Building Bridges for Web Query Classification

Dou Shen[†], Jian-Tao Sun[‡], Qiang Yang[†], Zheng Chen[‡]

[†]Department of Computer Science and Engineering,
Hong Kong University of Science and Technology
{dshen,qyang}@cs.ust.hk

[‡]Microsoft Research Asia, Beijing, P.R.China
{jtsun, zhengc}@microsoft.com

ABSTRACT

Web query classification (QC) aims to classify Web users' queries, which are often short and ambiguous, into a set of target categories. QC has many applications including page ranking in Web search, targeted advertisement in response to queries, and personalization. In this paper, we present a novel approach for QC that outperforms the winning solution of the ACM KDDCUP 2005 competition, whose objective is to classify 800,000 real user queries. In our approach, we first build a bridging classifier on an intermediate taxonomy in an offline mode. This classifier is then used in an online mode to map user queries to the target categories via the above intermediate taxonomy. A major innovation is that by leveraging the similarity distribution over the intermediate taxonomy, we do not need to retrain a new classifier for each new set of target categories, and therefore the bridging classifier needs to be trained only once. In addition, we introduce category selection as a new method for narrowing down the scope of the intermediate taxonomy based on which we classify the queries. Category selection can improve both efficiency and effectiveness of the online classification. By combining our algorithm with the winning solution of KDDCUP 2005, we made an improvement by 9.7% and 3.8% in terms of precision and F1 respectively compared with the best results of KDDCUP 2005.

Categories and Subject Descriptors

H.3.m [**Information Storage and Retrieval**]: Miscellaneous; I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*

General Terms

Algorithms, Experimentation

Keywords

Web Query Classification, Bridging Classifier, Category Selection, KDDCUP 2005

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.

1. INTRODUCTION

With exponentially increasing information becoming available on the Internet, Web search has become an indispensable tool for Web users to gain desired information. Typically, Web users submit a short Web query consisting of a few words to search engines. Because these queries are short and ambiguous, how to interpret the queries in terms of a set of target categories has become a major research issue. In this paper, we call the problem of generating a ranked list of target categories from user queries the query classification problem, or QC for short.

The importance of QC is underscored by many services provided by Web search. A direct application is to provide better search result pages for users with interests of different categories. For example, the users issuing a Web query "apple" might expect to see Web pages related to the fruit apple, or they may prefer to see products or news related to the computer company. Online advertisement services can rely on the QC results to promote different products more accurately. Search result pages can be grouped according to the categories predicted by a QC algorithm. However, the computation of QC is non-trivial, since the queries are usually short in length, ambiguous and noisy (e.g., wrong spelling). Direct matching between queries and target categories often produces no result. In addition, the target categories can often change, depending on the new Web contents as the Web evolves, and as the intended services change as well.

KDDCUP 2005 (http://www.acm.org/sigkdd/kddcup) highlighted the interests in QC, where 800,000 real Web queries are to be classified into 67 target categories. Each query can belong to more than one target category. For this task, there is no training data provided. As an example of a QC task, given the query "apple", it should be classified into "Computers\Hardware; Living\Food&Cooking".

The winning solution in the KDDCUP 2005 competition, which won on all three evaluation metrics (precision, F1 and creativity), relied on an innovative method to map queries to target categories. By this method, an input query is first mapped to an intermediate category, and then a second mapping is applied to map the query from the intermediate category to the target category. However, we note that this method suffers from two potential problems. First, the classifier for the second mapping function needs to be trained whenever the target category structure changes. Since in real applications, the target categories can change depending on the needs of the service providers, as well as the distribution of the Web contents, this solution is not flexible

enough. What would be better is to train the classifiers once and then use them in future QC tasks, even when the target categories are different. Second, the winners used the Open Directory Project (ODP) taxonomy as the intermediate taxonomy. Since the ODP contains more than 590,000 different categories, it is costly to handle all mapping functions. It is better to select a portion of the most relevant parts of the intermediate categories.

In this paper, we introduce a novel QC algorithm that solves the above two problems. In particular, we first build a bridging classifier on an intermediate taxonomy in an offline mode. This classifier is then used in online mode to map users' queries to the target categories via the above intermediate taxonomy. Therefore, we do not have to build the classifier each time the target categories change. In addition, we propose a category-selection method to select the categories in the intermediate taxonomy so that the effectiveness and efficiency of the online classification can be improved.

The KDDCUP 2005 winning solution included two kinds of base classifiers and two ensemble classifiers of them. By comparing our new method with any base classifier in the winner's solution for the KDDCUP 2005 competition, we found that our new method can improve the performance by more than 10.4% and 7.1% in terms of precision and F1 respectively, while our method does not require the extra resource such as WordNet [8]. The proposed method can even achieve a similar performance to the winner's ensemble classifiers that achieved the best performance in the KDDCUP 2005 competition. Furthermore, by combining the our method with the base classifiers in the winner's solution, we can improve the classification results by 9.7% in terms of precision and 3.8% in terms of F1 as compared to the winner's results.

