Detecting and supporting known item queries in online public access catalogs

ABSTRACT

When users seek to find specific resources in a digital library, they often use the library catalog to locate them. These catalog queries are defined as known item queries. As known item queries search for specific resources, it is important to manage them differently from other search types, such as area searches. We study how to identify known item queries in the context of a large academic institution's online public access catalog (OPAC). We also examine how to recognize when a known item query has retrieved the item in question. Our approach combines techniques in machine learning, language modeling and machine translation evaluation metrics to build a classifier capable of distinguishing known item queries with an accuracy of 72% and correctly classifies titles for whether they are the known item sought with an accuracy of 86%. To our knowledge, this is the first report of such work, which has the potential to streamline the user interface of both OPACs and digital libraries in support of known item searches.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval: *search process;* H.4.m [**Information Systems Applications**]: Miscellaneous.

General Terms

Algorithms, Experimentation, Human Factors.

Keywords

Known item queries, query language model, query types.

1. INTRODUCTION

Consider a scenario in which an academic library patron wants to see whether his institution subscribes to a specific journal in his research area. Knowing the journal title makes it easy for him to construct a query to the local online public access catalog: for example, "journal of housing for the elderly". If the periodical is carried by the library and the title appears in the search results, the user can locate the journal. If the library does not carry it, the catalog may retrieve irrelevant resources that share words with the query. In this case, the user may have to sift through the returned results, only to conclude that the library does not carry the journal.

These types of queries, in which the user knows what resource he is seeking prior to searching, are often called *known item queries*.

Conference'04, Month 1–2, 2004, City, State, Country.

Slone [17] differentiates known item queries from two other types of catalog queries: a) unknown item queries, where the user tries to locate resources to solve a problem or address an issue, and b) area queries, where the patron uses the catalog to locate a library section or area. Since known item queries are used to retrieve a specific resource, results that do not match are irrelevant. This is in contrast with unknown item and area searches where partial matches to a query may meet the user's needs. As such, the detection and proper support of known item queries is important to the satisfaction of library users.

How important is known item search? In the setting of an OPAC, it is very important. Larson [9] points at the long term decline of subject searching in OPACs, in which known item search accounts for a growing proportion of library catalog searches, up to 50%. However, supporting these types of searches has largely been ignored by the information retrieval community, whose focus has been on topical search (e.g., TREC bakeoff competitions [2]). While these efforts have improved the state of the art for topical search, we see a need to support better known item query detection and retrieval.

To address this problem, we propose a system that a) detects known item queries, and b) detects when such queries retrieves the desired known items. The system performs both tasks by employing a learning paradigm. A query model is first built from past known item queries and their corresponding catalog search results. Our final query model itself consists of three components: a set of query features distilled from empirical observation, *n*-gram language models and matching models for returned search results using machine translation scoring metrics. Then, when a user issues a new query to the catalog, the system checks whether the known item query model applies to the new query and its search results. A key finding of our study is that knowing what results of the query are help to improve known item query detection significantly. The resulting system is able to distinguish known item queries at 80% of the level of human performance. We also show that it is possible to correctly identify when the known items are retrieved by the catalog with 95% of the level of human performance.

These two parts of the system can be integrated with an online public access catalog (OPAC) to improve catalog usability and user satisfaction. When the known item is in a library's collection, proper handling known item queries can route the user to the desired resources quickly. When the known item is not in a library's collection, the system can let the user know that the library does not carry the title. In both cases, the user does not have to wade through a list of search results, saving time for the user and avoiding frustration.

2. OUR APPROACH

To identify and support known item queries, we break down the process into two separate tasks:

1. to classify user queries as either known item queries or not;

2. to identify which, if any, of the returned search results are the known items sought.

We examine both tasks as machine learning problems. In machine learning, a system develops knowledge based on training data. Using the training data, we apply machine learning to create models that solve the two problems.

To perform this study we first needed to collect ground truth annotations to use as training data. We detail our data collection process and assess the level of inter-judge agreement to establish an upper bound for performance. We then detail our automated approaches to the two tasks. For known item query detection, we start with a simple model that uses only the query itself as input for the classification task. We then develop a more comprehensive model that adds the returned search results as input to this first task and shows how it improves performance. We then examine the second task, search result classification, to see whether automated methods can distinguish known items from other search results.

