Song R, Dou Z, Hon HW *et al.* Learning query ambiguity models by using search logs. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 25(4): 728–738 July 2010. DOI 10.1007/s11390-010-1056-9

Learning Query Ambiguity Models by Using Search Logs

Ruihua Song[1,2] (宋睿华), *Member*, *ACM*, Zhicheng Dou[2] (窦志成), Hsiao-Wuen Hon[2] (洪小文), *Fellow*, *IEEE* and Yong Yu[1] (俞　勇)

[1] *Department of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, China*

[2] *Microsoft Research Asia, Beijing 100190, China*

E-mail: {rsong, zhichdou, hon}@microsoft.com; yyu@cs.sjtu.edu.cn

Received May 15, 2009; revised February 23, 2010.

Abstract Identifying ambiguous queries is crucial to research on personalized Web search and search result diversity. Intuitively, query logs contain valuable information on how many intentions users have when issuing a query. However, previous work showed user clicks alone are misleading in judging a query as being ambiguous or not. In this paper, we address the problem of learning a query ambiguity model by using search logs. First, we propose enriching a query by mining the documents clicked by users and the relevant follow up queries in a session. Second, we use a text classifier to map the documents and the queries into predefined categories. Third, we propose extracting features from the processed data. Finally, we apply a state-of-the-art algorithm, Support Vector Machine (SVM), to learn a query ambiguity classifier. Experimental results verify that the sole use of click based features or session based features perform worse than the previous work based on top retrieved documents. When we combine the two sets of features, our proposed approach achieves the best effectiveness, specifically 86% in terms of accuracy. It significantly improves the click based method by 5.6% and the session based method by 4.6%.

Keywords ambiguous query, log mining, query classification

1 Introduction

Ambiguous queries are an important portion of real Web queries. Previous work estimates that about 16% of Web queries have more than one meaning[1]. Users may have completely different information needs when issuing an ambiguous query. This has motivated research on personalized search for a long time and, most recently, search results diversity.

Identifying an ambiguous query is in demand for both personalized search and search results diversity. One goal of personalization technologies is to enhance user experiences for ambiguous queries, such as "java" and "apple". When different users are searching the same query, search results are re-ranked according to user preferences, such as preferred categories. However, most previous work applied personalization to all queries, although many queries cannot be improved or are even hurt by personalization[2]. If we can identify an ambiguous query, we can apply personalization technologies to these kinds of queries and thus improve user experiences. Similar to this application, we can apply diversification technologies that are specifically helpful for ambiguous queries if these kinds of queries are identified. In addition, studies on search result diversification are hampered by a lack of test collections containing ambiguous queries, as Sanderson argued in [3]. Identifying ambiguous queries from logs is important for constructing a large-scale representative query set. The problem is that it is difficult and time-consuming to identify ambiguous queries manually. If, without researching on a query, an assessor is not confident in judging an ambiguous query due to the limited background knowledge, one question arises: can we identify an ambiguous query automatically?

Most of the past word sense disambiguation work focused on ambiguity of words found in dictionaries, which have a poor coverage of real Web queries[3]. The most relevant previous work was done by Song *et al.*[1] To clarify the various terms used in the literature, such as "ambiguous query", "sub-topic query", "general term", "broad topic", and "diffuse topic", they summarized query taxonomy as follows:

- Type A (Ambiguous Query): a query that has more than one meaning. For example, "giant" could refer to "Giant Company Software, Inc" (an Internet security software developer), "Giant" (a film produced in 1956), "Giant Bike" (a bicycle manufacturer), or "San Francisco Giants" (National League baseball team).
- Type B (Broad Query): a query that covers a

Regular Paper

variety of subtopics. In practice, users issue such a query first, and then narrow down to a subtopic. For example, "songs" covers subtopics such as "song lyrics", "love songs", "party songs", and "download songs".

• Type C (Clear Query): a query that covers a narrow topic and has a specific meaning. For example, "University of Chicago" and "Billie Holiday". A clear query usually means a successful search, and users can find high quality results in the first results page.

Their user study shows that, if classifying queries into type A and type $\bar{A}$ (i.e., type B and C), human annotators' judgments reach an agreement as high as 90%, while it is difficult for the annotators to discriminate whether a query belongs to type B or type C. Furthermore, they find that human annotators tend to classify ambiguous queries as clear queries if logs are used to assist labeling. They propose using top retrieved documents to learn a binary classifier for ambiguous queries.

We adopt the definition of ambiguous queries and the goal of learning a binary classifier as [1] does. Different from the work, we argue that if using query logs in a grace way, it is possible to identify ambiguous queries effectively. Actually, query logs contain valuable data on diverse information needs from real users. Click-through logs would be helpful, because when users have different goals by searching a query, most likely they will browse the result list and click on different pages. Another useful source is a query session containing the queries that users submit consecutively in a short time. Radlinski and Dumais[4] proposed to select related queries from sessions to retrieve more documents on different intentions before diversifying the results. In addition, we hypothesize that user clicks and query sessions have complementary information of different user needs. For example, when users' target meanings are not shown on the first search result page, they may have to reformulate the query and search again. As a result, although the information on users' diverse requirements is not available in click-through logs, we can find some clues in session logs. Therefore, in this paper, we take advantage of both click logs and session logs in ambiguous query classification.

To the best of our knowledge, this paper is the first work that uses query logs to address the problem of automatic ambiguous query identification. First, given a query, we represent users and their clicks as a user-document matrix and propose rules to extract relevant follow up queries from sessions. Second, we make use of a text classifier to map the clicked documents and the relevant queries into predefined categories as used in KDD-Cup 2005[5]. Our observation is that, in most cases, the documents or queries on different interpretations of a given query belong to different categories. Third, we propose a series of features that measure the distribution of clicked documents and relevant queries in category space. Based upon these features, we apply the Support Vector Machine (SVM) algorithm[6] to learn a query ambiguity classifier.