This rest of the paper is organized as follows. We define the query classification problem in Section 2. Section 3 presents the methods of enriching queries and target categories. In Section 4, we briefly introduce the previous methods and put forward a new method. In Section 5, we compare the approaches empirically on the tasks of KDDCUP 2005 competition. We list some related works in Section 6. Section 7 gives the conclusion of the paper and some possible future research issues.

2. PROBLEM DEFINITION

The query classification problem is not as well-formed as other classification problems such as text classification. The difficulties include short and ambiguous queries and the lack of training data. In this section, inspired by KDDCUP 2005, we give a stringent definition of the QC problem.

Query Classification:

* *The aim of query classification is to classify a user query Q_i into a ranked list of n categories C_{i1}, C_{i2}, ..., C_{in}, among a set of N categories $\{C_1, C_2, \ldots, C_N\}$. Among the output, C_{i1} is ranked higher than C_{i2}, and C_{i2} is higher than C_{i3}, and so on.*

* *The queries are collected from real search engines submitted by Web users. The meaning and intension of the queries are subjective.*

* *The target categories are a tree with each node representing a category. The semantic meaning of each category is defined by the labels along the path from the root to the corresponding node.*

In addition, the training data must be found online because, in general, labeled training data for query classification are very difficult to obtain.

Figure 1 illustrates the target taxonomy of the KDDCUP 2005 competition. Because there are no data provided to define the content and the semantics of a category, as in conventional classification problems, a new solution needs be found. As mentioned above, an added difficulty is that the target taxonomy may change frequently. The queries in this problem are from the MSN search engine (http://search.msn.com). Several examples of the queries are shown in Table 1. Since a query usually contains very few words, the sparseness of queries becomes a serious problem as compared to other text classification problems.

Table 1: Examples of queries.

1967 shelby mustang
actress hildegarde
a & r management" property management Maryland
netconfig.exe

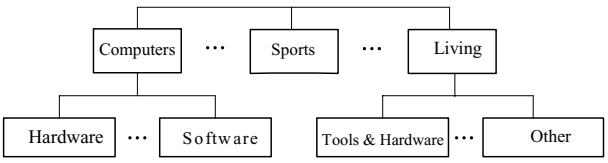

Figure 1: An Example of the Target Taxonomy.

3. QUERY AND CATEGORY ENRICHMENT

In this section, we discuss the approaches for enriching queries and categories, which are critical for the query classification task.

3.1 Enrichment through Search Engines

Since queries and categories usually contain only a few words in the QC problem, we need to expand them to obtain richer representations. One straightforward method is to submit them to search engines to get the related pages (for categories, we can take their labels as the queries and submit them to search engines, such as "Computers\Hardware" in Figure 1). The returned Web pages from search engines provide the context of the queries and the target categories, which can help determine the meanings/semantics of the queries and categories.

Given the search results for a query or category, we need to decide what features should be extracted from the pages to construct the representation. Three kinds of features are considered in this paper: the title of a page, the snippet generated by the search engines, and the full plain text of a page. The snippet is in fact a short query-based summary of a Web page in which the query words occur frequently. The full plain text is all the text in a page with the html tags removed. Since the title of a page is usually very short (5.2 words on average for our data set), we combine it with

other kinds of features together. These features are studied in our experiments.

Besides the above textual features, we can also obtain the category information of a Web page through the directory information from search engines. For example, Google's "Directory Search" can provide the labels of the returned Web pages. Such labels will be leveraged to classify a query, as stated in Section 4.1.

3.2 Word Matching Between Categories

The query classification problem can be converted to a traditional text classification problem by finding some training data online for each category in the target taxonomy. Our method of collecting the training data is by finding documents in certain intermediate taxonomies that are found online. To do so, we need to construct mapping functions between the intermediate categories and the target categories. Given a certain category in an intermediate taxonomy, we say that it is directly mapped to a target category if and only if the following condition is satisfied: one or more terms in each node along the path in the target category appear along the path corresponding to the matched intermediate category. For example, the intermediate category "Computers\Hardware \Storage" is directly mapped to the target category "Computers\Hardware" since the words "Computers" and "Hardware" both appear along the path $Computers \rightarrow Hardware \rightarrow Storage$ as shown in Figure 2. We call this matching method *direct matching*.