3. DATA COLLECTION

To provide the necessary data for both training and evaluation, we need to examine actual catalog queries and their search results. The sample queries are drawn from the transaction logs of INNOPAC, the local OPAC system at the *anonymized University*[1]. These logs consist of approximately 290,000 unique queries issued to the simple keyword search interface during a period of about fifteen months. We took sample queries from the logs for annotation and analysis. The queries were carefully chosen to represent diverse types of queries that were encountered in the logs. We detail this procedure in Section 4.1.

Given queries, we need to know whether they are known item queries or not. The optimal way to make this judgment is to ask the user who issued the original query, as was done in previous library studies [17]. However, our transaction logs do not collect the identity of the original library user, as most catalog queries come from anonymous access via public access terminals or the internet. As such, we had to simulate this judgment using a separate set of human subjects. A total of 9 participants (including the authors, all of whom were university students or staff) took part in annotating the set of queries and their search results. The participants were directed to judge queries and catalog search results. Instructions that defined the tasks and gave example answers were given to the subjects before they were allowed to annotate. Online help with the evaluation was available at all times during the annotation process. The annotation process for a query consisted of three tasks: 1. query judgment (using only the query), 2. query judgment (using the query and the search results) and 3. search results judgment. Each of the 320 queries was annotated by at least one subject.

In the first query judgment task, participants were instructed to decide whether a given query was a known item query or not. The subjects were given just the query for the first task and asked to rate the query on a 9-point Likert scale, ranging from "Definitely a known item query (1)" to "Definitely not (9)".

In the second query judgment task, illustrated in Figure 1, the participants re-judged the query given the results of the search in the local OPAC. We showed participants the first page of results returned by the catalog. Subjects were asked to re-judge the query on the known item task on the same 9 point Likert scale, with their previous choice highlighted and chosen as a default. We believed that this additional source of information might help polarize judgments. For example, a query that may have been unfamiliar to the subject might be re-judged as a known item if there are exact title matches returned by the OPAC.

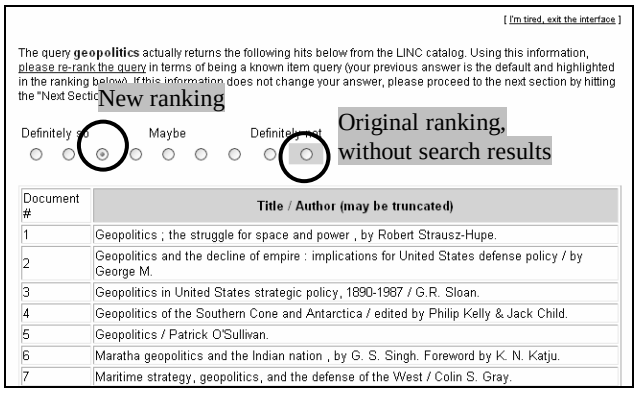

Figure 1: The query re-judging task, using search results. The query "geopolitics" has been re-judged to likely be a known item query.

A final search results judgment task followed. Here, the participants were asked to assume that the query was actually a known item query (*i.e.*, that the user who originated the query had a specific resource in mind to retrieve) and to judge whether each of the search results was the intended item. Again, the participants were asked to classify the titles along the 9-point Likert scale: "Definitely the item sought (1)" to "Definitely not the item sought (9)". Note that multiple items may be classified as the known item, as the catalog can return identically named titles when the library has multiple copies of the title or has the title in multiple formats (*e.g.*, "To Kill a Mockingbird [large-print]" and "To Kill a Mockingbird [videocassette]").

Using this interface, a total of 320 query judgments and re-judgments were collected, along with 1,500 query results judgments.

3.1 Inter-judge Agreement

Classifying queries as known item or non known item searches may seem easy. Queries such as "The Catcher in the Rye" or "cosmos carl sagan" are simple to classify as known items when one knows of the title in question. However, many book titles are similar to general subject areas, perhaps to enhance a book's merchantability. In such cases, it may be difficult to distinguish whether the query is known item or not without prior knowledge. For example, to digital library researchers and practitioners "practical digital libraries" could be a known item query searching for the text by Michael Lesk but "practical digital archiving" is not. To the non-specialist, the two may be viewed as equivalent.

As the same item may be rated by subjects differently, the known item query and title judgment tasks are not rigorously defined. Still, we argue that the concept of a known item query is meaningful and that the tasks that are well-defined. To support this, we calculated Pearson's R among the users. This correlation score measures inter-judge agreement: a value of '1'

[1] References to authors' institution withheld for blind review.

indicates perfect correlation, whereas values of '0' and '-1' indicate no correlation and perfect opposition, respectively. Well-defined tasks can be defined as having an R value greater than .4. Table 1 shows the correlation analysis for the three tasks, respectively. We show both per-item R scores as well as averaged R scores over all subject pairs.

