Automatic Classification of Web Queries Using Very Large Unlabeled Query Logs

STEVEN M. BEITZEL and ERIC C. JENSEN
Illinois Institute of Technology
DAVID D. LEWIS
David D. Lewis Consulting
and
ABDUR CHOWDHURY and OPHIR FRIEDER
Illinois Institute of Technology

Accurate topical classification of user queries allows for increased effectiveness and efficiency in general-purpose Web search systems. Such classification becomes critical if the system must route queries to a subset of topic-specific and resource-constrained back-end databases. Successful query classification poses a challenging problem, as Web queries are short, thus providing few features. This feature sparseness, coupled with the constantly changing distribution and vocabulary of queries, hinders traditional text classification. We attack this problem by combining multiple classifiers, including exact lookup and partial matching in databases of manually classified frequent queries, linear models trained by supervised learning, and a novel approach based on mining selectional preferences from a large unlabeled query log. Our approach classifies queries without using external sources of information, such as online Web directories or the contents of retrieved pages, making it viable for use in demanding operational environments, such as large-scale Web search services. We evaluate our approach using a large sample of queries from an operational Web search engine and show that our combined method increases recall by nearly 40% over the best single method while maintaining adequate precision. Additionally, we compare our results to those from the 2005 KDD Cup and find that we perform competitively despite our operational restrictions. This suggests it is possible to topically classify a significant portion of the query stream without requiring external sources of information, allowing for deployment in operationally restricted environments.

Categories and Subject Descriptors: H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*

General Terms: Algorithms, Experimentation

Authors' current addresses: S. M. Beitzel, Telcordia Technologies, One Telcordia Drive, RRC-1K329, Piscataway, NJ 08854; email: steve@research.telcordia.com; E. C. Jensen, A. Chowdhury, and O. Frieder, Department of Computer Science, Illinois Institute of Technology, 10 West 31st Street, Room 236, Chicago, IL 60616-3793; email: {ej,abdur,ophir}@ir.iit.edu; D. D. Lewis, David D. Lewis Consulting, 858 W. Armitage Ave., # 29, Chicago, IL 60614; email: davelewis@daviddlewis.com.

Additional Key Words and Phrases: Lexical semantics, perceptron, semisupervised learning, text categorization

ACM Reference Format:

S. M. Beitzel et al. 2007. Automatic classification of Web queries using very large unlabeled query logs. ACM Trans. Inform. Syst. 25, 2, Article 9 (April 2007), 29 pages. DOI = 10.1145/1229179.1229183 http://doi.acm.org/10.1145/1229179.1229183

1. INTRODUCTION

Understanding the topical sense of user queries is a problem at the heart of Web search. Successfully mapping incoming user queries to topical categories, particularly those for which the search engine has domain-specific knowledge, can bring improvements in both the efficiency and the effectiveness of Web search. Much of the potential for these improvements exists because many of today's search engines, both for the Web and for enterprises, often incorporate the use of topic-specific back-end databases when performing general Web search. That is, in addition to traditional document retrieval from general indices, they attempt to automatically route incoming queries to an appropriate subset of specialized back-end databases and return a merged listing of results, often giving preference to the topic-specific results. This strategy is similar to that used by metasearch engines on the Web [Glover et al. 1999]. The hope is that the system will be able to identify the set of back-end databases that are most appropriate to the query, allowing for more efficient and effective query processing. Correct routing decisions can result in reduced computational and financial costs for the search service, since it is impractical to send every query to every backend-database in the (possibly large) set. If a search service has a strict operational need for low response time, erroneous routing decisions could force much larger scaling than is necessary [Chowdhury and Pass 2003]. An accurate classification of a query to topic(s) of interest can assist in making correct routing decisions, thereby reducing this potential cost.

Additionally, topical query classifications, sometimes coupled with source-specific query rewriting, have broad potential for improving the effectiveness of the result set. In addition to results that are specifically targeted for the identified topical domain, the presentation of results may be altered accordingly. For example, a geographic query might result in the presentation of a map; a shopping query could result in an extracted list of comparison prices, etc. Topical classifications can also assist the search service in delivering auxiliary information to the user. Advertisements, for instance, can be tailored based on category rather than query keywords. Finally, categorization of the query stream as a whole allows tracking of trends over time. A search service can use these data as a means of identifying areas that may require additional resources (e.g., additional hardware, new back-end databases). To fully realize these gains, a classification system is required that can automatically classify a large portion of the query stream with a reasonable degree of accuracy. In the case of large-scale Web search, this task is particularly challenging:

—The Web and its users are highly dynamic. The available content on the Web changes by as much as 8% per week, with fluctuations in servers and pages

Table I. Query Stream Breakdown

Autos	3.46%	Personal Finance	1.63%
Business	6.07%	Places	6.13%
Computing	5.38%	Porn	7.19%
Entertainment	12.60%	Research	6.77%
Games	2.38%	Shopping	10.21%
Health	5.99%	Sports	3.30%
Holidays	1.63%	Travel	3.09%
Home & Garden	3.82%	URL	6.78%
News & Society	5.85%	Misspellings	6.53%
Orgs.&Insts.	4.46%	Other	15.69%

[Cho et al. 1998; Lawrence and Giles 1998; Ntoulas et al. 2004]. Additionally, the population of users, their interests, and their model of how a search works, is simultaneously varying on time scales from minutes to years [Beitzel et al. 2004].

—Traffic at major search engines reaches hundreds of millions of queries per day [Sullivan 2006] and is highly diverse [Beitzel et al. 2004; Beitzel et al. 2006; Spink and Jansen 2004], requiring a large sample of queries to adequately represent the population.

—It is difficult to determine the user's desired task and information need from Web queries, which are typically very short [Jansen et al. 2000]. This complicates manual categorization of queries (for study, or to provide training data), and poses great challenges for automated query classification [Gravano et al. 2003; Kang and Kim 2003].

The key contribution of this work, introduced in Beitzel et al. [2005a, 2005b], is the use a large Web search engine query log as a source of unlabeled data to aid in automatic classification. This approach allows a classification system to take into account the language and vocabulary of a vast number of real users' queries, and to use this information to classify portions of the query stream that are unreachable using other techniques. Additionally, no external resources, such as the contents of Web directories (e.g., The Open Directory Project) or retrieved documents for a query are used with this approach. This allows it to be deployed under circumstances of operational restriction where the cost of utilizing such data is prohibitively high for large volumes of queries. Furthermore, this approach is inherently robust to changes in the query stream because it draws information from the very thing that is changing most: the users' queries themselves.

In Table I, we present a topical breakdown of a manually classified sample of approximately 600 queries taken from one week's worth of queries posed to AOL™ search. This search service has a large number of backend databases with topic-specific information. Accurate classification of queries into categories like those listed in Table I (as well as more specific categories further down the hierarchy) would improve effectiveness of search results, reduce the computational and financial costs of system operation, and provide new opportunities for revenue.

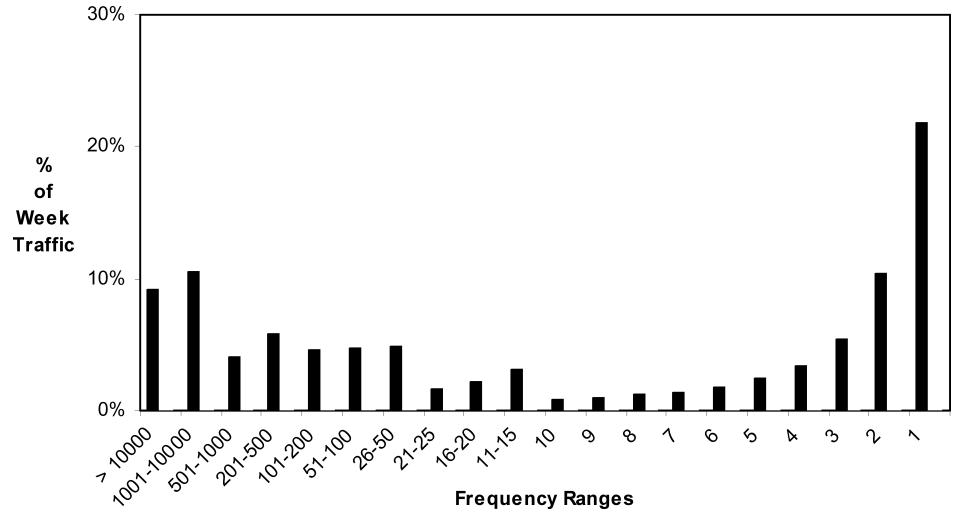

Fig. 1. Frequency distribution of 1 week of queries.

To realize these improvements, a query classification system must classify a large portion of the query population. Previously proposed classification techniques do not, however, seem adequate to this task. Manual classification is expensive, and so can be applied to only a small portion of the query population. Attempts to target manual classification to the most frequent and/or valuable queries are undermined by dynamic nature of the query stream. Generalizing from manually labeled queries by supervised machine learning improves coverage, but cannot make up for inadequate coverage of vocabulary in the training data and a worsening of this problem as query vocabulary changes over time.