After constructing the above mapping functions by exact word matching, we may still miss a large number of mappings. To obtain a more complete mapping function, we expand the words in the labels of the target taxonomy through a thesaurus such as the WordNet [8]. For example, the keyword "Hardware" is extended to "Hardware & Devices & Equipments". Then an intermediate category such as "Computers\Devices" can now be mapped to "Computers\Hardware". This matching method is called *extended matching* in this paper.

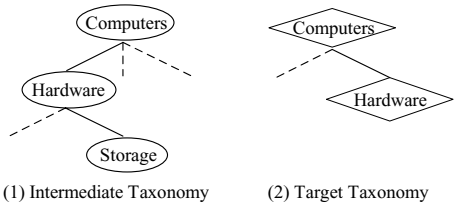

Figure 2: Illustration of the matching between taxonomies.

4. CLASSIFICATION APPROACHES

In this section, we first describe the state-of-the-art query classification methods. Then we describe our new bridging classifier to address the disadvantages of the existing methods.

4.1 Classification by Exact Matching

As described in Section 3.1, a query can be expanded through search engines which results in a list of related Web pages together with their categories from an intermediate taxonomy. A straightforward approach to QC is to leverage the categories by exact matching. We denote the categories in the intermediate taxonomy and the target taxonomy as C^I and C^T respectively. For each category in C^I, we can detect whether it is mapped to any category in C^T according to the matching approaches given in Section 3.2. After that, the most frequent target categories to which the returned intermediate categories have been successfully mapped are regarded as the classification result. That is:

$$c^* = \arg\max_{C_j^T} \left(\sum_{i=1}^{n} I(C^I(i) \; is \; mapped \; to \; C_j^T) \right) \quad (1)$$

In Equation (1), $I(\cdot)$ is the indicator function whose value is 1 when its parameter is true and 0 otherwise. $C^I(i)$ is the category in the intermediate taxonomy for the i^{th} page returned by the search engine. n result pages are used for query classification and the parameter n is studied in our experiments.

It is not hard to imagine that the exact matching approach tends to produce classification results with high precision but low recall. It produces high precision because this approach relies on the Web pages which are associated with the manually annotated category information. It produces low recall because many search result pages have no intermediate categories. Moreover, the exact matching approach cannot find all the mappings from the existing intermediate taxonomy to the target taxonomy which also results in low recall.

4.2 Classification by SVM

To alleviate the low-recall problem of the exact matching method, some statistical classifiers can be used for QC. In the KDDCUP 2005 winning solution, Support Vector Machine (SVM) was used as a base classifier. Query classification with SVM consists of the following steps: 1) construct the training data for the target categories based on mapping functions between categories, as discussed in Section 3.2. If an intermediate category C^I is mapped to a target category C^T, then the Web pages in C^I are mapped into C^T; 2) train SVM classifiers for the target categories; 3) for each Web query to be classified, use search engines to get its enriched features as discussed in Section 3.1 and classify the query using the SVM classifiers. The advantage of this QC method is that it can improve the recall of the classification result. For example, assume two intermediate categories, C_1^I and C_2^I, are semantically related with a target category C_1^T. C_1^I can be matched with C_1^T through word matching but C_2^I cannot. For a query to be classified, if a search engine only returns pages of C_2^I, this query cannot be classified into the target category if the exact matching classification method is used. However, if the query is classified by a statistical classifier, it can also be assigned the target category C_1^T, as the classifier is trained using pages of C_1^I, which may also contain terms of C_2^I because the two intermediate categories are similar in topic.

Although statistical classifiers can help increase the recall of the exact matching approach, they still need the exact matching for collecting the training data. What is more, if the target taxonomy changes, we need to collect the training data by exact matching and train statistical classifiers again. In the following sections, we develop a new method to solve the above problems.

4.3 Our New Method: Classifiers by Bridges

4.3.1 Taxonomy-Bridging Algorithm

We now describe our new QC approach called taxonomy-bridging classifier, or bridging classifier in short, by which we connect the target taxonomy and queries by taking an intermediate taxonomy as a bridge. The idea is illustrated in Figure 3, where two vertical lines separate the space into three parts. The square in the left part denotes the queries to be classified; the tree in the right part represents the target taxonomy; the tree in the middle part is an existing intermediate taxonomy. The thickness of the dotted lines reflects the similarly relationship between two nodes. For example, we can see that the relationship between C_i^T and C_j^I is much stronger than that between C_i^T and C_k^I. Given a category C_i^T in the target taxonomy and a query to be classified q_k, we can judge the similarity between them by the distributions of their relationship to the categories in the intermediate taxonomy. By defining the relationship and similarity under the probabilistic framework, the above idea can be explained by Equation (2).