Table 1: Pearson's R for data collection tasks. Per-item and per-user averages shown for both raw and binary classes.

Task	9-point scale		2 class	
	Per-item	Avg Per-subject	Per-item	Avg. Per-subject
1. Query (just query)	.577	.546	.392	.411
2. Query (w/ results)	.604	.664	.593	.656
3. Query Results	.745	.763	.739	.750

In applications, we often need to simplify the problem to a binary classification: for example, whether a query is a known item query or not. The last two columns of Table 1 show the same correlation analysis for the tasks, in which the 9-point scale is collapsed to two classes: known item queries (raw values 1 and 2) and not known item queries (raw values 3-9). Note here that the reduction to two classes is asymmetric: we collapse only the top two classes for which subjects had marked as most sure as known items queries. This is because we want the detection of known item queries to be more precise (at the expense of non-known item queries).

The results in both the raw 9-point and the binary classification show similar characteristics. Each of the tasks seems well-defined, having correlations largely above .4. Additionally, we see that having the query results helps raters to establish more consistent ratings for their known item judgments, moving the tasks from a medium level of correlation (.4 to .6) to a high level (above .6). This validates our hypothesis that the search results are important to use in the query classification task. Examining the data we see that when the OPAC does return items that exactly match or are very similar to the query, users agree that the search was a known item one. We also see that the title classification task is much easier. This is because the large bulk of the returned items are not known item matches.

Aside from showing that these tasks are well-formed, the correlation scores also act as an upper bound on the performance of automated methods. We consider an automatic method successful if it can achieve an R score at or close to the level of human-human agreement.

4. KNOWN ITEM QUERY CLASSIFICATION

We now turn our attention to automatic methods to perform these three tasks. As humans, one can often identify a known item query by simple inspection. Slone [17] notes that words from the title and author's name frequently help to distinguish a known item query from other types of queries. In support of this, Kilgour *et al.* [5][8] explored how to best conduct known item queries efficiently – and concluded that the author's surname plus first and/or last title words can help to pinpoint the known item within the first 20 results over 98% of the time in their university OPAC. Our work is complementary to these works, as we cooperatively identify whether or not a query is a known item one, rather than prescribe guidelines for constructing them.

Aside from these hints, we consider other features that may help identify known queries. We look at two examples: "Hill Raymond Coding Theory – A First Course" and "japan and cultural". The first query is a known item query and the second one is most likely not a known item query. By examining a small training portion of the collected data of the annotation results we can make the following general observations about known item queries:

- **They are longer:** Users tend to type more when they have a specific resource in mind. In an academic setting such as ours, they may be copying a reference or citation from a syllabus or pasting one from a web site into the search field. We believe that the latter case would be more prominent in digital library scenarios in which catalog access is often juxtaposed with internet access. The problems of query construction and formulation are alleviated in known item searches, and as such, longer queries come more easily to the known item searcher.
- **They contain determiners:** In English titles, determiners (such as "*the*", "*an*", and "*a*") are often parts of book titles and are thus also prevalent in known item queries. In contrast, most area or unknown item searchers do not type determiners into search boxes as many know that such words are often ignored by OPACs.
- **They contain proper nouns:** Similar to Slone and Kilgour *et al.*'s studies, we have observed occurrences of author and editor names in known item queries, as in the first example. When seen together with other features such as common nouns (which may indicate title words), proper nouns are a strong indicator of a known item query. Surnames alone may indicate an author search, but not necessarily a known item search (as an author may have written several works).
- **They contain mixed case:** Known item queries often match exactly the title's orthographic case, as the user is likely to be pasting or typing in the query from another resource at hand. In our local OPAC, the catalog search is case insensitive. Even though library users may be aware of this, disregarding case is an active process that requires a user to think about their input. A user who may be copying or pasting from another source is unlikely to engage in this process and is more likely to copy the title into the search box verbatim.
- **They contain certain advanced operators:** INNOPAC allows users to limit portions of their search to specific fields. For example, "(a: author)" specifies that "author" should appear in the author / editor field of the retrieved result. A strong indicator of a known query is when both the author or title limit operators are present. However, not all OPAC operators indicate a known item query. In our observation, truncation, grouping and proximity operators (*i.e.*, "**", "*()*" and "*within <digit>*" or "*near*" in INNOPAC) are strong indicators of area or unknown item search. Boolean "*and*" and "*or*" operators are more difficult to characterize: they do appear in titles of books and therefore in known item queries, but are also used to bridge subjects in area and unknown item search.
- **They contain keywords:** Keywords that indicate a type of publication are strong indicators of a known item.