Experimental results indicate that both click features and session features perform worse than the top retrieved document features proposed in [1]. This confirms the challenges of using query logs in ambiguous query classification. However, when combining these two groups of features extracted from query logs, we achieve dramatically better performance, which is also comparable to the previous approach. The significant improvements support our hypothesis that user clicks and query sessions are complementary.

The rest of the paper is organized as follows. Section 2 reviews the literature. Section 3 provides the details on our proposed log based approach. In Section 4, we conduct experiments to compare approaches. We conclude the paper and discuss our future work in Section 5.

2 Related Work

In this section, we examine the related past work in the following topics: research on word sense ambiguity, research on search result diversity, and research on query classification.

2.1 Word Sense Ambiguity

Word sense ambiguity has been studied for many years in the Natural Language Processing (NLP) community. The problem of word sense disambiguation (WSD) is formulated as selecting a sense for a word from a set of predefined possibilities. A comprehensive review of the studies on WSD can be found in Mihalcea and Pedersen's tutorial[7]. Most of the disambiguation work focused on ambiguity of words in dictionaries.

Query ambiguity is a different topic from word sense ambiguity and worth further research. Dictionaries have poor coverage of common queries such as proper nouns (e.g. acronyms) and phrases that have more than one word[1,3]. Sanderson[3] compared ambiguous words in WordNet (http://wordnet.princeton.edu), a widely used thesaurus in NLP, and Wikipedia (http://www.wikipedia.org), a user contributed Web thesaurus in which entries are closer to queries in search engines. He found obvious differences in statistics. For example, only 7% of ambiguous words in WordNet are multi-term words, while 39% of ambiguous entries in Wikipedia are multi-term words. The user study

conducted by Song *et al.* [1] shows that about one third of the sampled queries cannot be found in either traditional dictionaries or Wikipedia. Previous work[8-9] also found that disambiguation of word sense rarely benefits retrieval as the meaning of an ambiguous query word can be clarified by the other query words. For example, despite the word "bank" being ambiguous, the query "Bank of America" is unlikely to retrieve top ranked documents on a river bank.

2.2 Search Result Diversity

Previous work proposed general approaches to diversify rankings for queries. In terms of whether a topic taxonomy, such as Open Directory Project (ODP) (http://www.dmoz.org), is used, the diversification approaches can be split into two kinds. Maximal Marginal Relevance (MMR) is an influential piece of work in the group without any topic taxonomy[10]. In this, Carbonell and Goldstein optimized a trade-off function between novelty and relevance. Zhai *et al.* presented the problem called sub-topic retrieval[11] and proposed using the correlations among the results to solve the problem[12]. Chen and Karger[13] developed a greedy optimization algorithm to approximately optimize the metrics that consider diverse information needs. Different from this kind of work, Agrawal *et al.*[14] classified queries and documents into a predefined topic taxonomy, and created a diverse set of results in accordance with the taxonomy.

The approaches are supposed to both ensure that different sub-topics of a broad query are retrieved and ensure different interpretations of an ambiguous query are retrieved. However, as Sanderson noted in [3], "without test collection containing ambiguous topics with associated relevance judgments that reflect a range of interpretations of that topic, the worth of much of the work described here may not be fully understood." Identifying ambiguous queries from logs is exactly the first step when constructing a test collection. Our work aims to address this problem.

2.3 Query Classification

There is a lot of previous work on query classification for different applications. Some work classifies queries as informational or navigational[15-16]. For example, Lee, Liu, and Cho took advantage of user-click behavior and anchor-link distribution, because for a navigational query, there is a dominant URL that is clicked by users for most of time and/or is linked by the majority of anchors that match the query well[15]. Some work classifies queries into a topic taxonomy. For example, in the KDD-Cup 2005 competition[5], there was a task to categorize 800,000 queries to predefined topics. As a query is short, most of participants gathered extra content, like search result snippets and titles, and applied text categorization approaches. Some work used binary classification to identify a particular kind of query, such as commercial intention[16] or location intention[17].

Our work is in the last group and our goal is to identify ambiguous queries. At the same time we exploit text classification technology to generate effective features. Our addressed problem is similar to that in [1] as described in Section 1. The difference is that we focus on using query logs instead of search results. We will compare these methods by experiments in Section 4.

3 Our Approach

In this section, we formulate the problem of ambiguous query classification and propose using search logs to address the problem.

3.1 Problem Formulation and Framework Overview

Similar to [1], we formulate the problem of identifying ambiguous queries into a classification problem:

$$f : Q \longmapsto \{A, \bar{A}\}$$

Q is the query space:

$$Q = \{q_1, q_2, \ldots\}$$

f is the function that identifies whether a query is ambiguous:

$$f(q_i) = \begin{cases} A, & \text{if } q_i \text{ is ambiguous,} \\ \bar{A}, & \text{if } q_i \text{ is broad or clear.} \end{cases}$$

The framework of learning the model f is shown in Fig.1.

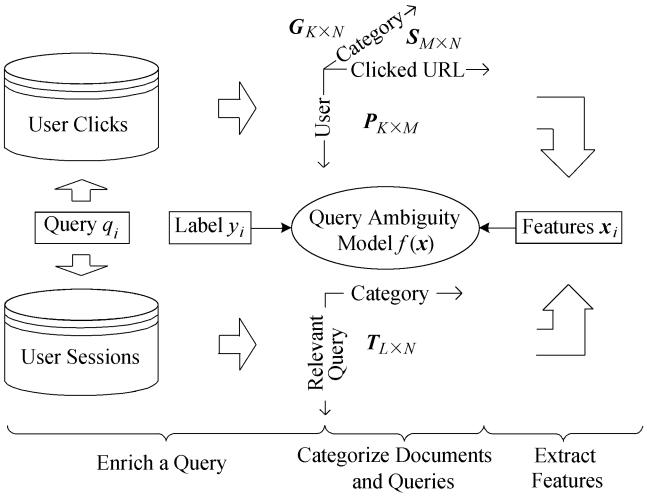

Fig.1. Framework of ambiguous query classification based on search logs.