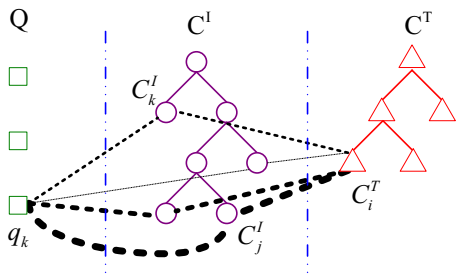

Figure 3: Illustration of the Bridging Classifier.

$$\begin{aligned} p(C_i^T|q) &= \sum_{C_j^I} p(C_i^T, C_j^I|q) \\ &= \sum_{C_j^I} p(C_i^T|C_j^I, q)p(C_j^I|q) \\ &\approx \sum_{C_j^I} p(C_i^T|C_j^I)p(C_j^I|q) \\ &= \sum_{C_j^I} p(C_i^T|C_j^I)\frac{p(q|C_j^I)p(C_j^I)}{p(q)} \\ &\propto \sum_{C_j^I} p(C_i^T|C_j^I)p(q|C_j^I)p(C_j^I) \end{aligned} \tag{2}$$

In Equation (2), $p(C_i^T|q)$ denotes the conditional probability of C_i^T given q. Similarly, $p(C_i^T|C_j^I)$ and $p(q|C_j^I)$ denotes the probability of C_i^T and q given C_j^I respectively. $p(C_j^I)$ is the prior probability of C_j^I which can be estimated from the Web pages in C^I. If C_i^T is represented by a set of words $(w_1, w_2, \ldots, w_n)$ where each word w_k appears n_k times, $p(C_i^T|C_j^I)$ can be calculated through Equation (3)

$$p(C_i^T|C_j^I) = \prod_{k=1}^{n} p(w_k|C_j^I)^{n_k} \tag{3}$$

where $p(w_k|C_j^I)$ stands for the probability that the word w_k occurs in class C_j^I, which can be estimated by the principle of maximal likelihood. $p(q|C_j^I)$ can be calculated in the same way as $p(C_i^T|C_j^I)$.

A query q can be classified according to Equation (4):

$$c^* = \arg\max_{C_i^T} p(C_i^T|q) \tag{4}$$

To make our bridging classifier easier to understand, we can explain it in another way by rewriting Equation (2) as Equation (5),

$$\begin{aligned} p(C_i^T|q) &= \sum_{C_j^I} p(C_i^T, C_j^I|q) \\ &= \sum_{C_j^I} p(C_i^T|C_j^I, q)p(C_j^I|q) \\ &\approx \sum_{C_j^I} p(C_i^T|C_j^I)p(C_j^I|q) \\ &= \sum_{C_j^I} \frac{p(C_j^I|C_i^T)p(C_i^T)}{p(C_j^I)}p(C_j^I|q) \\ &= p(C_i^T)\sum_{C_j^I} \frac{p(C_j^I|C_i^T)p(C_j^I|q)}{p(C_j^I)} \end{aligned} \tag{5}$$

Let us consider the numerator on the right side of the Equation (5). Given a query q and C_i^T, $p(C_j^I|C_i^T)$ and $p(C_j^I|q)$ are fixed and $\sum_{C_j^I} p(C_j^I|C_i^T) = 1$, $\sum_{C_j^I} p(C_j^I|q) = 1$. $p(C_j^I|C_i^T)$ and $p(C_j^I|q)$ represent the probability that C_i^T and q belong to C_j^I. It is easy to prove that $p(C_i^T|q)$ tends to be larger when q and C_i^T tends to belong to the same category in the intermediate taxonomy. The denominator $p(C_j^I)$ reflects the size of category C_j^I which acts as a weighting factor. It guarantees that the higher the probability that q and C_i^T belong to the smaller sized category (where size refers to the number of nodes underneath the category in the tree) in the intermediate taxonomy, the higher the probability that q belongs to C_i^I. Such an observation agrees with our intuition, since a larger category tends to contain more sub-topics while a smaller category contains fewer sub-topics. Thus we can say with higher confidence that q and C_i^I are related to the same sub-topic when they belong to the same smaller category.

4.3.2 Category Selection

The intermediate taxonomy may contain enormous categories and some of them are irrelevant to the query classification task corresponding with the predefined target taxonomy. Therefore, to reduce the computation complexity, we should perform "Category Selection" in a similar sense of "Feature Selection" in text classification [15]. Two approaches are employed in this paper to evaluate the goodness of a category in the intermediate taxonomy. After sorting the categories according to the scores calculated by the following two approaches, category selection can be fulfilled by selecting the top n categories.