Keywords such as "journal", "course" and "textbook" usually connote the desired type of resource, rather than a keyword search for the word. Similarly, many titles in libraries but few subject headings consist of these words.

To decide whether a particular query is a known item query or not, we combine these separate pieces of evidence to make a decision. This is done using machine learning, in which the annotated queries from Section 3 are used to learn a model based on features distilled from the above evidence. Each query is transformed into a set of numeric values that reflect the query's length and its words' parts-of-speech, as obtained from a publicly available part-of-speech tagger [15]. We also encode whether certain advanced operators or publication specific keywords are present. At the end of this process, a query is represented as a set of 16 basic feature values.

4.1 Query Sampling by Query Types

Ideally, we would annotate the full set of 290,000 historical queries from the logs. This is infeasible as we have limited resources and thus a sampling of the queries must be made. A random sampling is a logical choice if we have no prior bias or information. However, since we believe that the features above are important to known item queries classification, we can choose to annotate queries that maximize the discrimination between the two classes as much as possible.

This is done by choosing queries that possess different values in their 16 basic features. All queries that exhibit the exact same feature values could be represented by a single, annotated query. We say that queries that exhibit the same feature values belong to the same *query type*. This makes a simplifying assumption that the features of query length, keyword content, parts-of-speech and case uniquely determine whether it is a known item query or not. Although this assumption is false, it allows us to sample the query space more effectively to cover more feature values.

Table 2: Sample query types and representative queries.

Query Type Description	Sample Query	Instances (% of corpus)
Two word, average length word query, no proper nouns	diagnosis surgery	35,014 (12%)
Single word, long query, tagged as proper noun	lobachevsky	1,070 (0.3%)
Medium length, multiword query with determiner "the"	the melodrama's role	999 (0.3%)
5 words, short query, containing advanced operators	short stor* and (t:familiar)	34 (0.001%)

In this way, we processed all 290,000 queries through our feature extraction program to obtain their query type. This resulted in a total of 1,690 unique query types. A few of these query types and their demographics are shown in Table 2. As expected, query distribution among the query types is not uniform: for example, we see partial evidence that very few of the queries contain advanced operators, which supports earlier findings in query analysis.

We ordered the 1,690 query types by their descending frequency (representing their coverage of the query set). The first 320 were annotated by our human subjects in our study. These 320 annotated query types cover about 94% of all queries in our corpus.

4.2 Bigram Query Language Modeling

While we believe the features discussed earlier are important for known item query discrimination, there are potentially limitless words and word patterns that might help discriminate known item queries. Although we may not be able to enumerate all such patterns, we still want to create a language model that characterizes known item queries. Given annotated queries, we first build a language model for known item queries and another for non-known item queries. At run-time, we pass the target query to each language model to find which model is a better fit for the query. This information can then be used as an additional feature for machine learning.

Language modeling is common in speech recognition [6], and a popular approach here is to use n-gram modeling. This approach has also recently been used for authorship attribution and genre detection [14] and recently for known-item finding on the web [12]. For our work *n*-gram is defined as a sequence of *n* words (as opposed to characters, for example). An n-gram language model estimates the probability of a word sequence $\boldsymbol{P}(w_1,w_2,\ldots w_n)$ by decomposing the problem as a product of simpler conditional probabilities.

As we have limited annotated data, we use a simple bigram (n=2) model, which limits the context to two words:

$$P(w_1, w_2, ..., w_m) \cong \prod_i^m P(w_i \mid w_{i-1})$$

Such a model uses the previous word w_{i-1} to predict the current word w_i. We preprocess the annotated queries, canonicalizing spaces and performing case folding. The training queries are then used to construct smoothed nine separate bigram language models, each one corresponding to one raw Likert-scale value. Then, to evaluate a test query, we test its goodness of fit to the language models by calculating its perplexity:

$$Perplexity = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P(w_i \mid w_1..w_i)}}$$

The language model with the lowest perplexity is the model that has the highest probability of generating the test query, and is thus the language models' best guess of the classification of the query.

4.2.1 Bootstrapping a language model

We can construct a language model using just the 320 annotated queries. However, since the annotated corpus is small in size, the resulting model may not generalize well to test queries (*i.e.*, the model may overfit the data). To address this problem of data sparsity, we need to find comparable queries to the annotated ones to use as training data to construct the language model.