Given a query q_i, we first use search logs to enrich the information on q_i. By using click-through data, we know which URLs were clicked and by who. From user sessions, we can mine frequent query chains in which the queries are searched consecutively. Thus we get the relevant follow up queries that follow q_i if there are any. Then, to conquer the sparseness of queries and clicks, we take advantage of text categorization technologies to classify the clicked documents and the relevant queries into predefined categories. Finally, we extract features $\boldsymbol{x}_i$ from our mined data to represent the query q_i.

We assume that we have some queries pre-labeled and thus each labeled query can be represented as $(\boldsymbol{x}_i, y_i)$ where y_i is its ambiguity label and $y_i \in \{-1, 1\}$. Then we apply SVM[6] to learn a classifier that minimizes the generalization error, or at least an upper bound on it, from the training data. We use the Smox toolkit that is based on Platt's Sequential Minimal Optimization algorithm [18]. Through experiments, we finally select the nonlinear SVM with RBF (Radial Basis Function) kernel as our setting because it always outperforms the linear SVM on our dataset.

As our work focuses on how to represent a query by the features extracted from logs, we will describe the related modules from Subsection 3.2 to Subsection 3.4.

3.2 Enriching a Query

We obtain half-year processed search logs from Bing Search (http://www.bing.com). The time spans from January 2008 to June 2008. In this section, we describe how to use user clicks and sessions to enrich a query by supplying the information on various user needs.

3.2.1 Using User Clicks

In the search logs, all real user related information has been stripped to ensure user privacy. Different users can only be distinguished by processed IDs. Furthermore, the search engine logs the query terms and all clicked URLs with ranks at search time.

For a given query, suppose there are K users who ever issued this query and there are M documents that were clicked. We count click times on a document by a user, and represent them in a $K \times M$ matrix $\boldsymbol{X}$. The element $x_{k,m} = c$ indicates that user k clicked document m a c amount of times. If the user has not clicked the document, then $x_{k,m} = 0$.

When dividing each element $x_{k,m}$ by the sum of all elements in the row, we get the click probability:

$$p_{k,m} = \frac{x_{k,m}}{\sum_{i=1}^{M} x_{k,i}}.$$

Thus, we transform $\boldsymbol{X}$ to the matrix $\boldsymbol{P}_{K\times M}$.

$$\boldsymbol{P} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K]^{\mathrm{T}}, \quad \boldsymbol{u}_k = [p_{k,1}, \ldots, p_{k,M}]^{\mathrm{T}}$$

Here, a row vector $\boldsymbol{u}_k^{\mathrm{T}}$ represents the probabilities of documents that were clicked by the user k.

The user-document matrix $\boldsymbol{P}$ is useful in identifying ambiguous queries. When users have different intentions by issuing a query, the vectors may diverge from each other. However, for a non-ambiguous query, users click a bunch of similar URLs and thus the vectors are also similar.

3.2.2 Using User Sessions

We construct session data by considering the time interval between two queries issued by a user. We separate two consecutive queries into two sessions if the interval exceeds 30 minutes as Cao *et al.* proposed in [19]. Then we apply their proposed method to discover frequent co-occurring queries. Some mined examples are shown in Table 1.

Table 1. Examples of Relationship Between Queries in Sessions

Query Relation	Example Session
Acronym	act ⇒ acceptance and commitment therapy
Generalization	macromedia flash player ⇒ flash player
Specialization	act ⇒ act scores
	act ⇒ act mouthwash
	hotmail ⇒ hotmail australia
Spelling Change	ask jeves ⇒ ask jeeves
Peer Queries	act (American College Test) ⇒ SAT (Reasoning test)
	Cannon digital camera ⇒ Nikon digital camera
Unrelated Queries	ask jeeves ⇒ mapquest
	ask jeeves ⇒ youtube
	ask jeeves ⇒ hotmail

In addition to some well-known relationships, such as acronym, specialization, generalization, and spelling changes, we find two other cases, in which the user goal may drift to another. In the first case, users may ask a query that is a peer/sibling to the original query. For example, the "SAT" test was queried after "ACT" because both are tests related to college education. In the second case, users may issue totally unrelated queries, e.g., "youtube" and "hotmail" after "ask jeeves", because whatever a user is seeking, he/she probably leaves from the old topic and navigates to a popular website within 30 minutes. Although the original query has a

clear intention, the next irrelevant queries may damage the effectiveness of an ambiguous query classifier. Therefore, it is needed to filter such noisy queries.

One intuitive method of filtering popular website queries is based on session frequency, i.e., how many sessions a query appears in. The queries with a frequency higher than a threshold can be regarded as stop-word queries. However, we find two problems in trying this method: 1) the stop-word list cannot cover many irrelevant queries because the follow up queries are not so frequently-asked but they are what the users happen to be interested in, e.g., "volvo" after "ask jeeves"; 2) if the original query is a popular query, such as "hotmail", the stop-word list will filter relevant queries. Consequently, we do not use session frequency, but propose using content to filter irrelevant queries.

Suppose q_{next} is a follow up query issued after the original query q. q_{next} is considered as relevant to q if it satisfies any of the following two conditions:

1) q_{next} and q share at least one exact same term;

2) q is judged as the acronym of q_{next} according to a simple rule, i.e., q matches the concatenation of the first characters from each term in q_{next} with or without stop-words.

All the relevant follow up queries compose a set $R(q)$. We will show how the two conditions work by experiments in Subsection 4.5.