Total Probability (TP): this method gives a score to each category in the intermediate taxonomy according to its probability of generating the categories in the target taxonomy, as shown in Equation (6).

$$Score(C_j^I) = \sum_{C_i^T} P(C_i^T|C_j^I) \tag{6}$$

Mutual Information (MI): MI is a criterion commonly used in statistical language modeling of word associations and other related applications [15]. Given a word t and a

category c, the mutual information between t and c is defined as:

$$MI(t,c) = \log \frac{P(t \wedge c)}{P(t) \times P(c)} \quad (7)$$

By considering the two-way contingency table for t and c, where A is the number of times t and c co-occur, B is the number of times that t occurs without c, C is number of times c occurs without t and N is the total number of documents, then the mutual information between t and c can be estimated using:

$$MI(t,c) \approx \log \frac{A \times N}{(A+C) \times (A+B)} \quad (8)$$

Since the name of a category in the target taxonomy usually contains more than one term, we define the "mutual information" between a category in the intermediate taxonomy C_j^I and a category in the target taxonomy C_i^T as:

$$MI(C_i^T, C_j^I) = \frac{1}{|C_i^T|} \sum_{t \in C_i^T} MI(t, C_j^I) \quad (9)$$

where $|C_i^T|$ is the number of terms in the name of C_i^T.

To measure the goodness of C_j^I in a global category selection, we combine the category-specific scores of C_j^I by:

$$MI_{avg}(C_j^I) = \sum_{C_j^T} MI(C_i^T, C_j^I) \quad (10)$$

4.3.3 Discussions

As we can see, in the bridging classifier, we do not need to train a classifier function between an intermediate taxonomy and the target taxonomy. We only need to build the classifiers on the intermediate taxonomy once and it can be applied to any target taxonomy. The framework can be extended in two directions. One is to include some training data for each target category. With the training data, we do not have to treat the labels of the target categories as queries and retrieve related Web pages through search engines to represent the categories. We can extract features from the training data directly. The second extension is to use other sophisticated models such as the n-gram model [9] or SVM [10] for computing $p(C_i^T|C_j^I)$ and $p(q|C_j^I)$.

5. EXPERIMENTS

In this section, we first introduce the data set and the evaluation metrics. Then we present the experiment results and give some discussions.

5.1 Data Set and Evaluation Metrics

5.1.1 Data sets

In this paper, we use the data sets from the KDDCUP 2005 competition which is available on the Web[1] . One of the data sets contains 111 sample queries together with the category information. These samples are used to exemplify the format of the queries by the organizer. However, since the category information of these queries is truthful, they can serve as the validation data. Another data set contains 800 queries with category information labeled by three human labelers. In fact, the organizers provided 800,000 queries in total which are selected from the MSN search logs for testing the submitted solutions. Since manually labeling all the 800,000 queries is too expensive and time consuming, the organizers randomly selected 800 queries for evaluation.

We denote the three human query-labelers (and sometimes the dataset labeled by them if no confusion is caused) as L1, L2 and L3, respectively. Each query has at most five labels in ranked order. Table 2 shows the average precision and F1 score values of each labeler when evaluated against the other two labelers. The average values among the three labelers are around 0.50 which indicates that the query classification problem is not an easy task even for human labelers. In this paper, all the experiments use only the 800 queries, except in the ensemble classifiers, where we use the 111 sample queries to tune the weight of each single classifier.

Table 2: The Average Scores of Each Labeler When Evaluated Against the Other Two Labelers

	L1	L2	L3	Average
F1	0.538	0.477	0.512	0.509
Pre	0.501	0.613	0.463	0.526

The existing intermediate taxonomy used in the paper is from Open Directory Project (ODP, http://dmoz.org/). We crawled 1,546,441 Web pages from ODP which spanned over 172,565 categories. The categories have a hierarchical structure as shown in Figure 2(1). We can consider the hierarchy at different levels. Table 3 shows the number of categories on different levels. The first row counts all the categories while the second row counts only the categories containing more than 10 Web pages. Table 4 summarizes the statistics of Web page numbers in the categories with more than 10 documents on different levels. As we can see, when we move down to the lower levels along the hierarchy, more categories appear while each category contains fewer Web pages. In order to remove noise, we consider the categories with more than 10 pages in this paper.

Table 3: Number of Categories on Different Levels

	Top 2	Top 3	Top 4	Top 5	Top All
#doc > 0	435	5,300	24,315	56,228	172,565
#doc > 10	399	4,011	13,541	23,989	39,250

Table 4: Statistics of the Numbers of Documents in the Categories on Different Levels

	Top 2	Top 3	Top 4	Top 5	Top All
Largest	211,192	153,382	84,455	25,053	920
Smallest	11	11	11	11	11
Mean	4,044.0	400.8	115.6	61.6	29.1

5.1.2 Evaluation Measurements

In KDDCUP 2005, precision, performance and creativity are the three measures to evaluate the submitted solutions. "creativity" refers to the novelty of the solutions judged by experts. The other two measures are defined according to the standard measures to evaluate the performance of classification, that is, precision, recall and F1-measure [12]. Pre-

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

cision (P) is the proportion of actual positive class members returned by the system among all predicted positive class members returned by the system. Recall (R) is the proportion of predicted positive members among all actual positive class members in the data. F1 is the harmonic mean of precision and recall as shown below:

$$F1 = 2 \times P \times R/(P + R) \quad (11)$$

"performance" adopted by KDDCUP 2005 is in fact F1. Therefore, we denote it by F1 instead of "performance" for simplicity.