As we have annotated instances of query type that represent sets of queries that have the same feature values, we can use all the queries that belong to a query type to help build the language model. This is a semi-supervised approach that enlarges the 320 labeled instances to over 270,000 noisily labeled queries.

4.3 Methods and Evaluation

We thus process the 320 annotated queries into the 17 features total: the 16 basic features and a language model feature. For the language model, we assess both using the standard supervised language model using only the 320 annotated query instances as well as the bootstrapped method that uses over 270,000 query instances. We combine these features with the participant's labels from the first query classification task. We use a decision tree model and support vector machines to create suitable machine learned models. We use standard ten-fold cross-validation to perform training and testing partitioning of the annotated data set. Although there are many models for machine learning, we chose SVMs, as past work has shown good performance on a wide range of tasks and decision trees as the resulting model is readily understandable to humans. We use the SVM and decision tree (named "J48") algorithm implementation in the freely-available Weka [19] machine learning package.

A majority baseline would pick the mode of both classes not exhibit any correlation with the human judges, although by pure instance accuracy, it would achieve 20% accuracy on the 9-point scale problem (always choosing "3" (likely a known item query)) and 60% accuracy on the two class problem (not a known item query). In this paper, we do not consider the instance accuracy of the learner as we have a categorization problem that has a skewed distribution. To measure performance that favors good classification accuracy on a per-category basis, we can use macro-averaged metrics. In our case, since we have numeric classes, we can use correlation with human judgment, which is the metric which we will use throughout the remainder of the paper.

Table 3: Correlation of the models on the basic query task.

Configuration	Pearson's R correlation	
	9 point Likert	Two Class
Majority Baseline	0	0
Inter-judge Agreement	.546	.411
Bigram LM only (*i.e.*, BLM)	.150	.133
Bootstrapped BLM only (*i.e.*, BBLM)	.295	.210
J48 (BF)	.327	.057
J48 (BF + BLM)	.438	.151
J48 (BF + BBLM)	.284	.093
SVM (BF)	.377	.195
SVM (BF + BLM)	.318	.133
SVM (BF + BBLM)	.402	.281

We show the performance of the different learners and feature combinations in terms of their Pearson's R values in Table 3. For features we use "BF" to denote the 16 basic features, "BLM" to denote the bigram language model, and "BBLM" to denote the bootstrapped version of the bigram language model. Note that the language models themselves can be used without a machine learning framework to predict the resulting class.

Performance on this task is upper bounded by human agreement and lower bounded by the majority baseline. Inter-judge agreement as measured by Pearson's R is capped by .546 for the 9-point scale and .411 for the simplified 2 class problem. Our learned models perform at .438/.546 = 80% of human performance using decision trees with the simple bigram language model for the 9-point scale, and at .281 / .411 = 68% using a bootstrapped bigram language model with SVMs in the two class problem. Also, both the simple and bootstrapped language model features turn out to be weak predictors of human judgment, but the bootstrapped model has a much better correlation, possibly because it is provided with a much larger data set for model building (albeit a noisy one). When suitably paired with a learner, is able to improve upon the performance of the learners using just the basic features.

An analysis of the best performing decision tree model reveals that the language model feature is the most important for classification, as it is at the root of the decision tree. The number of words and whether the words of the query are capitalized are also strong features used by the decision tree. The presence of advanced features and publication keywords do not contribute much in the decision tree classification performance.

5. QUERY CLASSIFICATION USING SEARCH RESULTS

Thus far we have only used evidence within the query itself to accomplish known item query discrimination. In Section 3.1, we showed that agreement among human subjects increased when they were allowed to examine the catalog titles returned by the OPAC.

A logical extension of the previous experiments is to leverage these catalog results in the learning framework. Similar to the first task, we distill four observations about how known items and the queries that attempt to retrieve them match each other:

- **The sequence of words in the item's title overlap significantly with the query:** Matching individual words between the item and the query is not enough, but a known item query and its corresponding item will share many word sequences. For example, the word sequence "*family in a changing society*" is shared entirely by the known item and the query. A title such as "*Family Caregiving in a Changing Society*" may share all of these keywords but is not likely to be a known item. Thus the order of the words in the query plays an especially important role in determining whether a retrieved item is a known item.
- **The overlap between the query and the title should be a significant portion of the title:** Titles that have all of the word sequences found in the query are not necessarily known item matches. Consider the query "*the bible*" and the item "*Great people of the Bible and how they lived*". Although the query is fully embedded within the item's title, it is not likely the known item being sought after. In our observation, this is especially true for long titles and short queries.
- **First and last word sequences in the query are important to match in the title:** This is illustrated in the example in Figure 3. Here, we see that the beginning two words ("*marriage and*") and the last two ("*changing society*") are matched only in the twelfth, correct item. These two positions are important to distinguish known items from distracters.