The relevant follow up queries $R(q)$ contain useful information on the ambiguity of the query q. For an ambiguous query, users probably clarify their information needs by re-submitting queries. Take "act" as an example. A user submits the query "act scores" next, which means he/she is looking for the information on American College Test. A user refines the query as "acceptance and commitment therapy", which has nothing related to the test. Another user looks for ACT fluoride rinse and then he issues "ACT mouthwash". We see that the follow up queries can easily distinguish between different user needs. However, for a clear query, the follow up queries are expected to be on the same topic. Therefore, we will make use of $R(q)$ in identifying ambiguous queries.

3.3 Categorizing Documents and Queries

Although two clicked documents or two relevant queries are different, they may be on the same intention of the query. For example, both "act scores" and "act exam dates" are about the test of ACT (American College Test). Fortunately, they are similar to each other in terms of the categories that they belong to. Thus we make use of a text classifier to map the clicked documents and the relevant queries into categories. In this paper, we use pre-defined categories to identify these kinds of documents and queries.

3.3.1 Text Classification

We apply a text classifier similar to that used in [20] to classify a document/query into the predefined taxonomy provided by KDD-Cup'05[5].

As defined, there are 10 first-level categories and 64 second-level categories. Usually, the classifier outputs the top five second-level categories that the document/query most likely belongs to and the confidence $s_{m,i}$ corresponding to the category c_i. Thus, the document d_m is represented as:

$$d_m = \langle s_{m,1}, s_{m,2}, \ldots, s_{m,n} \rangle$$

where n is the number of categories, e.g., $n = 64$ if we use the second-level categories. Take the document "http://www.uchicago.edu/" as an instance. It is classified into the following categories:

$$\begin{aligned} &\textit{Library}\backslash\textit{Education}\\ &\textit{Library}\backslash\textit{Society}\\ &\textit{Work\&Money}\backslash\textit{Business}\\ &\textit{Library}\backslash\textit{Sciences}\\ &\textit{Library}\backslash\textit{Humanities} \end{aligned} \quad (1)$$

with the confidence 0.2150, 0.1200, 0.0933, 0.0839, and 0.0486 respectively. Consequently, we represent the document as:

$$\langle \ldots, 0.215, 0.12, 0.0839, 0.0486, \ldots, 0.0933, \ldots \rangle.$$

The first-level categories may be more reliable than the second-level categories in predicting query ambiguity. Given a query, the related documents or queries may belong to different second-level categories, although all of them talk about the same interpretation of the query. In contrast, if the related documents or queries belong to different first-level categories, we have more confidence that they refer to different interpretations of the query. Thus we aggregate the confidence of all sub-categories into the first-level categories as a complementary vector to the second-level one. In the above example, we get the following first-level vector:

$$\langle 0, 0, 0.4675, 0, 0, 0, 0, 0, 0, 0.0933 \rangle.$$

In our experiments, we extract features from both the first-level vector and the second-level vector.

To smooth a document/query vector, we normalize the confidence $s_{m,i}$ by the following equation to make sure that $\sum_{i=0}^{N} s_{m,i} = 1$:

$$s_{m,i} = s^o_{m,i} + \frac{1}{N}\Big(1 - \sum_{j=0}^{N} s^o_{m,j}\Big)$$

where $s^o_{m,i}$ is the original confidence returned by the classifier. We equally distribute the remaining confidence, i.e., one minus the sum of all non-zero confidence, to all categories. For example, the zeros in the above first-level vector are smoothed to 0.0549, and the last confidence is normalized to 0.1482.

For a query, we adopt a search snippets based method proposed in [20] to classify the query. In our experiments, we use Bing Search to retrieve the top 20 results for each query in $R(q)$ and regard the 20 snippets as a query document. Similar to a document vector, the categories that the query document belongs to are used to compose a query vector. At last, we smooth the query vectors, of both first-level categories and second-level categories, in the same way as described above.

3.3.2 Category Related Matrices

By categorizing the clicked documents of the matrix $\boldsymbol{P}$ and the relevant queries in $R(q)$, we then obtain three category related matrices.

First, given M clicked documents, we use a document-category matrix $\boldsymbol{S}_{M\times N}$ to represent the categories of documents. Each element $s_{m,n}$ is the smoothed confidence with which the document m is categorized into the category n.

$$\boldsymbol{S} = [\boldsymbol{d}_1, \ldots, \boldsymbol{d}_M]^{\mathrm{T}}, \quad \boldsymbol{d}_m = [s_{m,1}, \ldots, s_{m,N}]^{\mathrm{T}}$$

where each row vector $\boldsymbol{d}^{\mathrm{T}}_m$ denotes the probabilities of categories that the document m belongs to.

Second, we compute the product of matrices $\boldsymbol{P}_{K\times M}$ and $\boldsymbol{S}_{M\times N}$, and deduce a user-category matrix $\boldsymbol{G}_{K\times N}$ for this query:

$$\boldsymbol{G} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_K]^{\mathrm{T}}, \quad \boldsymbol{a}_k = [g_{k,1}, \ldots, g_{k,N}]^{\mathrm{T}}$$

where each row vector $\boldsymbol{a}^{\mathrm{T}}_k$ corresponds to user k's category preferences for the query. $g_{k,n}$ is the probability that user k selected documents which belong to category n.

$$g_{k,n} = \sum_{i=0}^{M} p_{k,i} * s_{i,n}.$$

Third, suppose there are L relevant queries in $R(q)$, we use a query-category matrix $\boldsymbol{T}_{L\times N}$ to represent the categories of relevant follow up queries. Each element $s_{l,n}$ is the smoothed confidence with which the query l is categorized into the category n.

$$\boldsymbol{T} = [\boldsymbol{q}_1, \ldots, \boldsymbol{q}_L]^{\mathrm{T}}, \quad \boldsymbol{q}_l = [s_{l,1}, \ldots, s_{l,N}]^{\mathrm{T}}$$

where each row vector $\boldsymbol{q}^{\mathrm{T}}_l$ denotes the probabilities of categories that the relevant query l belongs to.