As 3 labelers were asked to label the queries, the results reported are averaged over the values evaluated on each of them.

5.2 Results and Analysis

5.2.1 Performance of Exact matching and SVM

In this section, we study the performance of the two methods which tightly depend on word matching: exact matching and SVM, as well as the effect of query and category expansion. Table 5 shows the results of the category expansion through intermediate taxonomy by word matching, that is the results of collecting training data for the target taxonomy. Each element in the table represents the number of documents collected for the target categories. The first row contains the results by direct matching while the second row contains the results after expanding the category names through extended matching. We can see that after extending the names of the target categories, the number of documents collected for the target categories increases. We expect that the expansion with the help of WordNet should provide more documents to reflect the semantics of the target categories which is verified by Table 6.

Table 5: Number of Pages Collected for Training under Different Category Expansion Methods

	Min	Max	Median	Mean
Direct Matching	4	126,397	2,389	14,646
Extended Matching	22	227,690	6,815	21,295

Table 6 presents the result comparisons of the exact matching method and SVM. We enrich the query by retrieving the relevant pages through Google (http://www.google.com). The top n returned pages are used to represent the query where n varies from 20 to 80, with the step size of 20. Two approaches are used to extract features from the returned pages. One is to extract the snippet of the returned pages and the other is to extract all the text in the Web pages except the HTML tags. The Web pages' titles will be added to both of these two kinds of features. The column "0" means that we use only the terms in the query without enrichment.

In our experiments, we expand the target categories through the ODP taxonomy; that is, we collect the training data for the target categories from ODP. When constructing the mapping relationship as shown in Section 3.2, if we use direct matching, we denote SVM and the exact matching method with "SVM-D" and "Extact-D" respectively. Otherwise,if we use the extended matching method, we denote SVM and the exact matching method with "SVM-E" and "Extact-E" respectively. The exact matching method needs the category list of the retrieved Web pages for each query. The category information is obtained through Google's "Directory Search" service (http://www.google.com/dirhp).

Table 6: Performance of Exact Matching and SVM

(1)Measured by F1

n		0	20	40	60	80
Exact-D		Null	**0.251**	0.249	0.247	0.246
Exact-E		Null	0.385	**0.396**	0.386	0.384
SVM-D	snippet	0.205	0.288	**0.292**	0.291	0.289
	full text		0.254	**0.276**	0.267	0.273
SVM-E	snippet	0.256	0.378	**0.383**	0.379	0.379
	full text		0.316	**0.340**	0.327	0.336

(2) Measured by Precision

n		0	20	40	60	80
Exact-D		Null	**0.300**	0.279	0.272	0.268
Exact-E		Null	0.403	**0.405**	0.389	0.383
SVM-D	snippet	0.178	**0.248**	**0.248**	0.244	0.246
	full text		0.227	0.234	**0.242**	0.240
SVM-E	snippet	0.212	**0.335**	0.321	0.312	0.311
	full text		0.288	**0.309**	0.305	0.296

From Table 6 we can see that "Exact-E" is much better than "Exact-D", and "SVM-E" is much better than "SVM-D". This indicates that the extended matching with the help of WordNet can achieve a more proper representation of the target category. We can also observe that "Exact-E" performs better than "SVM-E". Another observation is that the "snippet" representation outperforms "full text" consistently. The reason is that the "snippet" provides a more concise context of the query than the "full text" which tends to introduce noise. We can also see that most of the classifiers achieve the highest performance when the queries are represented by the top 40 search result pages. Therefore, in the later experiments, we use snippets of the top 40 pages to represent queries.

5.2.2 Performance of the Bridging Classifier

As we can see in the above experiments, the thesaurus WordNet plays an important role in both the exact matching method and SVM since it can help expand the words in the labels of the target categories, which can further improve the mapping functions. However, the effect of a thesaurus may be limited due to the following reasons: 1) there may be no thesaurus in some fields; 2) it is hard to determine the precise expansion of the words even with a high-quality thesaurus, especially with the rapidly changing usage of words on the Web. Therefore, we put forward the bridging classifier which only relies on the intermediate taxonomies.