Query stream statistics provide insight into the limitations of manual and supervised approaches. In Figure 1, we show the frequency distribution of all queries in a week's worth of traffic from AOL™ search. The y-axis represents the proportion of all queries that have individual frequencies in the range of the buckets on the x-axis. For example, Figure 1 shows that more than 20% of all queries observed in a week occur only once. This distribution shows that the Web query stream is tail-heavy, with nearly 50% of queries seen occurring five or fewer times. In contrast, only a relatively small portion of the query stream ($\sim$10%) occurred more than 10,000 times that week. Manual classification efforts are likely to be biased toward the most popular queries, as they are typically the most familiar to human assessors and are most often encountered. Additionally, classification methods that use frequency in their calculations, including supervised machine learning algorithms and retrieved-document classifiers, will have a difficult time attaining sufficient recall because there is little frequency information available for the majority of queries. To mitigate this recall problem, we require a classification system that is able to leverage the terms and phrases comprising a sufficiently large population of Web queries, popular and rare alike.

We develop a rule-based automatic classifier produced using a novel application of the computational linguistics notion of *selectional preferences*. We use this technique to perform a cooccurrence analysis on a large unlabeled query log containing hundreds of millions of queries. This analysis reveals strong preferences for query terms to be associated with particular categories, and these preferences are used to develop rules for classification. We then combine these selectional preference rules with exact matching against a large set of manually classified queries and a weighted automatic classifier trained using supervised learning to achieve the most effective classification possible. Our combined approach bridges the gap between several related areas of information science, applying elements of machine learning and computational linguistics to the problem of automatic query classification. Manual classification provides a high-precision element while supervised machine learning leverages these manual classifications for additional benefit. Selectional preferences utilize massive unlabeled query logs to suggest classification rules that are able to change along with the query stream, adapting over time as users' interests (and, by extension, their queries) change.

Our combined approach leverages the strengths of each individual technique to achieve the most effective classification possible, outperforming each individual method and achieving a high level of classification recall. This mitigates the recall problem due to feature sparseness encountered in prior query classification studies, and allows us to classify a much larger portion of the query stream than any single technique alone while remaining robust to changes in the query stream over time. Finally, we are able to perform these classifications without the assistance of external sources of data, which is a significant departure from previous work in automatic topical query classification.

In Section 2, we give an overview of prior efforts in classification. In Section 3, we describe the individual methodologies of each classification technique in our approach. We discuss our evaluation methods and give details on our datasets in Section 4, and present our results and analysis and discussion in Section 5. Finally, in Section 6 we state some conclusions and give directions for future work.

2. PRIOR WORK

Classifying texts and understanding queries are both fundamental concerns in information retrieval, and so we can mention only a fraction of the potentially relevant work. In particular, while work on classifying documents is relevant to classifying queries, we focus on query-oriented work as being most relevant here. Query classification makes very different demands on a classification system than does document classification. Queries are very short (the studies discussed below find average Web query lengths between 2.0 and 2.6 terms), and thus have few features for a classifier to work with. Further, unlike a full text document, a query is just a stand-in for the thing we would really like to classify: the user's interest. Sebastiani [2002] has surveyed recent work on document classification.

More narrowly, our particular interest is in queries to Web search engines. Past research can generally be divided (with some overlap) into work

employing either manual or automatic methods of query classification. Manual classification (discussed in Section 2.1), as stated previously, typically involves using human editors or simple heuristics to create a set of static classification decisions. Automatic classification (discussed in Section 2.2) usually refers to using more complex algorithmic methods (supervised machine learning is common) to cluster similar queries or perform direct task or topic-based classification. Until recently there was little published research that specifically addressed the problem of automatic topical classification of Web queries, creating a key gap in the existing literature. The 2005 KDD Cup (discussed in Section 2.2.1) partially addressed this, as it focused specifically on automatic topical classification of Web queries. However, top performers in the competition all relied on external sources of information that are not practical for use in real-time classification systems that process large volumes of queries. The primary focus of this study is to discuss and compare our classification techniques, which require only a query log and a small seed of manual classifications, to existing automatic classification techniques such as those used in the 2005 KDD Cup.

2.1 Manual Query Classification

Several studies have attempted to classify user queries in terms of the informational actions of users. In early work, Broder [2002] manually classified a small set of queries into transactional, navigational, and informational tasks using a pop-up survey of AltaVista users, and manual inspection.

One set of studies [Jansen et al. 2005; Spink et al. 2001; Spink et al. 2002a, 2002b] looked at query logs from the Excite™ search engine from the years 1997, 1999, and 2001, and the FAST™ search engine from 2001. Sets of about 2500 queries from each log were manually classified into 11 overlapping topic categories developed by an information science class. (Our own category scheme bears some similarities to this one, but is slightly finer-grained.) Query frequencies were found to vary year-to-year, and between a mostly U.S. (Excite) versus mostly European (FAST) user population. Even with this coarse classification, no single query type accounted for more than 25% of traffic, emphasizing the diversity of user needs.

There have also been attempts to study query logs using automated classification techniques. Beitzel et al. [2004] used exact lookup of queries from AOL™ 's Web search engine in a large database of semiautomatically categorized queries for the purpose of analyzing query behavior over time. They used a set of 16 categories and found the most frequent category (Other) accounted for only 16% of queries. They also found that some categories were much more variable hour-to-hour in frequency of queries and composition of queries within the category. We hypothesized that these types of categories would give us the most trouble in automated categorization. As with other studies, this one showed a short mean query length (2.2 words).

2.2 Automatic Query Classification

Other work on query classification is focused less on understanding large-scale properties of query logs and more on improving system effectiveness on each

individual query. Much of this work is proprietary and unpublished, but there is a growing base of academic literature.

A number of researchers have automatically classified queries into Broder's [2002] informational/navigational/transactional taxonomy (or very similar taxonomies) and applied different query rewriting and weighting strategies for each category. Kang and Kim [2003], for example, used query term distribution, mutual information (cooccurrence), usage rate in anchor text, part-of-speech info, and combinations of the above to automatically classify queries into Broder's [2002] taxonomy. They used this classification to decide which query processing algorithms to use for retrieval, achieving at best modest improvements in retrieval effectiveness. Their study exhibited a problem common to all query classification studies: the difficulty of achieving accurate classification, and in particular high recall, for the inevitably short queries.

Kang and Kim [2003] made use of data from Web search evaluations at the TREC conferences. A number of groups participating in Web search evaluations at TREC have attempted classification with respect to similar categories, but the benefits from these attempts have not been clear [Craswell et al. 2003] and an effective alternative seems to be to design search approaches that do well on all question types without explicit classification [Craswell and Hawking 2004]. On the other hand, high-scoring participants in the TREC QA (question answering) [Voorhees 2004] track appear to be making increasing use of query classification, in the sense of attempting to determine, by linguistic or other means, the class of answer a question is seeking. It should be noted that QA track questions are not typical Web queries.

Gravano et al. [2003] used machine learning techniques in an automatic classifier to categorize queries by geographical locality. They sampled small training, tuning, and testing sets of a few hundred queries each from a large 2.4 million-query Excite™ log from December 1999. Their analysis revealed that data sparseness was a problem for their classifier, noting in particular that most queries were very short, and, specific to their geographical task, queries rarely contained key trigger words likely to identify whether or not a query was "local" or "global" in geographical scope. They concluded that successful automatic classification would have to draw on auxiliary sources of information, such as user feedback or supplemental databases, to compensate for the small number of features in a single query.

An alternative to classifying queries into manually defined categories is to allow classes to emerge from a query set itself, through clustering or other unsupervised learning methods. There is a long history of such approaches in the field of information retrieval (IR) [Grossman and Frieder 2004; Salton et al. 1975], with attempts in the past often stymied by the small query logs available. In clustering Web queries, the problem is no longer lack of queries, but lack of features in any individual query. This is arguably an even greater problem for unsupervised than supervised approaches, which can at least latch on to individual features correlated with a class.

Some query clustering methods have attacked this problem by clustering "session data," containing multiple queries and click-through information from a single user interaction. Beeferman and Berger [2000] mined a log of 500,000

click-through records from the Lycos™ search engine and used a bipartite-graph algorithm to discover latent relationships between queries based on common click-through documents linking them together. Unfortunately, only anecdotal results were reported in the study. A similar approach was used by Wen et al. [2001a, 2001b, 2002], who also took into account terms from result documents that a set of queries has in common. They concluded that the use of query keywords together with session data is the most effective method of performing query clustering. However, their test collection was an encyclopedia, so the applicability of their results to general Web search is limited.

2.2.1 *KDD Cup 2005.* The ACM Conference on Knowledge and Data Discovery (KDD) holds an annual competition known as the *KDD Cup*. The task varies from year to year, usually focusing on an area in the information sciences. For 2005, the task was topical query categorization. The dataset consisted of 800,000 Web queries, and 67 possible categories, with each category pertaining to a specific topic ("Sports-Baseball" was one example of a KDD cup category). Each participant was to classify all queries into as many as five categories. An evaluation set was created by having three human assessors independently judge 800 queries that were randomly selected from the sample of 800,000. On average, the assessors assigned each query to 3.2 categories for their judgments. Once complete, these evaluations were used to calculate the classification precision and F_1 score for each submission (F_1 is defined as the harmonic mean of precision and recall [van Rijsbergen 1979]). Equation (1) gives definitions for each of these classification evaluation metrics. Each submission was evaluated by finding the mean F_1 and precision scores across individual assessors. In all, there were 37 classification runs submitted by 32 individual teams.

$$\begin{aligned} P &= \frac{\#TruePos}{\#TruePos + \#FalsePos}, \\ R &= \frac{\#TruePos}{\#TruePos + \#FalseNeg}, \\ F_1 &= \frac{2PR}{P + R}, \end{aligned} \tag{1}$$

where #*TruePos* is the number of *true positives*: the number of queries which belonged to the class, and which the classifier assigned to the class; #*FalsePos* is the number of *false positives*: the number of queries which did not belong to the class, but which the classifier assigned to the class; and #*FalseNeg* is the number of *false negatives*: the number of queries which belonged to the class, but which the classifier did not assign to the class.