Now we have represented the information from logs as four matrices: $\boldsymbol{P}$, $\boldsymbol{S}$, $\boldsymbol{G}$ from user clicks and $\boldsymbol{T}$ from user sessions.

3.4 Extracting Features

In this section, we extract features for a given query. The features are designed to distinguish ambiguous queries from others.

3.4.1 Basic Features

There are three basic features that may be useful in identifying ambiguous queries:

- *TermNum*: number of terms in the query;
- *AvgClkTimes*: average number of clicked URLs over users;
- *AvgMaxClkPos*: average maximum rank of clicked URLs over users.

Here, *TermNum* is a general feature, while *AvgClkTimes* and *AvgMaxClkPos* are extracted from user click logs.

The number of query terms is assumed to be useful because most of the ambiguous queries are very short, usually one-term only. When a query is vague, users may have to scan deep and click various documents to seek different information. Thus, *AvgClkTimes* and *AvgMaxClkPos* are supposed to be large.

3.4.2 Matrix Based Features

We extract a group of features that measure diversity over the matrices $\boldsymbol{P}$, $\boldsymbol{S}$, $\boldsymbol{G}$, and $\boldsymbol{T}$.

The basic idea is that when a query is ambiguous, users have different information needs and thus their behaviors are diverse. For example, the clicked URLs from one user may be quite different from those clicked by another. This can be partially measured by the divergence of the user-document matrix $\boldsymbol{P}$. If mapping the clicked documents into the category space, we observe that the user clicked documents are probably scattered in different categories for an ambiguous query. We propose to measure this by the diversity of the document-category matrix $\boldsymbol{S}$ and the user-category matrix $\boldsymbol{G}$. In addition, when a query is vague, users may refine the query in the same session and the next relevant queries may diverge in topics. Such divergence corresponds to the query-category matrix $\boldsymbol{T}$.

We use entropy, diameter, mean of distance, etc. to measure diversity over matrices. The distance between a pair of vectors can be also calculated by three formulas: Euclidean distance, Jensen-Shannon divergence, and cosine distance.

To describe what a feature means, we name the feature as "{Matrix}-{Diversity Measure}-{Distance

Formula}". The first part is based on which matrix the feature is extracted. The matrix could be $\boldsymbol{P}$, $\boldsymbol{S}$, $\boldsymbol{G}$, or $\boldsymbol{T}$. The second part is which measure we choose to quantitate diversity over the matrix. Given a general matrix:

$$\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_U]^{\mathrm{T}}, \quad \boldsymbol{v}_i = [p_i^1, \ldots, p_i^W]^{\mathrm{T}}.$$

We denote the centroid of all $\boldsymbol{v}_i$ as $\boldsymbol{v}_c$.

$$\boldsymbol{v}_c = [p_c^1, \ldots, p_c^W]^{\mathrm{T}}.$$

We use the following diversity measures:

- *Entropy*:

$$\sum_{w=1}^{W} (-p_c^w \log_2 p_c^w).$$

- *Diameter*:

$$\max_{\forall i,j} Dis(i,j)$$

where, $Dis(i,j)$ is the distance between $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$.

- *DMean and DSD*:

Mean and Standard Deviation over the set $\{Dis(i,j)\}$.

- *Radius*:

$$\max_{\forall i} Dis(i,c).$$

- *RMean and RSD*:

Mean and Standard Deviation over the set $\{Dis(i,c)\}$.

In the third part, distance can be calculated by the following formulas:

- *Euclidean Distance*:

$$Dis_{\mathrm{euc}}(i,j) = \sqrt{\sum_{w=1}^{W} (p_i^w - p_j^w)^2}.$$

- *Jensen-Shannon Divergence Distance*[21]:

$$Dis_{\mathrm{jsd}}(i,j) = \sqrt{D(i,j) + D(j,i)}$$

where,

$$D(i,j) = \sum_{w=1}^{W} p_j^w \log \frac{p_i^w}{\frac{1}{2}(p_i^w - p_j^w)}.$$

- *Cosine Distance*:

$$Dis_{\cos}(i,j) = -\frac{\boldsymbol{v}_i \cdot \boldsymbol{v}_j}{|\boldsymbol{v}_i||\boldsymbol{v}_i|}.$$

For example, the feature *T-Diameter-cos* is the maximum cosine distance between any pair of row vectors of the matrix $\boldsymbol{T}$. It is supposed to measure how diverse the categories of related queries are.

4 Experiments

In this section, we conduct experiments to evaluate our proposed methods and compare them with previous work. First, we select a minimal set of useful features for each method and tune parameters of SVM. Second, we compare the effectiveness of our proposed log based method with different methods, such as the user click based method, the user session based method, and the top retrieved documents based method proposed in [1]. Third, a set of learning experiments are conducted to show the insights on which features are the most useful in discovering ambiguous queries. Fourth, over a set of labeled follow up queries, we evaluate three variant methods of generating relevant queries from user sessions and investigate the relations between the effectiveness of identifying relevant queries and that of classifying ambiguous queries.

4.1 Experimental Setup

Our dataset of queries with ambiguity labels comes from the data used in [1]. In the original dataset, there are 253 queries comprising 94 ambiguous queries and 159 non-ambiguous queries. To be fair for comparison, we removed the queries without any clicked URLs from the dataset. Thus we get a subset of 235 queries, in which 90 queries are ambiguous. We call it Set-A in the paper. We use the dataset to evaluate query ambiguity models and features in Subsections 4.2∼4.5. For Set-A, the positive category is the ambiguous query category.