In order to expand a target category, we can treat its name as a query and submit it to search engines. We use the snippet of the top n returned pages to represent a category since we learned from the query expansion that snippet performs better than "full text". The parameter n varies from 20 to 100. Table 7 shows the results when "top all" categories in the ODP taxonomy are used for bridging the queries and the target taxonomy. The effect of different levels of the intermediate taxonomy will be studied later. From Table 7, we can

see that the bridging classifier achieves the best performance when n equals 60. The best F1 and precision achieved by the bridging classifier is higher than those achieved either by the exact matching method or SVM. The relative improvement is more than 10.4% and 7.1% in terms of precision and F1 respectively. The main reason for the improvement is that the bridging classifier can make thorough use of the finer grained intermediate taxonomy in a probabilistic way. While the previous methods including the exact matching method and SVM exploit the intermediate taxonomy in a hard way when constructing the mapping function as shown in Section 3.2.

Table 7: Performances of the Bridging Classifier with Different Representations of Target Categories

n	20	40	60	80	100
F1	0.414	0.420	**0.424**	0.421	0.416
Precision	0.437	0.443	**0.447**	0.444	0.439

Table 8: Performances of the Bridging Classifier with Different Granularity

	Top 2	Top 3	Top 4	Top 5	Top All
F1	0.267	0.285	0.312	0.352	**0.424**
Precision	0.270	0.291	0.339	0.368	**0.447**

Table 8 shows the performance of the bridging classifier when we change the granularity of the categories in the intermediate taxonomy. To change the granularity of the categories, we use the categories on the top L level by varying L. It is clear that the categories have larger granularity when L is smaller. From Table 8, we can see that the performance of the bridging classifier improves steadily by reducing the granularity of categories. The reason is that categories with large granularity may be a mixture of several target categories which prohibit distinguishing the target categories.

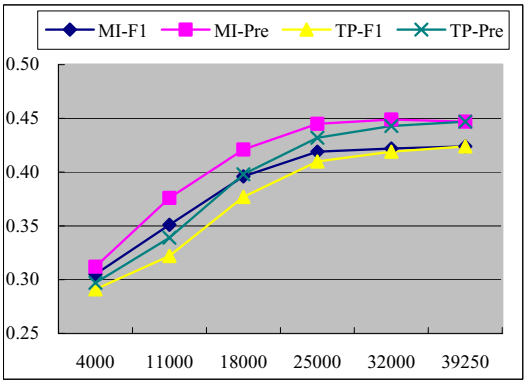

Figure 4: Effect of category selection.

However, reducing the granularity of categories in the intermediate taxonomy will certainly increase the number of the intermediate categories which will thus increase the computation cost. One way to solve this problem is to do category selection. Figure 4 shows the performance of the bridging classifier when we select the categories from all the ODP taxonomy through the two category selection approaches proposed in Section 4.3.2. We can see that when the category number is around 18,000, the performance of the bridging classifier is comparable to, if not better than, the previous approaches, including the exact matching method and SVM. MI works better than TP in that MI can not only measure the relevance between the categories in the target taxonomy and those in the intermediate taxonomy, but also favors the categories which are more powerful to distinguish the categories in the target taxonomy. However, TP only cares about the merit of relevance.

5.2.3 *Ensemble of Classifiers*

The winner of the KDDCUP 2005 competition found that the best result was achieved by combining the exact matching method and SVM. In the winning solution, besides the exact matching method on Google's directory search, two other exact matching methods are developed using LookSmart (http://www.looksmart.com) and a search engine based on Lemur (http://www.lemurproject.org) and their crawled Web pages from ODP [11]. Two classifier-combination strategies are used, with one aiming at higher precision (denoted by EV, where 111 samples are used as the validation data to tune the weight of each base classifier) and the other aiming at higher F1 (denoted by EN in which the validation data set is ignored). EV assigns a weight to a classifier proportional to the classifier's precision while EN gives equal weights to all classifiers. We follow the same strategy to combine our new method with the winner's methods, which is denoted as *"Exact-E"+"SVM-E"+Bridging* as shown in Table 9. The numbers in the parentheses are the relative improvement. Note that the bridging classifier alone achieves similar F1 measurement as the KDDCU 2005 winning solution ("Exact-E"+"SVM-E" with the EV combination strategy) but improves the precision by 5.4%. From Table 9 we can also find that the combination of the bridging classifier and the KDDCUP 2005 winning solution can improve the performance by 9.7% and 3.8% in terms of precision and F1, respectively, when compared with the winning solution. This indicates that the bridging classifier works in a different way as the exact matching method and SVM, and they are complimentary to each other.