Participants were not restricted in regard to the use of particular resources to aid in making classification decisions. As a result, several runs made use of various forms of external information.

Shen et al. [2005] used an ensemble approach of several different classification techniques to create the winning submission for the 2005 KDD Cup. They built synonym-based classifiers to map the category hierarchies used by search engines to the one employed at the KDD Cup. Specifically, this mapping function

was constructed by matching keywords between each category hierarchy. They extened the keyword matching to include various grammatical forms, and also via WordNet.[1] Three synonym-based classifiers were built, using the category hierarchies from Google™, Looksmart™, and an internal search engine based on Lemur,[2] searching a crawl of the ODP[3] hierarchy. They also built a statistical classifier using SVM_light[4] [Joachims 1999]. They collected training data by using the mappings found for the synonym classifiers to find pages in the relevant ODP categories for each query. The terms from the pages, their snippets, titles, and category names were stemmed and processed for the removal of stopwords and used to train the SVM classifier. They also used two ensemble classifiers, one using the 111-query training set to learn weights for each component classifier, and one giving equal weight to each component classifier. This approach resulted in an F_1 score that outperformed all other participants by nearly 10%.

Kardkovacs et al. [2005] proposed an approach called the *Ferrety Algorithm* for their participation. Their approach was similar to that of the winning team, employing the Looksmart and Zeal search engines to build their taxonomy mapping, enriching the mapping process with stemming, stopword removal, and the application of WordNet. They used these data to train their own learning software and make classification decisions, and they experimented with several different methods of feature selection, achieving good results on the KDD Cup data.

Vogel et al. [2005] contributed the runner-up submission for classification performance (F_1). Like the other top performers, they also relied on a Web directory to drive part of their classification. They built their category mapping by searching ODP with Google™ , and collecting the categories of the top 100 retrieved documents. They also made use of spelling corrections and alternate queries suggested automatically by Google™, which helped mitigate some of the noisy queries in the KDD Cup query set. They then manually inspected all of the categories returned by Google, and mapped each directory node to up to three KDD Cup categories. Finally, they combined all available information and generate probability scores for each category via a logistic regression. Weights were determined through an iterative procedure and covered several different facets, including the number of result documents for a category, category depth, redundancy of mapping, and others. They used hill-climbing to find optimal thresholds for maximizing F_1 and precision separately, submitting a run for F_1 that was good for second place overall.

At this time, no formal proceedings for the 2005 KDD Cup have been published; however, the datasets and a summary of results as well as a PowerPoint™ presentation with an overview of the task and some of the top teams' techniques are all available from the competition's Web site.[5] Some prior

[1] http://wordnet.princeton.edu.
[2] http://www.lemurproject.org.
[3] http://www.dmoz.com.
[4] http://svmlight.joachims.org.
[5] http://www.acm.org/sigs/sigkdd/kdd2005/kddcup.html.

studies that are similar to common approaches used by KDD Cup participants are Bot et al. [2005], Das-Neves et al. [2005], and Kowalczyk et al. [2004].

Although the participants were able to achieve encouraging results, with the median F_1 score at 0.23 (max. 0.44) and a median precision of 0.24 (max. 0.75), the top-teams took advantage of techniques that are impractical in large-scale operational environments. For example, running the query, retrieving the top ranked documents, classifying those documents, and using those classification decisions to classify the query is computationally prohibitive for a Web search engine, where the query volume can reach hundreds of millions per day [Sullivan 2006]. To that end, our approach must be viewed from a slightly different angle, as we are imposing operational restrictions on what kinds of information can be used by our classifier, in an effort to address the task of automatic topical query classification as it appears to search practitioners in the real world. To the best of our knowledge, no participating team leveraged unlabeled data similar to search engine query logs in their KDD Cup experiments. Furthermore, combining several disparate techniques with complementary strengths and weaknesses allows our approach to remain robust and effective even when confined to real-world operational requirements. We evaluate our system on the KDD Cup dataset and discuss our performance in Section 5.2.

3. RESEARCH MOTIVATIONS AND CLASSIFICATION APPROACHES

We explore three major research questions in this study:

(1) Is it possible to design a viable automatic query classification technique that utilizes search engine query logs?

As discussed in Section 1, search engine query logs describe the fundamental language of Web search. Our motivation here is to explore whether we can leverage those logs into a system capable of making topical query classifications.

(2) Can we combine several techniques to improve classification recall?

The "recall problem" that is fundamental to applying traditional text classification methods to the topical classification of Web queries is paramount here. We explore whether several different classification techniques can be successfully combined to mitigate the recall problem.

(3) How does an operationally restricted classification approach compare to those which use external resources?

As mentioned in Section 2.2.1, the most effective current techniques for automatic topical classification of Web queries rely on external sources of information such as online directories and the contents of documents retrieved for a query. We propose the development of a classification system that is not allowed to use external sources of data due to operational restrictions. To that end, it is interesting to compare and contrast the performance of this approach with existing high-performance approaches such as those used in the 2005 KDD Cup.

We describe the manual and automatic classification techniques that are used in our approach and explain our motivations for using each technique.

We also introduce a new rule-based automatic classification technique based on mining selectional preferences from a large unlabeled query log. Combining these approaches allows us to cover a large portion of the query stream with adequate precision, while not requiring external resources.

3.1 Exact-Matching Using Labeled Data

The simplest approach to query classification is looking the query up in a database of manually classified queries, that is, rote learning [Mitchell 1997]. This is not quite as crazy as it sounds. At any given time, certain queries (the reader can anticipate some of these) are much more popular than others. By combining manual classification of these queries with large authority files of proper nouns in certain categories (personal names, products, geographic locations, etc.), nontrivial coverage of the query stream can be achieved.

We explored this technique by using 18 lists of categorized queries produced in the above fashion by a team of AOL™ editors. We examined the coverage of these categories and found that even after considerable time and development effort, they only represented ∼12% of the general query stream. In addition to poor coverage, a key weakness in this approach is that the query stream is in a constant state of flux [Beitzel et al. 2004]. Any manual classifications based on popular elements in a constantly changing query stream will become ineffective over time. Additionally, manual classification is very expensive. It is infeasible to label enough queries, and to repeat this process often enough, to power an exact match classification system that covers a sufficient amount of the query stream. Finally, there are several factors that contribute to precision being lower than expected. These categories were developed for operational use, and their definitions have evolved over time leading to ambiguity and mismatches, which are potentially exacerbated when authority files of proper nouns are added to particular categories. Also, test queries used for evaluation are likely to be judged by a group of assessors different from the editors who created these categories, leading to assessor disagreement, which can further dampen precision.

3.1.1 *n-Gram Matching Using Labeled Data.* A major problem with exact matching, particularly against databases of names in some canonical form, is that queries are structurally diverse (see Beitzel et al. [2004] and the articles referenced therein; also Eastman and Jansen [2003]). Punctuation and search operators may be misused or misparsed, and words added or deleted, in a fashion that causes exact matching against even the correct entity to fail. To mitigate this, we explored a variety of relaxed lookup methods using n-gramming (partitioning the query into a set of subsequences that are n units long, where units may be words or characters). We experimented with both word-based and character-based n-grams, and found that the best approach is to use inclusive word-based n-grams, splitting each query into all of its subsequences between one and four words long. We then considered the query to match a manually classified database entry (and get that entry's category) if one of the query's n-grams exactly matched the entry. For example, if we encountered the query "yahoo mail," it would be broken up into the word-based n-grams "yahoo" and

"mail," and would match a category entry of "yahoo." Details on these experiments are given in Section 5. While n-gram matching substantially increases recall over exact matching (and unsurprisingly lowers precision), any form of rote learning is strongly limited by an inability to cover the long tail of low-frequency queries.

3.2 Supervised Machine Learning

A natural next step is to use supervised machine learning to generalize from the manually classified queries, not just match against them. The idea is that a classifier can abstract from the particular combinations of features seen in stored database queries and recognize new combinations of those features as also indicating category members. A classifier that combines many features usually also allows more flexible trading off of false positives and false negatives. A challenge for this approach is that Web queries are short, averaging between two and three terms per query [Beitzel et al. 2004, 2006; Jansen et al. 2000]. So even if the classifier successfully combines and balances a large collection of features, its decision for any particular query is inevitably based on that query's small number of nonnull feature values.

We trained linear classifiers using the Perceptron with Margins algorithm [Krauth and Mezard 1987], which has been shown to be competitive with state-of-the-art algorithms such as support vector machines in text categorization, and is very computationally efficient [Li et al. 2002]. Our implementation normalizes training and test feature vectors to unit length (Euclidean normalization), which we have found to increase classification accuracy on other datasets. We trained one perceptron classifier for each category, using all manually classified queries assigned to that category as positive examples, and all manually classified queries not assigned to that category as negative examples. We tested combinations of word, word n-gram, and character n-gram features for representing queries, and found on validation data that using all and only complete query words as binary features was most effective.