We also ask an annotator to label follow up queries as relevant/irrelevant to a query. Given a query, we first collect all the follow up queries from the log. Then the annotator judges whether each follow up query is relevant to the original query. Peer queries and unrelated queries, as shown in Table 1, are specified as irrelevant. Due to the limited labeling resource, we only randomly selected 50 queries from Set-A to annotate their follow up queries. This dataset forms Set-B. We use Set-B to evaluate the methods of identifying relevant follow up queries in Subsection 4.5. For Set-B, the positive category is relevant follow up query category.

We apply four standard metrics, i.e., Precision, Recall, F1 (the harmonic mean of Precision and Recall), and Accuracy, to evaluate the methods.

4.2 Feature Selection and Parameter Tuning

To perform solid comparison experiments, we select optimal features for different methods. As the features described in Subsection 3.4 may have redundant information, not all of them are beneficial in learning an ambiguous query model, given the limited number of training samples. Therefore, we conduct experiments

to select a minimal set of features that achieves the optimal result.

We apply a simple approach to select features. Given a method, we start with all the features. The feature set is evaluated by learning ambiguous query models based on the feature set. For all the learning experiments, we conduct ten-fold cross validation. The average accuracy is recorded as the initial best result. Next, we remove a feature and evaluate the remaining features. If the accuracy decreases, we will move the feature back to the minimal set because it does contribute to the best result. Otherwise, the feature will not be included in the optimal set. The procedure will continue until all features have been processed. Finally, we have a minimal set of features that is approximately optimal.

The selected features are shown in Table 2. For the click based method, in addition to the basic features, all the three matrices, i.e., $\boldsymbol{P}$, $\boldsymbol{S}$, and $\boldsymbol{G}$, contribute useful features to the best result. It indicates that the three matrices provide useful information from different perspectives. For the session based method, three distance formulas are used in calculating the features. The selected features show that the additional gains have been achieved by using different distance measures. For the combined log based method, the minimal query set is similar to the union of the features from the two individual methods. The difference is that No. 12 feature is added while No. 14 and No. 15 features are not selected. It indicates that the features from clicks provide some redundant information to the features from sessions. Such redundant information also influences the result of feature selection.

Table 2. Minimal Sets of Useful Features

No.	Feature	Click	Session	Log
1	TermNum	Y	Y	Y
2	AvgClkTimes	Y		Y
3	AvgMaxClkPos	Y		Y
4	P-Entropy	Y		Y
5	P-Diameter-euc	Y		Y
6	S-DMean-euc	Y		Y
7	S-Radius-euc	Y		Y
8	G-DMean-euc	Y		Y
9	T-Diameter-cos		Y	Y
10	T-DMean-euc		Y	Y
11	T-Radius-cos		Y	Y
12	T-RSD-euc			Y
13	T-RSD-jsd		Y	Y
14	T-Diameter-jsd		Y	
15	T-DMean-jsd		Y	

In addition, the variance, denoted by σ, of the RBF kernel is a sensitive parameter in SVM learning. We first set a default σ in feature selection experiments, and then tune σ on the minimal feature sets respectively. The optimal σ is 0.2 for the click based method, 1.25 for the session based method, and 1.25 for the combined log based method.

4.2 Query Ambiguity Models Experiment

We carried out experiments to compare five ambiguous query classification approaches on Set-A. We randomly select 90% of queries as training data and regard the others as testing data. Then we evaluate all methods and record the results. Such a procedure is repeated 50 times. The final evaluation results are averaged over the 50 trials.

Five approaches for comparison are: 1) the log based approach that combines both features from user clicks and query sessions, 2) the click based approach, 3) the session based approach, 4) the top retrieved document based approach that was proposed in [1], and 5) the combination of log based approach and document based approach. Table 3 shows the results. Please note: the performance of the document based approach is not the same as that in [1] because different datasets are used as described in Subsection 4.1. In addition, the last column of the table is whether the difference between this approach and the log based approach is statistically significant. "Y" means "yes" when the p-value is less than 0.05 in t-test, while "N" means "no" otherwise.

Table 3. Comparing Four Query Ambiguity Models on Set-A

Approach	Precision	Recall	F1	Accuracy	Significant
Log	**0.862**	**0.847**	**0.854**	**0.860**	–
Click	0.809	0.805	0.807	0.815	Y
Session	0.819	0.812	0.815	0.824	Y
Document	0.849	0.843	0.846	0.851	N
Log + Document	0.860	0.837	0.848	0.854	N

The combined log based approach significantly improves both the click based approach and the session based approach. This supports our conjecture: user clicks and query sessions do contain complementary information to each other in identifying ambiguous queries. For instance, the first search page may contain the documents that are relevant to the major meanings of an ambiguous query. Thus it is unnecessary for users to reformulate the original query. Instead, they click the relevant documents. In such a case, query sessions cannot provide information on query ambiguity, but user clicks can. In another example, the first search page is dominated by one meaning of an ambiguous query sometimes, and thus user clicks are focused. However, those users, who seek other meanings of the query, will re-submit queries to clarify their

needs. Here, the session data can provide more useful information than user clicks in judging whether the query is ambiguous. Therefore, when we combine the features from user clicks and query sessions, the accuracy increases dramatically to 86%.

As Table 3 shows, our log based approach performs slightly better than the document based approach proposed in [1], but the difference is not statistically significant. The comparison is done on the queries that have logs. Therefore, we can conclude that our proposed log based approach is comparable to the document based approach when query logs are available. When we combine the features from these two approaches, the accuracy is not further improved. Nevertheless, the document based approach is applicable for new queries. Thus, it is still meaningful to combine the document based approach with the log based approach to expand query coverage in real applications.

