply the PQC solution to other personalized services such as personalized search and advertising.

6. ACKNOWLEDGMENTS

Bin Cao, Derek Hao Hu and Qiang Yang are supported by a grant from MSRA (MRA07/08.EG01).

7. REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual ACM SIGIR conference*, pages 19–26, Seattle, Washington, USA, 2006. ACM.

[2] S. M. Beitzel, E. C. Jensen, D. D. Lewis, A. Chowdhury, and O. Frieder. Automatic classification of web queries using very large unlabeled query logs. *ACM Trans. on Info. Syst. (TOIS)*, 25, Apr. 2007.

[3] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd ICML*, pages 89–96, Bonn, Germany, 2005. ACM.

[4] H. Cao, D. H. Hu, D. Shen, D. Jiang, J.-T. Sun, E. Chen, and Q. Yang. Context-aware query classification. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 3–10, New York, NY, USA, 2009. ACM.

[5] J. M. Carroll and M. B. Rosson. The paradox of the active user. In J. M. Carroll, editor, *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, chapter 5, pages 80–111. Bradford Books/MIT Press, 1987.

[6] P. A. Chirita, C. S. Firan, and W. Nejdl. Personalized query expansion for the web. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 7–14, Amsterdam, The Netherlands, 2007. ACM.

[7] N. Craswell and M. Szummer. Random walks on the click graph. In *Proceedings of the 30th annual ACM SIGIR conference*, pages 239–246, Amsterdam, The Netherlands, 2007. ACM.

[8] C. Ding, T. Li, and W. Peng. NMF and PLSI: equivalence and a hybrid algorithm. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 641–642, Seattle, Washington, USA, 2006. ACM.

[9] Z. Dou, R. Song, and J. Wen. A large-scale evaluation

Table 5: A PQC example: a user submits query "video". The clickthrough logs show this user is more interested in the entertainment

Query	QC	PQC(User A)	PQC(User B)
video	1: Information\Arts & Humanities	1: Computers\Multimedia	1: Entertainment\Movies
	2: Computers\Multimedia	2: Entertainment\Games & Toys	2: Information\Arts& Humanities
	3: Entertainment\Celebrities	3: Entertainment\Movies	3: Computers\Multimedia
	4: Entertainment\Games & Toys	4: Entertainment\Celebrities	4: Entertainment\Celebrities
	5: Entertainment\Movies	5: Information\Arts & Humanities	5: Entertainment\Games & Toys

and analysis of personalized search strategies. In *Proceedings of the 16th international conference on World Wide Web*, pages 581–590, Banff, Alberta, Canada, 2007. ACM.

[10] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.

[11] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual ACM SIGIR conference*, pages 154–161, Salvador, Brazil, 2005. ACM.

[12] T. Joachims, H. Li, T.-Y. Liu, and C. Zhai. Learning to rank for information retrieval (lr4ir 2007). *SIGIR Forum*, 41(2):58–62, 2007.

[13] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in web search. In *Proceedings of the 14th international conference on World Wide Web*, pages 391–400, Chiba, Japan, 2005. ACM.

[14] Y. Li, Z. Zheng, and H. K. Dai. Kdd cup-2005 report: facing a great challenge. *SIGKDD Explor. Newsl.*, 7(2):91–99, 2005.

[15] Z. Ma, G. Pant, and O. R. L. Sheng. Interest-based personalized search. *ACM Trans. Inf. Syst.*, 25(1):5, 2007.

[16] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 727–736, New York, NY, USA, 2006. ACM.

[17] D. Shen, R. Pan, J. Sun, J. Pan, K. Wu, J. Yin, and Q. Yang. Q2C@UST: Our winning solution to query classification in KDDCUP 2005. In *SIGKDD Explorations*, volume 7, pages 100–110. ACM, Dec. 2005.

[18] D. Shen, J. Sun, Q. Yang, and Z. Chen. Building bridges for web query classification. In *SIGIR'06*, pages 131–138, New York, NY, USA, 2006. ACM Press.

[19] X. Shen, B. Tan, and C. Zhai. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th annual ACM SIGIR conference*, pages 43–50, Salvador, Brazil, 2005. ACM.

[20] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 824–831, New York, NY, USA, 2005. ACM.

[21] K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *Proceedings of the 13th international conference on World Wide Web*, pages 675–684, New York, NY, USA, 2004. ACM.

[22] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 382–390, New York, NY, USA, 2005. ACM.

[23] B. Tan, X. Shen, and C. Zhai. Mining long-term search history to improve search accuracy. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 718–723, Philadelphia, PA, USA, 2006. ACM.

[24] M. Weimer, A. Karatzoglou, Q. Le, and A. Smola. Cofi rank - maximum margin matrix factorization for collaborative ranking. In *Advances in Neural Information Processing Systems 20*, pages 1600, 1593. MIT Press, 2007.

[25] G.-R. Xue, H.-J. Zeng, Z. Chen, Y. Yu, W.-Y. Ma, W. Xi, and W. Fan. Optimizing web search using web click-through data. In *CIKM '04*, pages 118–126, New York, NY, USA, 2004. ACM.