Figure 2 shows DET (Detection Error Tradeoff—see Manmatha et al. [2002] and Martin et al. [1997] for details) curves for perceptron classifiers for five of our categories (test set details are given in Section 4.1). Each point on the curve for a category corresponds to a particular numeric value of the threshold for the perceptron classifier. The abscissa plots the test set false-positive rate at that point, with the test set false-negative rate on the ordinate. While Figure 2 shows the perceptron classifier does generalize beyond the coverage of the exact match system, its coverage is still inadequate. There is also less ability to trade off false positives and false negatives than a linear classifier usually allows. The long flat regions of the graph (circled in Figure 2) correspond to a pair of threshold values between which no test set example's score falls. In other words, much of the useful range in which we would like to be able to trade off false positives and false negatives is unavailable to the Perceptron classifier. For the best-performing category (Porn), reducing the false-negative rate to 10% results in a false-positive rate of almost 50%.

The fundamental problem is that queries are both short and diverse in their vocabulary. The database of manually classified queries, partly due its manner

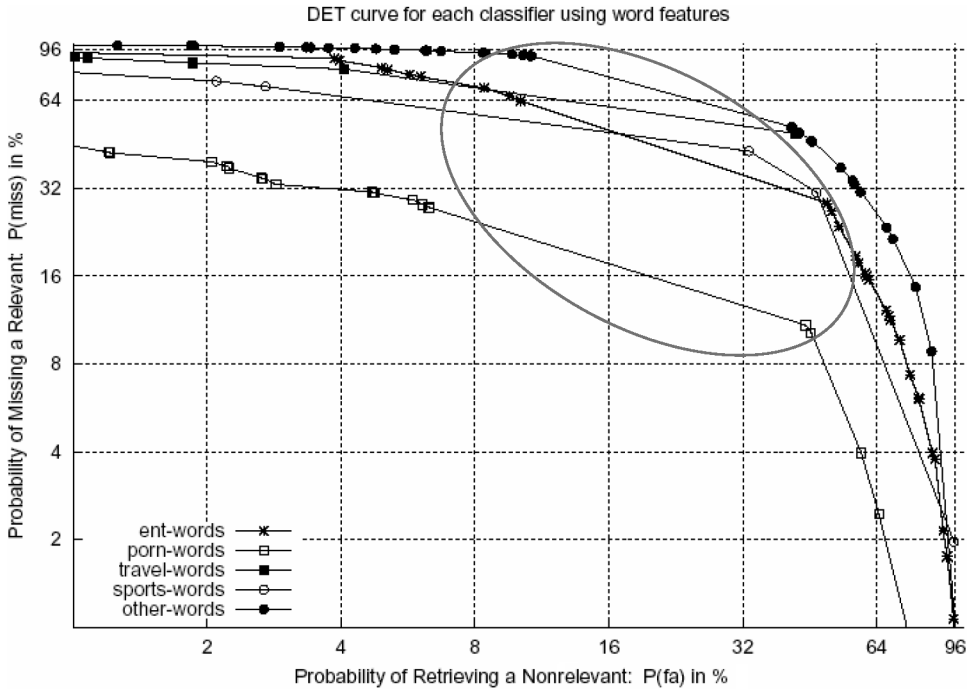

Fig. 2. DET curve for perceptron learner on five example categories.

of construction and partly due to ongoing changes in the query stream, does not systematically cover the space of queries. Many important predictor features are inevitably missing or underrepresented, and no process that uses the manually classified queries as its only evidence can overcome this.

3.3 Selectional Preferences

Our supervised learning approach treated classification of queries in the same fashion it would treat classification of documents. It therefore suffered from the fact that queries, with their short length, provide many fewer features than documents. We wondered, however, if this weakness could be turned into a strength. Because a query is so short, the fact that it has been manually classified into a particular category seemed to us to convey more information about the meaning of its component words than would the manual classification of a 1000 word document. Further, the short length and simple structure of queries suggested that knowledge about one word even in an unclassified query might let us infer knowledge about other words in that query. These insights suggested we might profitably view classifying a query as more like interpreting the meaning of a linguistic structure than classifying a document.

Computational linguists have studied many approaches to associating meanings with linguistic entities. The technique we explored is based on *selectional preferences* [Manning and Schutze 1999]: the tendency for words to prefer that their syntactic arguments belong to particular semantic classes. For instance,

the verb *eat* requires (except in metaphorical uses) that its direct object be something edible. Learning selectional preferences was an appealing approach to us, since the computations are simple, efficient, and can leverage large amounts of unlabeled data. We describe below how selectional preferences are typically approached in computational linguistics, and then how we adapted this technique to query classification.

3.3.1 *Selectional Preferences in Computational Linguistics.* A selectional preference study begins by parsing a corpus to produce pairs, (x, y), of words that occur in particular syntactic relationships. The verb-object relationship is most widely studied, but others such as adjective-noun have been used [McCarthy and Carroll 2003]. The interest is in finding how x, the context, constrains the meaning of y, its argument.

Resnik [1993] presented an influential information theoretic approach to selectional preference. He defined the *selectional preference strength* $S(x)$ of a word x, as in Equation (2), where u ranges over a set U of semantic classes. $P(U)$ is the probability distribution of semantic classes taken on by words in a particular syntactic position, while *P(U|x)* is that distribution for cases where a contextual syntactic position is occupied by x. S(x) is simply the KL divergence [Cover and Thomas 1991] of *P(U|x)* from $P(U)$.

$$\begin{aligned} \mathrm{S(x)} &= \mathrm{D(P(U|x)||P(U))} \\ &= \sum_{\mathrm{u}} \mathrm{P(u|x)log_2}\left(\frac{\mathrm{P(u|x)}}{\mathrm{P(u)}}\right). \end{aligned} \tag{2}$$

Ideally we would estimate the probabilities in Equation (2) from a set of (x, y) pairs where each y has been tagged with its semantic class, perhaps produced by parsing a semantically tagged corpus. The maximum likelihood estimates (MLEs) would be

$$\begin{aligned} \ddot{P}(u) &= \frac{n_u}{N}, \\ \ddot{P}(u|x) &= \frac{n_{xu}/N}{n_x/N} = \frac{n_{xu}}{n_x}, \end{aligned}$$

where N is the total number of pairs, n_u is the number of pairs with a word of class u in the syntactic position of interest, $n_x$is the number of pairs with x providing a context for that position, and n_{xu} is the number of pairs where both are true. Resnik [1993], Appendix A, discussed other estimates but said they give similar results to the MLE.

Large semantically tagged corpora are rare, however. A more common approach is to use unlabeled data, plus a thesaurus specifying which semantic classes each y can belong to. If a given y has only one class u it can belong to, we can replace each pair (x, y) with the pair (x, u). Some y's will not appear in the thesaurus, and the corresponding (x, y)'s are usually discarded. Conversely, some y's will be ambiguous (belong to multiple semantic classes) and so must be handled specially. Resnik [1993] treated each occurrence of such a

y as contributing fractionally to the count of each of its word senses, so that

$$n_u = \sum_{y \in W_u} \frac{n_y}{|U_y|},$$
$$n_{xu} = \sum_{y \in W_u} \frac{n_{xy}}{|U_y|},$$

where W_u is the set of words which have $u$as one of their classes, and U_y is the set of classes to which word y belongs. So, for instance, if a word y belongs to two classes, an occurrence of the word contributes a count of $\frac{1}{2}$ to each of the two classes.

3.3.2 *Selectional Preferences in Query Logs.* Selectional preferences can be used for disambiguation and semantic interpretation. If x strongly favors y's that belong to class u, then u is a good prediction for the class of an ambiguous, or previously unknown, y in that context. Indeed, many studies have evaluated the quality of learned selectional preferences by measuring the accuracy with which they can be used for disambiguation [Light and Greiff 2002].

To take advantage of this disambiguation effect to classify queries, we use a large log of unlabeled queries and do the following:

(1) Convert queries in an unlabeled query log to a set of head-tail (x, y) pairs.
(2) Convert each (x, y) pair to one or more weighted *forward*, that is, (x, u), pairs and *backward*, that is, (u, y) pairs, where u represents a category. Forward pairs are produced only when we have one or more u's associated with y, and backward pairs are produced only when we have one or more u's associated with x.
(3) Mine the weighted pairs to find lexemes that prefer to be followed or preceded by lexemes in certain categories (preferences).
(4) Use the mined preferences to assign test queries to semantic classes

Step 1 is straightforward for queries, vw, of length 2. We have only two possible "syntactic" relationships: the first token providing context for the second, or the second providing context for the first. These produce the *forward* pair *(_v, w)* and the *backward* pair *(w_, v),* where the underscore indicates matching at the front or the back of the query, respectively. We keep pairs of the two types separate, and call selectional preferences mined from (v, w) pairs *forward preferences*, and those from (w, v) pairs *backward preferences*. It is clear that this technique is not applicable to single-term queries, as there is nothing present to provide context for a single term (it should be noted that approximately 15% of all queries in our query log are composed of a single term).