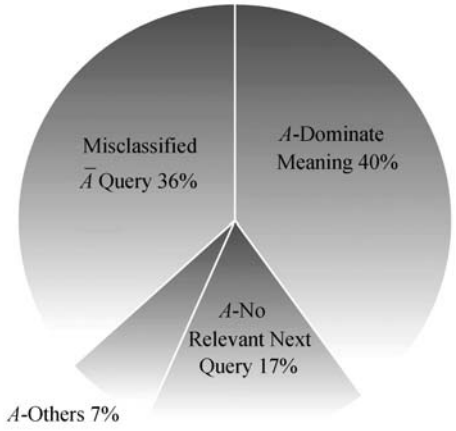

Fig.2. Analysis on misclassified queries.

We further investigated why some queries are wrongly classified by the log based approach. As Fig.2 shows, among misclassified queries, about 64% are ambiguous. By looking closely into the wrongly classified ambiguous queries, we found that the top reason is dominant intentions. By this we mean that for more than two thirds of misclassified A queries, one of their meanings is much more popular than any of the other meanings, so that the user logs are all about the dominant meaning, which makes the query look like a clear query. For example, the query "Levis" could be a place, people, or a motorcycle manufacturer, but search logs are dominated by Levi's Jeans. In addition, some ambiguous queries are misclassified because no relevant follow up queries are found in user session data. Perhaps only a few users have ever searched the query, or the user's need was fulfilled by the search results as he/she was seeking the most popular meaning. In summary, the two main causes of classification errors are dominant interpretation and data sparseness.

4.4 Feature Contribution Experiment

We conducted experiments to evaluate the contribution of individual features in the log based approach. First, we remove each feature from the minimal set of the best model, as shown in Table 2. Then, we re-train models based on the remaining features. Next, when evaluating the new models, we calculate the accuracy drop. Finally, we rank the features by accuracy drop and show the results in Fig.3. The bigger the drop the absence of a feature brings, the more the feature contributes.

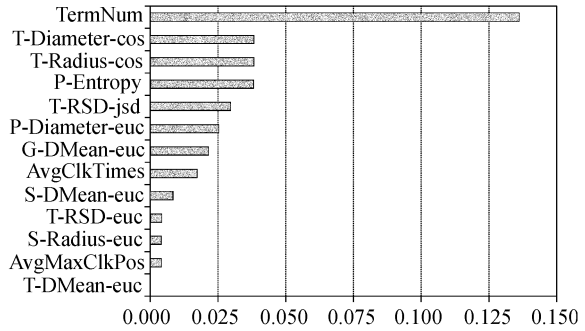

Fig.3. Accuracy drops if excluding a feature from the best minimal feature set.

The most significant feature is TermNum because the most ambiguous queries have only one keyword. This is consistent with the remarks in [1]. In their user study, about 90% of ambiguous queries are single-term queries. The second and third most significant features are T-Diameter-cos and T-Radius-cos. This confirms that the relevant queries mined from user sessions do provide complementary information on query vagueness. For an ambiguous query, the relevant follow up queries are diverse in topics. The fourth most significant feature is P-Entropy. It indicates that different user click patterns do exist if a query is ambiguous. In addition, we see that the different distance formulas are useful. If using the cosine distance only, we will lose the additional gains from the JSD distance and Euclid distance.

4.5 Relevant Query Identification Experiment

In the best log based approach, we identify relevant queries by matching a whole query term and detecting acronyms by a simple rule. In this experiment, we will compare this method with two variants: 1) loosing the keyword matching by allowing a partial match of a term, denoted as LM, and 2) skipping acronym detection, denoted as NA. We extract the same features

from the relevant follow up queries that are produced by the two alternative methods and learn ambiguous query models. The results are shown in Table 4.

Table 4. Comparing Related Query Identification Methods by Classifying Ambiguous Queries on Set-A

Approach	Precision	Recall	F1	Accuracy	Significant
Session_LM	0.803	0.802	0.802	0.810	N
Session_NA	0.818	0.811	0.814	0.822	N
Session	**0.819**	**0.812**	**0.815**	**0.824**	–
Log_LM	0.828	0.817	0.822	0.832	Y
Log_NA	0.854	0.843	0.848	0.857	N
Log	**0.862**	**0.847**	**0.848**	**0.860**	–

In terms of accuracy, we see that the performance goes down when we loose key term matching. The t-test results show that the drop is significant when we apply the log based approach. It indicates that condition 1 described in Subsection 3.2.2 is important in learning a query ambiguity model. If the follow up query is not required to share at least one exact same term with the key query, noisy queries will be brought in and will hurt the accuracy of classifiers. In contrast, condition 2 on identifying acronyms slightly influences the accuracy as the difference between the two methods is not significant.

We use Set-B for evaluating the effectiveness of related query extraction. Given a query q, the positive category consists of all the follow up queries that are relevant to the query q. For example, for the query "act", example positive queries are "act practice test", "act test dates", "tax act" and "acceptance and commitment therapy", whereas example negative queries are "facebook", "gmail", "aol", and "mapquest". Precision and recall are the main measures in the evaluation. We show the results in Table 5. The method proposed in Subsection 3.2.2 is denoted as "Final". Without acronyms detected, both precision and recall fall slightly, whereas loosing the keyword matching increases recall dramatically while lowering precision.

Table 5. Comparing Relevant Queries Identification Performance on Set-B

Method	Precision	Recall	F1
LM	0.739	**0.891**	**0.808**
NA	0.741	0.775	0.758
Final	**0.744**	0.785	0.764

We find that classification performance is correlated with the precision of relevant queries identification. Referring to Table 4 and Table 5, our proposed method achieved the highest precision and it also performed the best in learning experiments. In contrast, although loosing keyword matching improves recall a lot, the loss in precision results in the lowest performance in ambiguous query classification. Therefore, precision of relevant query identification plays a more important role than recall in classifying ambiguous queries.