Table 9: Performances of Ensemble Classifiers

		"Exact-E" + "SVM-E"	"Exact-E" + "SVM-E" +Bridging
EV	F1	0.426	0.429(+0.007)
	Precision	0.424	0.465(+0.097)
EN	F1	0.444	0.461(+0.038)
	Precision	0.414	0.430(+0.039)

6. RELATED WORK

Though not much work has been done on topical query classification, some work has been conducted on other kinds of query classification problems. Gravano et al. classified the Web queries by geographical locality [3] while Kang et al. proposed to classify queries according to their functional types [4].

Beitzel et al. studied the same problem in [2] as we pursued in this paper, with the goal to classify the queries according to their topic(s). They used two primary data sets

containing the queries from the AOL web search service. These queries were manually classified into a set of 18 categories. The main difference between our problem and that of [2] is that we did not have training data as given input. In fact, it is a very difficult and time consuming task to provide enough training examples, especially when the target taxonomy is complicated. Another potential problem related to the training data, as pointed out in [2], is caused by the ongoing changes in the query stream, which makes it hard to systematically cover the space of queries. In this paper, we just rely on the structure and category names of the target taxonomy without training data, which is consistent with the task of KDDCUP 2005.

KDDCUP 2005 provides a test bed for the Web query classification problem. There are a total of 37 solutions from 32 teams attending the competition. As summarized by the organizers [6], most solutions expanded the queries through search engines or WordNet and expanded the category by mapping between some pre-defined/existing taxonomy to the target taxonomy. Some solutions require human intervention in the mapping process [5, 13].

Besides classifying the queries into target taxonomy, we can also cluster the queries to discover some hidden taxonomies through unsupervised methods. Both Beeferman [1] and Wen [14] used search engines' clickthrough data to cluster the queries. The former makes no use of the actual content of the queries and URLs, but only how they co-occur within the clickthrough data, while the latter exploits the usage of the content. Although the work in [1] and [14] proved the effectiveness of the clickthrough data for query clustering, we did not utilize them in our solution due to the following two reasons: 1) the clickthorugh data can be quite noisy and is search engine dependent; 2) it is difficult to obtain the clickthrough data due to privacy and legal issues.

7. CONCLUSION AND FUTURE WORK

This paper presented a novel solution for classifying Web queries into a set of target categories, where the queries are very short and there are no training data. In our solution, an intermediate taxonomy is used to train classifiers bridging the queries and target categories so that there is no need to collect the training data. Experiments on the KDDCUP 2005 data set show that the bridging classifier approach is promising. By combining the bridging classifier with the winning solution of KDDCUP 2005, we made a further improvement by 9.7% and 3.8% in terms of precision and F1 respectively compared with the best results of KDDCUP 2005. In the future, we plan to extend the bridging classifier idea to other types of query processing tasks, including query clustering. We will also conduct research on how to leverage a group of intermediate taxonomies for query classification.

8. ACKNOWLEDGMENTS

Dou Shen and Qiang Yang are supported by a grant from NEC (NECLC05/06.EG01). We thank the anonymous reviewers for their useful comments.

9. REFERENCES

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

[2] S. M. Beitzel, E. C. Jensen, O. Frieder, D. Grossman, D. D. Lewis, A. Chowdhury, and A. Kolcz. Automatic web query classification using labeled and unlabeled training data. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 581–582, 2005.

[3] L. Gravano, V. Hatzivassiloglou, and R. Lichtenstein. Categorizing web queries according to geographical locality. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 325–333, 2003.

[4] I.-H. Kang and G. Kim. Query type classification for web document retrieval. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 64–71, 2003.

[5] Z. T. Kardkovács, D. Tikk, and Z. Bánsághi. The ferrety algorithm for the kdd cup 2005 problem. *SIGKDD Explor. Newsl.*, 7(2):111–116, 2005.

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

[7] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classication. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[8] G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Introduction to wordnet: an on-line lexical database. *International Journal of Lexicography*, 3(4):23–244, 1990.

[9] F. Peng, D. Schuurmans, and S. Wang. Augmenting naive bayes classifiers with statistical language models. *Inf. Retr.*, 7(3-4):317–345, 2004.

[10] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*. MIT Press, 1999.

[11] D. Shen, R. Pan, J.-T. Sun, J. J. Pan, K. Wu, J. Yin, and Q. Yang. Q2c@ust: our winning solution to query classification in kddcup 2005. *SIGKDD Explor. Newsl.*, 7(2):100–110, 2005.

[12] R. C. van. *Information Retrieval*. Butterworths, London, second edition edition, 1979.

[13] D. Vogel, S. Bickel, P. Haider, R. Schimpfky, P. Siemen, S. Bridges, and T. Scheffer. Classifying search engine queries using the web as background knowledge. *SIGKDD Explor. Newsl.*, 7(2):117–122, 2005.

[14] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Query clustering using content words and user feedback. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 442–443, 2001.

[15] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 412–420, 1997.