If all two token queries were simple noun phrases (a modifier followed by a noun), then forward preferences would capture the degree to which a particular modifier (noun or adjective) constrained the semantics of the head noun. Backward preferences would capture the reverse. In practice, two token queries can arise from a variety of other syntactic sources: verb-object pairs, single words spelled with two tokens, noncompositional compounds, proper names, etc. A user may also intend the query as a pair of single words in no particular

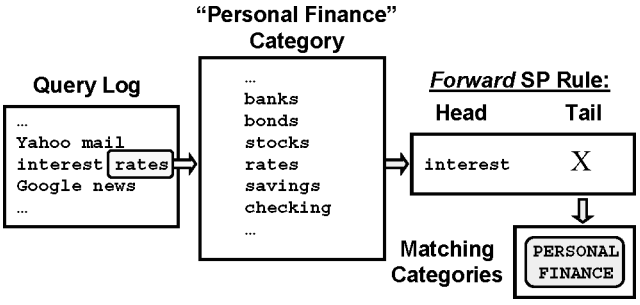

Fig. 3. Selectional preference rule generation—Step 2.

syntactic relation. Typographical errors and other anomalies are also possible. Thus our forward and backward relations inevitably have a murky interpretation.

Longer queries have more structural possibilities. Rather than attempting to parse them, we derived from a query $abc \ldots uvw$ all pairs corresponding to binary segmentations, that is,

$$(a, bc \ldots w), (ab, c \ldots vw), \ldots (abc \ldots v, w)$$

and

$$(bc \ldots w, a), (c \ldots w, ab), \ldots (w, abc \ldots v).$$

For any given query, most of these pairs get screened out in Step 2.

In Step 2, we replace each pair (x, y) with one or more pairs (x, u), where u is a thesaurus class. Pairs where y is not present in the thesaurus are discarded, and pairs where y is ambiguous yield multiple fractionally weighted pairs, as discussed in the previous section. Our "thesaurus" is simply our database of manually classified queries, with each query interpreted as a single (often multitoken) lexical item. The possible semantic classes for a lexical item are simply the set of categories it appears under as a query. Step 2 is illustrated in Figure 3.

In Step 3, we compute $S(x)$ for each x, as well as the MLE of $P(u|x)$ for each (x, u) pair seen in the data. We then screen out pairs where $S(x) < 0.5$, a relatively low threshold on selectional preference strength determined by initial tests on our validation set. From each remaining pair we form a rule $[x \rightarrow u{:}P(u|x)]$, which is interpreted as saying that a query matching x gets a minimum score of $P(u|x)$ for category u. If (x, u) is a forward pair (i.e., x is "$x\_$"), we require x to match a prefix of the query for the rule to apply, while if (x, u) is a backward pair, we require $x$to match a suffix of the query to apply.

Finally, in Step 4 we use selectional preferences to classify test queries. We attempt to match each forward selectional preference against the initial tokens of the query, and each backward preference against the final tokens of the query. We give the query a score for each category u corresponding to the maximum $P(u|x)$ value of any rule that matches it. We then compare the maximum $P(u|x)$ values for a query against a threshold tuned to optimize classification effectiveness (Section 3.4), and assign the query to all u's with values that exceed

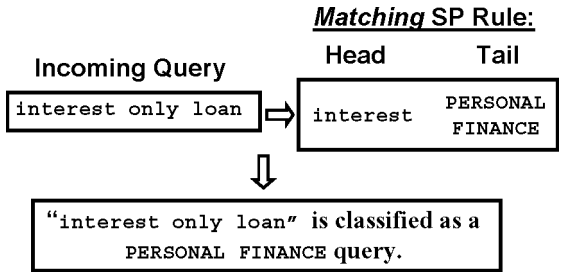

Fig. 4. Selectional preference rule generation—Step 4.

Forward Rules

- harlem club ***X***
  - ENT->0.722
  - PLACES->0.378
  - TRAVEL->1.531
- harley all stainless ***X***
  - AUTOS->3.448
  - SHOPPING->0.021
- harley chicks with ***X***
  - PORN->5.681

Backward Rules

- ***X*** gets hot wont start
  - AUTOS->2.049
  - PLACES->0.594
- ***X*** getaway bargain
  - PLACES->0.877
  - SHOPPING->0.047
  - TRAVEL->0.862
- ***X*** getaway bargain hotel and airfare
  - PLACES->0.594
  - TRAVEL->2.057

Fig. 5. Selectional preference rule examples.

the threshold. Tuning is necessary since the $P(u|x)$ values are estimates of the probability that a subpart of the query would be viewed as belonging to a category (with categories considered as mutually exclusive), not estimates of the probability that the whole query would be viewed as belonging to that category (with categories considered as overlapping). Step 4 is illustrated in Figure 4.

In Figure 5 we show examples of how some real queries from our log are classified by the rules mined from selectional preferences.

The above approach can be compared with conventional rule learning approaches in machine learning [Mitchell 1997]. Like rule learning approaches, it uses labeled data to learn logical relationships (in our case very simple ones) between predictor features and classes, and like some of these approaches uses weighted rules with conflict resolution by taking maximum score. Our linguistic criterion for filtering rules, $S(x)$, is somewhat different from those used for feature and rule selection in a typical rule learner, particularly since it implies a different view of category structure than the classifier itself uses. Our use of ambiguously labeled and structured training data is atypical in rule learning. Finally, most rule learners incorporate an inductive bias toward producing small sets of rules, while the short length of queries requires that as many rules (of sufficient quality) be produced as possible. With respect to this last point, our approach has similarities to association rule learning [Adamo 2000], which emphasizes producing all rules of sufficient quality. The focus in association rule learning is usually not on predictive accuracy, however, nor are issues of semantically ambiguous features typically dealt with.

Table II. Positive Example Overlap (N/A-Not Applicable)

	Exact Match	n-Gram Match	Perceptron
n-Gram match	.7406	N/A	N/A
Perceptron	.1243	.0957	N/A
Selectional preferences	.0894	.0708	.6513

3.4 Tuning and Combining Classifiers

One ideally tunes a text classifier to the need of the particular application [Lewis 1995]. Our exact match classifier (Section 3.1) and its *n*-gram variant (Section 3.1.1) both either match a query or do not, so no tuning is possible. Our perceptron learner (Section 3.2) and Selectional Preference (SP—Section 3.3) classifiers, on the other hand, produce a score for a query on each category, and we must convert these scores to classification decisions.[6] Therefore, for each experimental run we set the thresholds of these two classifiers to optimize the particular microaveraged F_β measure (Section 4.2) to be measured. Tuning was done on a sample (Section 4.1) distinct from the test set. A single numeric value was used as the threshold for the classifiers for all 18 categories. This helped avoid choosing extreme thresholds for categories with little training data, but may have hurt effectiveness to the degree that scores on different categories were not comparable.

Simply tuning the threshold of a classifier cannot make it do well on queries whose important features were simply not present in the training data. In Table II, we show that our component classifiers vary substantially in the category assignments they get correct. Additionally, each individual approach has different strengths and weaknesses. The exact-match and *n*-gram approaches are best for high-precision classification of popular queries (e.g., "nfl football"). The perceptron learner will be best at classifying queries with strong features that are well represented in the manually classified queries (e.g., queries that contain "city" are likely to be about travel or locations). The selectional preference classifier will find latent relationships between queries (e.g., during football season, "eagles" occurs more frequently with known football queries, but during baseball season it's more likely to be associated with ornithology). All of this suggested combining the individual classifiers for greater effectiveness.

We tested a simple combined classifier that assigned a query to each category that any of the component classifiers predicted (hereafter referred to as the disjunctive combination). We individually tuned each component classifier to optimize microaveraged F_β, but no tuning of the combined classifier as a whole was done. We also combined our techniques using a basic preference-order approach, in an effort to shore up precision without sacrificing too much of the recall gained through combination. This involved ranking the component techniques into a preference order by some criteria. Then, during combination, only the classification decisions from the highest-ranked technique that assigns at least one class were used. For these experiments, we determined preference order by ranking each approach by the level of precision it provided on its own

[6]The perceptron algorithm itself sets a threshold on each linear model, but these thresholds implicitly minimize error rate, not F-measure.

(using the validation set described in Section 4.1), highest first. We arrived at a preference order of exact-match lookup first, followed by the perceptron learner, 4-gram lookup, and finally selectional preferences. More sophisticated combination approaches are certainly possible (learning weights for a linear combination, for example), although they fall outside the scope of this study.

4. EVALUATION

Evaluation in the context of an evolving real-world system is always a challenge. We discuss the choices we made in structuring controlled experiments with our data sets. We give an overview of the datasets and effectiveness measures that we used to evaluate each individual technique and the combined techniques used in our query classification approach.

4.1 Data Sets

We used three different kinds of data in our experiments. The first data set (the "training set") was composed of several hundred thousand queries that were manually classified (either individually or as bulk sets of names) by human editors into the set of 20 categories listed in Table I. The "URLs" and "Misspellings" categories were omitted since these would best be handled by special techniques. The human editors were employees of AOL™ who monitored incoming query traffic and classified queries according to their best judgment. They were not domain experts in search, but for the task of assessing queries, assessors have only the query terms to help make their decision, so the necessity of expertise was perhaps less important than in traditional tasks with a manual assessment component, such as the Text Retrieval Conference (TREC), where assessors often have a more verbose definition of the item being judged.[7] The classifications were not governed by a specific set of guidelines, creating an open assessment environment that was not tightly controlled. These 20 categories were chosen to represent the top level of a concept hierarchy similar to the one used by the ODP. The results gathered from our initial work [Beitzel et al. 2005a, 2005b] served as a pilot for this study, motivating us to perform more detailed experiments. The queries in the 18 remaining categories provided exact-match and n-gram lookups for manual classification, were used as training data for the perceptron, and defined class memberships for lexeme's when mining a query log for selectional preferences.