5 Conclusion

This paper addresses the problem of learning an ambiguous query classifier by using search logs. We propose taking advantage of both click-through logs and query session logs to conquer the data sparseness of each. Four matrices, i.e., the user-document matrix, the document-category matrix, the user-category matrix, and the query-category matrix, are generated by mining the logs and categorizing the mined documents and queries. Based upon such log data, we extract some novel features that prove to be effective in identifying ambiguous queries. Combining click-through logs and query session logs significantly improves the effectiveness over clicks or sessions alone. The accuracy is comparable to the top returned document based approach proposed in previous work.

For future work, we are going to solve two problems: 1) how to discover different interpretations of an ambiguous query from logs; 2) how to leverage the clicked documents for different interpretations in evaluating search results diversity. Our eventual goal is to construct a large-scale test collection of ambiguous queries with low cost. Furthermore, solving these two research problems is also meaningful for developing new user interfaces and new rankings for ambiguous queries.

References

[1] Song R, Luo Z, Nie J Y, Yu Y, Hon H W. Identification of ambiguous queries in Web search. *Information Processing and Management*, 2008, 45(2): 216-229.

[2] Dou Z, Song R, Wen J R. A large-scale evaluation and analysis of personalized search strategies. In *Proc. the 16th International Conference on World Wide Web* (*WWW 2007*), Banff, Canada, May 8-12, 2007, pp.581-590.

[3] Sanderson M. Ambiguous queries: Test collections need more sense. In *Proc. the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2008*), Singapore, July 20-24, 2008, pp.499-506.

[4] Radlinski F, Dumais S. Improving personalized Web search using result diversification. In *Proc. the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2006*), Seattle, USA, Aug. 6-11, 2006, pp.691-692.

[5] Li Y, Zheng Z, Dai K. KDD CUP-2005 report: Facing a great challenge. *SIGKDD Explor. Newsl.*, 2005, 7(2): pp.91-99.

[6] Vapnik V N. Principles of Risk Minimization for Learning Theory. Advances in Neural Information Processing Systems 4, Morgan Kaufmann, 1992, pp.831-838.

[7] Mihalcea R, Pedersen T. Advances in word sense disambiguation. In *Tutorials at the 20th National Conference on Artificial Intelligence*, Pittsburgh, USA, July 9-13, 2005.

[8] Krovetz R, Croft B W. Lexical ambiguity and information retrieval. *ACM Trans. Inf. Syst.*, 1992, 10(2): 115-141.

[9] Voorhees E M. Using WordNet to disambiguate word senses for text retrieval. In *Proc. the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 1993*), Pittsburgh, USA, June 27-July 1, 1993, pp.171-180.

[10] Carbonell J, Goldstein J. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 1998*), Melbourne, Australia, Aug. 24-28, 1998, pp.335-336.

[11] Zhai C X, Cohen W W, Lafferty J. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In *Proc. the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2003*), Toronto, Canada, Jul. 28-Aug. 1, 2003, pp.10-17.

[12] Zhai C X, Lafferty J. A risk minimization framework for information retrieval. *Information Processing and Management*, 2006, 42(1): 31-55.

[13] Chen H, Karger D R. Less is more: Probabilistic models for retrieving fewer relevant documents. In *Proc. the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2006*), Seattle, USA, Aug. 6-11, 2006, pp.429-436.

[14] Agrawal R, Gollapudi S, Halverson A, Ieong S. Diversifying search results. In *Proc. the Second ACM International Conference on Web Search and Data Mining* (*WSDM 2009*), Barcelona, Spain, Feb. 9-12, 2009, pp.5-14.

[15] Lee U, Liu Z, Cho J. Automatic identification of user goals in Web search. In *Proc. the 14th International Conference on World Wide Web* (*WWW 2005*), Chiba, Japan, May 10-14, 2005, pp.391-400.

[16] Dai H (Kathy), Zhao L, Nie Z, Wen J R, Wang L, Li Y. Detecting online commercial intention (OCI). In *Proc. the 15th International Conference on World Wide Web* (*WWW 2006*), Edinburgh, UK, May 23-26, 2006, pp.829-837.

[17] Gravano L, Hatzivassiloglou V, Lichtenstein R. Categorizing web queries according to geographical locality. In *Proc. the Twelfth International Conference on Information and Knowledge Management* (*CIKM 2003*), New Orleans, USA, Nov. 2-8, 2003, pp.325-333.

[18] Platt J C. Fast Training of Support Vector Machines Using Sequential Minimal Optimization. Advanced in Kernel Methods: Support Vector Learning, MIT Press, 1998.

[19] Cao H, Jiang D, Pei J, He Q, Liao Z, Chen E, Li H. Context-aware query suggestion by mining click-through and session data. In *Proc. the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD 2008*), Las Vegas, USA, Aug. 24-27, 2008, pp.875-883.

[20] Shen D, Pan R, Sun J T, Pan J J, Wu K, Yin J, Yang Q. Q2C@UST: Our winning solution to query classification in KDDCUP 2005. *SIGKDD Explor. Newsl.*, 2005, 7(2): 100-110.

[21] Lin J. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 1991, 37(1): 145-151.

Ruihua Song is a researcher in Microsoft Research Asia. She received B.E. and M.E. degrees from Department of Computer Science and Technology, Tsinghua University. Her main research interests are Web information retrieval and Web information extraction.

Zhicheng Dou is an associate researcher in Microsoft Research Asia. He received the B.S. and Ph.D. degrees in computer science and technology from Nankai University in 2003 and 2008, respectively. His main research interests include Web information retrieval and data mining.

Hsiao-Wuen Hon is managing director of Microsoft Research Asia. As an IEEE fellow, Dr. Hon is an internationally recognized expert in speech technology. His recent research focuses on Web information retrieval and natural language processing.

Yong Yu is a professor in Computer Science Department of Shanghai Jiao Tong University. He got his Master's degree from East China Normal University. His research focuses on Web search and mining, semantic Web and peer-to-peer search.