The second data set (the "tuning and testing" set) was a random sample of 20,000 queries out of the entire query stream for 1 week. These queries were manually classified into the same set of 18 categories by a team of human assessors (separate from those who classified the first data set). As with our training data set, "URLs" and "Misspellings" were removed from consideration for these experiments. A tuning set was formed from approximately 25% of the remaining queries. This set was used to tune the thresholds for the perceptron and selectional preference classifiers. The remaining 75% of queries were the test set used for evaluation. A breakdown of the tuning and testing dataset is given in Table III.

[7]The International Text Retrieval Conference (TREC); go online to `http://trec.nist.gov`.

Table III. Tuning and Testing Data Breakdown

Description	Total queries	Total classifications
Original set	20,000	23,780
No URL & Misspell	18,627	21,119
Tuning set	5,113	5,283
Testing set	14,425	15,836

The third and final data set was the large, unlabeled query log that we mined to generate the selectional preference classification rules. The query log used for these experiments contained several hundred million queries[8] received over a period of several weeks by the AOL™ search service. The primary computing resource required for our experiments was disk space, as we used approximately 200 GB to store the query log, data sets, and various intermediate files. This was implementation-dependant to some degree, as it is certainly possible to implement our techniques using more storage-conscious principles.

4.2 Effectiveness Measures

Overlap between our categories was low, but a query could belong to more than one category. We therefore treated each category as a separate binary classification task. We summarized a classifier's effectiveness on a category k by the number of true positives (TP_k), false positives (FP_k), false negatives (FN_k), and true negatives (TN_k), where $TP_k + FP_k + FN_k + TN_k$ equals the number of test queries. Using these counts, we can define the usual measures recall, $R_k = TP_k/(TP_k + FN_k)$, precision, $P_k = TP_k/(TP_k + FP_k)$, and the F-measure [van Rijsbergen 1979]:

$$F_{\beta,k} = \frac{(\beta^2 + 1) \times TP_k}{(\beta^2 + 1) \times TP_k + FP_k + \beta^2 FN_k}, \tag{3}$$

where β lets us specify the relative importance of recall and precision. Values of β lower than 1.0 indicate precision is more important than recall, while values of β greater than 1.0 indicate the reverse.

To summarize effectiveness across all categories, we use the microaveraged [Tague 1981] values of recall, precision, and F:

$$\begin{aligned}
{}_{\mu}R &= \frac{\sum_k TP_k}{\sum_k TP_k + \sum_k FN_k}, \\
{}_{\mu}P &= \frac{\sum_k TP_k}{\sum_k TP_k + \sum_k FP_k}, \\
{}_{\mu}F_{\beta} &= \frac{(\beta^2 + 1) \times \sum_k TP_k}{(\beta^2 + 1) \times \sum_k TP_k + \sum_k FP_k + \beta^2 \times FN_k},
\end{aligned}$$

where summations are over the set of categories. Because our target classes were of nonuniform size, we used microaveraging for our experiments, which gives equal weight to each individual classification (although microaveraging

[8]The precise size of the query log is withheld, as it represents proprietary AOL data.

Table IV. Classification Effectiveness for Each Technique (Disjunctive Combination)—Sorted by Descending Microaveraged Recall

	Micro. precision	Micro. recall	F_1
All techniques	0.1539	**0.5510**	0.2406
4-Gram + perceptron + SP	0.1539	0.5510	0.2406
4-Gram + SP	0.1625	0.5044	0.2459
Exact match + 4-gram + SP	0.1625	0.5044	0.2459
Exact match + perceptron + SP	0.1622	0.4951	0.2443
SP + perceptron	0.1593	0.4758	0.2387
4-Gram + perceptron	0.1690	0.4689	0.2485
Exact match + 4-gram + perceptron	0.1690	0.4689	0.2485
Exact match + SP	0.1745	0.4343	0.2490
4-Gram match	0.1804	0.3940	0.2475
Exact match + 4-gram	0.1804	0.3940	0.2475
SP	0.1672	0.3856	0.2332
Exact match + perceptron	0.2141	0.3056	**0.2518**
Perceptron	0.2103	0.2790	0.2399
Exact match	**0.3079**	0.0990	0.1498

and macroaveraging gave similar results on our data). Note that the microaveraged value of F_β was not derived from the microaveraged precision and recall values.

5. RESULTS AND ANALYSIS

In Table IV, we show the effectiveness, as measured by microaveraged F_1,of all disjunctions of the individual classification approaches. We tuned both the perceptron learner and the selectional preferences classifiers to optimize microaveraged F_1 on the validation set.

We present these results in descending order by microaveraged recall, in keeping with our goal of classifying a reasonably large portion of the query stream.

First, we examine the performance of our four individual approaches. 4-Gram matching achieved the best recall (.3940) among the individual techniques, while still maintaining reasonable precision, with Selectional Preferences following in a close second, but with lower precision. This is especially encouraging, given that the overlap between 4-gram matches and Selectional Preferences was low (see Table II), indicating that substantial benefit might be gained from combining these two techniques. It should be noted that the selectional preferences classifier is only applicable to queries consisting of two or more terms. When evaluating the SP classifier while excluding single-term queries from the evaluation, its recall increased to 0.4434, which was an increase proportional to the percentage of single-term queries in our test set, as expected.[9] Additionally, one might expect that the selectional preferences classifier would benefit from a larger seed log. Based on some preliminary experiments, we concluded that any increased benefit is minimal once the size of the query log reaches the

[9]Precision was slightly higher than the in the case of SPs over all queries (0.1719 vs. 0.1672) because the optimal threshold was recalculated for this case and happened to increase slightly (ordinarily precision would remain unchanged, as the SP classifier never produces a classification for a single-term query).

hundreds of millions. As expected, the perceptron learner and exact-match approaches were able to achieve greater precision (.2103 and .3079, respectively), but at the cost of a substantially lower rate of recall, with the learner achieving a modest .2790, and the exact-match plummeting below 1 in 10 at .0990. It is notable that although precision on exact match seems low at ∼30%, the assessors who judged our test queries were a different group from the editors who classified our manual database. Thus, assessor disagreement, and the limitations of bulk classification of large sets of names, played a large role on inhibiting our maximum achievable precision.

Next, we examine the performance of all possible disjunctive combinations of the individual approaches. As noted above, we have observed low overlap among the individual approaches; therefore, we expected to see substantial improvements in recall when performing a disjunctive combination. As expected, nearly all of the combined approaches were able to classify a substantially larger portion of the query stream than any individual technique alone. In particular, when we combined all techniques, we saw an increase in recall of almost 40% (.5510 vs. .3940) over the individual technique with the best recall (4-grams), while suffering a nominal drop in precision (.1539 vs. .1804). From these results, it appears clear that we were achieving the "best of all worlds" and successfully taking advantage of the unique specialties of each individual technique. In particular, the large increase in recall suggests that the combined approach may be a tenable solution to the recall problem that has hindered past efforts at query classification. Additionally, these results strongly suggest that the selectional preferences classifier is reaching portions of the query stream that the other classifiers cannot classify. This is in keeping with the overlap numbers given in Table II, and demonstrated experimentally by the large increase in recall observed when the SP classifier was combined with any other single approach. This is an important benefit of our system, as the SP classifier is able to adapt to changes in the query stream over time by simply processing an updated query log. In this way, we are able to utilize fully unlabeled data to aid automatic classification, and we are not encumbered by a prohibitive dependence on external sources of information, such as the set of documents retrieved for a query.

To add further insight, we also give results for the preference-order combination of our individual techniques, as discussed in Section 3.4. We present these results in Table V, again ordered by descending microaveraged recall.

As Table V shows, this preference-order combination strikes a better balance between the precision/recall tradeoff; F_1 scores are slightly higher for most approaches. If a search service determines that maximum recall is less important than maintaining a predetermined minimum level of precision, combination strategies like this preference-order combination can be used to achieve more balanced results.

5.1 Changing the Value of β: The Precision/Recall Tradeoff

Different search services may put different values on precision and recall. As discussed above, we can specify a different tradeoff between recall and

Table V. Classification Effectiveness for Each Technique (Preference-Order Combination)—Sorted by Descending Microaveraged Recall

	Micro. precision	Micro. recall	F_1
Exact match + SP	0.1795	**0.4084**	0.2494
4-Gram + SP	0.1800	0.4032	0.2489
4-Gram match	0.1804	0.3940	0.2475
SP	0.1672	0.3856	0.2332
Exact match + 4-gram + SP	0.1849	0.3795	0.2487
Exact match + 4-gram	0.1855	0.3703	0.2472
Exact match + perceptron + SP	0.2065	0.3244	0.2523
All techniques	0.2082	0.3228	**0.2532**
Exact match + 4-gram + perceptron	0.2082	0.3226	0.2530
SP + perceptron	0.1951	0.3190	0.2421
4-Gram + perceptron + SP	0.1973	0.3189	0.2437
4-Gram + perceptron	0.1972	0.3186	0.2436
Exact match + perceptron	0.2236	0.2884	0.2519
Perceptron	0.2103	0.2790	0.2399
Exact match	**0.3079**	0.0990	0.1498

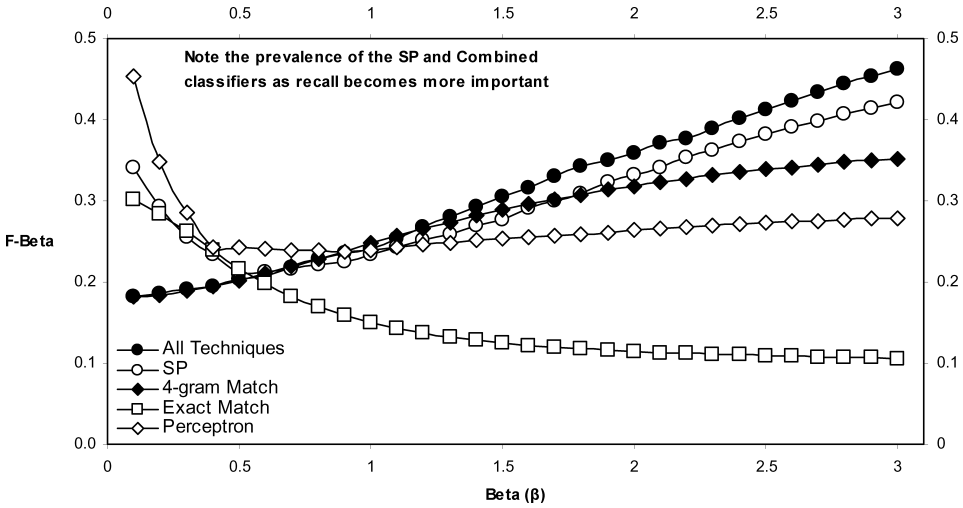

Fig. 6. F_β versus β for each automatic classification technique.

precision by varying the value of β in the F-measure, and retuning the automatic classification approaches and the combined approach to optimize that measure on our validation set. In Figure 6, we plot values of β against microaveraged F_β for each individual approach and the disjunctive combination. The tunable (SP and Perceptron) classifiers were tuned to optimize microaveraged F_β for each tested value of β on the validation set. We can see that both the combined approach and the selectional preferences increasingly outperform the other approaches as higher levels of recall are desired (higher β). This is of particular note because it demonstrates the robustness of the combination of multiple techniques, and selectional preferences in particular, over exact matching and basic machine learning when recall is highly weighted. It also makes intuitive sense, as the disjunctive combination achieves recall at

Table VI. KDD Cup Evaluation

	Precision	F_1
Assessor #1	0.2573	0.2644
Assessor #2	0.1623	0.2090
Assessor #3	0.2483	0.2492
Average	0.2226	0.2409
KDD mean	0.2545	0.2353
KDD median	0.2446	0.2327
KDD std. dev.	0.1338	0.0993

least as good (and almost certainly better) than that of its best component, but precision no better (and almost certainly worse) than that of its best component. The combination also maintains a slight edge over the selectional preferences when β is greater than one (more recall desired), but falls off as expected when precision is more important (lower β), as disjunctive combinations are generally at a disadvantage when precision is emphasized.

5.2 KDD Cup Performance

As discussed in Section 2.2.1, the task for the 2005 KDD Cup was automatic topical query classification. Although participants in the KDD Cup made frequent use of external resources outside the scope of our approach, we evaluated our system using the KDD dataset and judgments described previously in Section 2.2.1.

To facilitate this evaluation, we first needed to map our database of manually classified queries onto the set of specific categories used in KDD Cup. For many categories, this was difficult, as the set of categories initially used for our manual classifications were typically more general than the KDD Cup categories. For cases where it was not possible to map any of our preexisting classifications to a particular KDD Cup category (usually due to insufficient data or categories that were too general), we created a small set of manual classifications for the KDD Cup category in question, in an attempt to make some effort to represent it. We then used our mapped KDD Cup categories to perform classifications on the KDD Cup queries using the exact-match and selectional preferences-based techniques described in Section 3. Specifically, we compared the disjunctive combination of our exact-match lookups and selectional preferences to the median scores for Precision and F_1 from KDD. As done in the KDD Cup, we performed separate evaluations using each of the three sets of judgments from KDD assessors. The average performance was also calculated. We present the results of this evaluation along with the corresponding numbers from KDD Cup in Table VI.

As seen in Table VI, our system is able to succeed fairly well on the KDD Cup data, placing right around the median for both Precision and F_1. These results show a reasonable amount of assessor disagreement, which is to be expected. This level of assessor disagreement is also consistent with the conclusions from prior studies, although the presentation on the KDD Cup Website shows the assessors were almost always in agreement on the top three performers. Recall that one of our goals was to model the problem of automatic query classification

after how it would appear to a real Web search service. Because of this, we imposed operational requirements that prevented us from utilizing expensive outside resources, making our task slightly different and more difficult than that of the KDD Cup, where participants were uninhibited. When considering that the systems competing in KDD Cup frequently made use of information-rich external resources such as retrieved documents and online taxonomies, it is encouraging to see our approach performed as well as it did, outperforming approximately half of all competing approaches.

5.3 Discussion

In this section, we review our three primary research questions identified in Section 3 and discuss the key findings relating to each one.

(1) Is it possible to design a viable automatic query classification technique that utilizes search engine query logs?

As shown in section 5, the selectional preferences classification technique is able to classify queries competitively with the other component techniques. Furthermore, it is shown to be especially proficient in achieving high classification recall.

(2) Can we combine several techniques to improve classification recall?

Both combination methods that were used in this study proved to be very effective in improving classification recall over that of any single component technique. Additionally, there is a large body of research on the most effective means for data fusion and other forms of combining multiple evidence. As such, it is likely that still more sophisticated combination techniques can be applied to further increase classification precision, mitigating one of the biggest limitations of this study.

(3) How does an operationally restricted classification approach compare to those which use external resources?

It is clear that an approach to query classification that is not allowed to access external sources of information is placed at an immediate disadvantage. Despite this, we found that our approach could still compete ably with approaches used in the KDD Cup, outperforming roughly half of them. To achieve this level of performance in the face of these operational restrictions is encouraging.

To round out the discussion, we also examine the limitations of this study. One obvious limitation is the classification precision of our techniques. Clearly this is an area that needs improvement when false-positive classification decisions are a concern. In the absence of high-quality training data, supervised learning methods and broad rule-based techniques like the selectional preferences classifier are going to have depressed precision. This is likely best mitigated by pruning the classification rules according to some empirical criteria or enforcing a more stringent minimum selectional preference strength. Additionally, more training data (in the form of seed manual classifications and logged queries) will likely lead to higher precision. A further limitation is that the categories used in this study may be too general to be of use for all

applications. The experiments performed for our comparison to KDD Cup suggest that it is perfectly reasonable to use a larger set of categories that are individually more narrow in scope, even when some of these categories are not well specified. More detailed study of the performance of our approach under these circumstances is warranted. Finally, this study did not attempt to quantify the degree to which retrieval might be improved by the classification approaches presented, though we hope to evaluate this in future work.

6. CONCLUSIONS

Determining the topic of unrestricted Web queries is an important problem in modern Web search. This is increasingly true as modern search services increasingly use large numbers of specialized backend databases to provide special features (maps, stock quotes, advertisements, etc.). We proposed a system for automatic Web query classification that combines manually classified queries with techniques from machine learning and computational linguistics. This combined approach allows higher recall and smoother tradeoffs between recall and precision than any of the component approaches. Furthermore, it is competitive with other classification systems, as measured by the 2005 KDD Cup, despite not having access to information-rich external resources such as the results of text classification on documents retrieved for a query, or the contents of online taxonomies and directories. Our combined approach is able to leverage the specific strengths of each individual approach to classify a much larger portion of the query stream than would be possible using any of the individual methods alone. Moreover, by leveraging the unlabeled data contained in user query logs, we have created a classification system that is automatically robust to changes in the query stream, for as the users change their queries, a system that relies on query logs for information will be able to adapt to them immediately and automatically. Because of this, we can minimize the need for periodically labeling new training data to keep up with changing trends in the query stream over time, and we are not forced to rely on computationally prohibitive forms of external information to maintain our classification effectiveness.

There are several potential areas for future work, including the following:

—finding ways to improve the precision of both our component and combined techniques;
—expanding the existing manual classification with queries that are classified by our approach, and subsetting the manually classified queries by linguistic properties;
—improving our selectional preference algorithm by more explicit handling of semantic and structural ambiguity, and incorporating ideas from traditional rule learning; and
—examining specific topical categories where one approach outperforms another, and using this information to create more sophisticated combined classification techniques.

REFERENCES

ADAMO, J.-M. 2000. *Data Mining for Association Rules and Sequential Patterns: Sequential and Parallel Algorithms*. Springer, Berlin, Germany.

BEEFERMAN, D. AND BERGER, A. 2000. Agglomerative clustering of a search engine query log. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, New York, NY, 407–416.

BEITZEL, S. M., JENSEN, E. C., CHOWDHURY, A., GROSSMAN, D., AND FRIEDER, O. 2004. Hourly analysis of a very large topically categorized Web query log. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 321–328.

BEITZEL, S. M., JENSEN, E. C., LEWIS, D. D., CHOWDHURY, A., KOLCZ, A., AND FRIEDER, O. 2005a. Improving automatic query classification via semi-supervised learning. In *Proceedings of the Fifth IEEE International Conference on Data Mining*. IEEE Computer Society Press, Los Alamitos, CA, 42–49.

BEITZEL, S. M., JENSEN, E. C., LEWIS, D. D., CHOWDHURY, A., KOLCZ, A., FRIEDER, O., AND GROSSMAN, D. 2005b. Automatic Web query classification using labeled and unlabeled training data. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 581–582.

BEITZEL, S. M., JENSEN, E. C., CHOWDHURY, A., FRIEDER, O., AND GROSSMAN, D. 2006. Temporal analysis of a very large topically categorized web query log. *J. Amer. Soc. Inform Sci. Tech*. To appear.

BOT, R. S., WU, Y.-F. B., CHEN, X., AND LI, Q. 2005. Generating better concept hierarchies using automatic document classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management* (CIKM). ACM Press, New York, NY, 281–282.

BRODER, A. 2002. A taxonomy of Web search. *SIGIR For. 36*, 2 (Fall), 3–10.

CHO, J., GARCIA-MOLINA, H., AND PAGE, L. 1998. Efficient crawling through URL ordering. In *Proceedings of the 7th International World Wide Web Conference*. Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 161–172.

CHOWDHURY, A. AND PASS, G. 2003. Operational requirements for scalable search systems. In *Proceedings of the 12th ACM International Conference on Information and Knowledge Management* (CIKM). ACM Press, New York, NY, 435–442.

COVER, T. M. AND THOMAS, J. A. 1991. *Elements of Information Theory*. Wiley-Interscience, New York, NY.

CRASWELL, N. AND HAWKING, D. 2004. Overview of the trec 2004 Web track. In *Proceedings of the Thirteenth Text Retrieval Conference (TREC 2004)*. NIST, Gaithersburg, MD. 89–97.

CRASWELL, N., HAWKING, D., WILKINSON, R., AND WU, M. 2003. Overview of the TREC 2003 Web track. In *Proceedings of the Twelfth Text Retrieval Conference (TREC 2003)*. NIST, Gaithersburg, MD, 78–92.

DAS-NEVES, F., FOX, E. A., AND YU, X. 2005. Connecting topics in document collections with stepping stones and pathways. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management* (CIKM). ACM Press, New York, NY, 91–98.

EASTMAN, C. M. AND JANSEN, B. J. 2003. Coverage, relevance, and ranking: The impact of query operators on Web search engine results. *ACM Trans. Inform. Syst. 21*, 4 (Oct.), 383–411.

GLOVER, E. J., LAWRENCE, S., BIRMINGHAM, W. P., AND GILES, C. L. 1999. Architecture of a metasearch engine that supports user information needs. In *Proceedings of the 8th ACM International Conference on Information and Knowledge Management* (CIKM). ACM Press, New York, NY, 210–216.

GRAVANO, L., HATZIVASSILOGLOU, V., AND LICHTENSTEIN, R. 2003. Categorizing web queries according to geographical locality. In *Proceedings of the 12th ACM International Conference on Information and Knowledge Management* (CIKM). ACM Press, New York, NY, 325–333.

GROSSMAN, D. A. AND FRIEDER, O. 2004. *Information retrieval: Algorithms and Heuristics*. Springer, Berlin, Germany.

JANSEN, B. J., SPINK, A., AND PEDERSON, J. 2005. A temporal comparison of altavista Web searching. *J. Amer. Soc. Inform. Sci. Techno. 56*, 6, 559–570.

JANSEN, B. J., SPINK, A., AND SARACEVIC, T. 2000. Real life, real users, and real needs: A study and analysis of user queries on the web. *Inform. Process. Manage. 36*, 2 (Mar.), 207–227.

JOACHIMS, T. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods—Support Vector Learning*, B. Scholkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA.

KANG, I.-H. AND KIM, G. 2003. Query type classification for web document retrieval. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 64–71.

KARDKOVACS, Z. T., TIKK, D., AND BANSAGHI, Z. 2005. The ferrety algorithm for the KDD Cup 2005 problem. *SIGKDD Explor. 7*, 2 (Dec.), 111–116.

KOWALCZYK, P., ZUKERMAN, I., AND NIEMANN, M. 2004. Analyzing the effect of query class on document retrieval performance. In *17th Australian Joint Conference on Artificial Intelligence (AI-04)*. Springer-Verlag, Berlin, Germany, 550–561.

KRAUTH, W. AND MEZARD, M. 1987. Learning algorithms with optimal stability in neural networks. *J. Phys. A 20*, 745–752.

LAWRENCE, S. AND GILES, C. L. 1998. Searching the World Wide Web *Science*, 98–100.

LEWIS, D. D. 1995. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 246–254.

LI, Y., ZARAGOZA, H., HERBRICH, R., SHAWE-TAYLOR, J., AND KANDOLA, J. S. 2002. The perceptron algorithm with uneven margins. In *Proceedings of the 19th International Conference on Maching Learning*. Morgan Kaufmann San Francisco, CA, 379–386.

LIGHT, M. AND GREIFF, W. 2002. Statistical models for the induction and use of selectional preferences. *Cog. Sci. 26*, 3 269–281.

MANMATHA, R., FENG, A., AND ALLAN, J. 2002. A critical examination of TDT's cost function. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 403–404.

MANNING, C. D. AND SCHUTZE, H. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

MARTIN, A., DODDINGTON, G., KAMM, T., ORDOWSKI, M., AND PRZYBOCKI, M. 1997. The DET curve in assessment of detection task performance. In *Proceedings of the 5th ESCA Conference on Speech Communication and Technology* (Eurospeech '97), (Sept.). 1895–1898.

MCCARTHY, D. AND CARROLL, J. 2003. Disambiguating nouns, verbs, and adjectives using automatically acquired selectional preferences. *Computat. Ling.* 29, 4 (Dec.), 639–654.

MITCHELL, T. M. 1997. *Machine Learning*. McGraw-Hill, New York, NY.

NTOULAS, A., CHO, J., AND OLSTON, C. 2004. What's new on the Web? The evolution of the web from a search engine perspective. In *Proceedings of the 13th International Conference on the World Wide Web* (WWW). ACM Press, New York, NY, 1–12.

RESNIK, P. 1993. Selection and information: A class-based approach to lexical relationships. Unpublished manuscript. University of Pennsylvania, Philadelphia, PA.

SALTON, G., YANG, C. S., AND WONG, A. 1975. A vector-space model for automatic indexing. *Commun. ACM 18*, 11 (Nov.), 613–620.

SEBASTIANI, F. 2002. Machine learning in automated text categorization. *ACM Comput. Surv. 34*, 1 (Mar.), 1–47.

SHEN, D., PAN, R., SUN, J.-T., PAN, J. J., WU, K., YIN, J., AND YANG, Q. 2005. Q^2c@ust: Our winning solution to query classification in KDD Cup 2005. *SIGKDD Explor. 7*, 2 (Dec.), 100–110.

SPINK, A. AND JANSEN, B. J. 2004. *Web Search: Public Searching of the Web*. Springer, Berlin, Germany.

SPINK, A., JANSEN, B. J., WOLFRAM, D., AND SARACEVIC, T. 2002a. From e-sex to e-commerce: Web search changes. *IEEE Comput. 35*, 3 (Mar.), 107–109.

SPINK, A., OZMUTLU, S., OZMUTLU, H. C., AND JANSEN, B. J. 2002b. U.S. versus European Web searching trends. *SIGIR For. 36*, 2 (Fall), 32–38.

SPINK, A., WOLFRAM, D., JANSEN, B. J., AND SARACEVIC, T. 2001. Searching the Web: The public and their queries. *J. Amer. Soc. Inform. Sci. Tech. 52*, 3, 226–234.

SULLIVAN, D. 2006. Searches per day. Search Engine Watch. Go online to http://searchenginewatch.com/reports/article.php/2156461.

TAGUE, J. M. 1981. The pragmatics of information retrieval experimentation. In *Information Retrieval Experiment*, K. S. Jones, Ed. Butterworth-Heinemann, London, U.K. 59–102.

VAN RIJSBERGEN, C. J. 1979. *Information Retrieval*. Butterworth-Heinemann, London, U.K.

VOGEL, D., BICKEL, S., HAIDER, P., SCHIMPFKY, R., SIEMEN, P., BRIDGES, S., AND SCHEFFER, T. 2005. Classifying search engine queries using the Web as background knowledge. *SIGKDD Explor. 7*, 2 (Dec.), 117–122.

VOORHEES, E. M. 2004. Overview of the TREC 2004 question answering track. In *Proceedings of the Thirteenth Text Retrieval Conference* (TREC 2004, Nov.). NIST, Gaitheraburg, MD.

WEN, J.-R., NIE, J.-Y., AND ZHANG, H.-J. 2001a. Clustering user queries of a search engine. In *Proceedings of the 10th International Conference on the World Wide Web* (WWW). ACM Press, New York, NY, 162–168.

WEN, J.-R., NIE, J.-Y., AND ZHANG, H.-J. 2001b. Query clustering using content words and user feedback. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 442–443.

WEN, J.-R., NIE, J.-Y., AND ZHANG, H.-J. 2002. Query clustering using user logs. *ACM Trans. Inform. Sys. 20*, 1 (Jan.), 59–81.

Received March 2006; revised July 2006, October 2006; accepted October 2006