- **Publication keywords are important to match:** If the known item query has a publication specific keyword (again, words such as "journal", "atlas" or "guide"), they should be present in the title as well.

Thus, if the returned titles exhibit one or more of these matching characteristics, we have partial evidence that the query is a known item query. Note the difference when a query retrieves no search results (0 titles) as opposed to retrieving results that do not match well with the query in the categories above. In the former case, we have little evidence to change our decision (as our institution may simply not carry the title searched for), but in the latter, we have evidence that the search is intended to find subject material and not known items.

5.1 Computing Overlap with Machine Translation Metrics

Sequential matching of words between the query and the item title are important. To model this, we adopt metrics in machine translation (MT) that measure translation fidelity.

To evaluate an MT system, a reference translation is first created and an automated system's translation is compared against it. A translation with high fidelity should match the reference in terms of its vocabulary, word order and length. Similarly, a known item should match a query intended to retrieve it along the same criteria.

The NIST and BLEU machine translation metrics model the goodness of fit of a candidate translation to a reference translation. Both metrics use a modified precision count of the number of n-gram matches between the reference and candidate. BLEU is a normalized score between 0 and 1 that combines n-gram fidelity at several sizes [12], whereas NIST uses only trigram fidelity (n=3) and seeks to maximize the score difference between different candidate translations [9]. Both metrics also employ a length penalty for translations that are overly long.

Given a retrieved title and the search query, we can calculate both scores by using the title as the reference translation and the search query as the translation. These two metrics are used to measure the first two observations: that known item queries match the title sought in terms of vocabulary, order and length.

5.2 Other Features

To model the final two observations, we encode separate values for when the first and the last query bigrams are matched, the total number of titles returned by the OPAC, and a Boolean feature that indicates when a publication keyword in the query is matched. We encode the total number of hits returned by the catalog as an additional feature. A total of six features thus model the returned item's fit to the query.

Note that these features are created on a per search item basis, not a per query basis. We need to propagate appropriate values to the search queries. We take the maximum value of each feature over all of the returned search results as the value for the query feature. This is because a single search item that matches well with the query is likely to be a plausible known item and thus favor the classification of the query as a known item search.

5.3 Methods and Experiments

For our experiments, we repeat the same basic procedure as in the original query-only judgment task. Specifically, we first process the same 320 annotated queries and their corresponding search results to derive the maximal matching features values for each query. We combine the resulting feature vector with the participants' labeling on the second query judgment task. Again, using ten-fold cross validation and using both ME and decision tree learning to learn models and evaluate them. As in the previous case, the majority baseline does not show any correlation with human judgments (R = 0). Correlation is shown in Table 4, where "MTscores" denotes the two machine translation score based features, "other" denotes the other features based on our own manual analysis.

Table 4: Correlation on the query task using search results.

Configuration	9 point Likert	Two Class
Majority Baseline	0	0
Inter-judge Agreement	.664	.656
J48 (BF + MTscores)	.297	.261
J48 (BF + others)	.333	.224
J48 (BF + both)	.366	.285
J48 (BF + BLM + both)	.374	.269
J48 (BF + BBLM + both)	.305	.257
SVM (BF + MTscores)	.372	.320
SVM (BF + others)	.409	.333
SVM (BF + both)	.411	.334
SVM (BF + BLM + both)	.454	.351
SVM (BF + BBLM + both)	.447	.367

As the scores on this task are generally higher than in the previous judgment task without the query results, this validates that the search result information does make the task easier. We can see that both the machine translation metrics and the other matching features both help to improve classification accuracy. It is also encouraging to see that their combination reinforces each other, leading to better performance. In this task, we see that the results clearly favor the SVM learner, with best scoring configurations using both sets of query result features and language modeling, garnering .454 / .664 = 68% and .367 / .656 = 55% of the level of human performance on the 9-point Likert and two class problems, respectively. Although the percentage performance is lower than the previous task, absolute performance is increased by using search results: by 3% and 30% in the 9-point Likert and two class problem, respectively.

An analysis of the J48 classifier reveals that the both the machine translation and the manually distilled search result features are the most important features for classification. The language model feature is less useful, only used in a few branches of the final tree. This correlates with our expectations, as the addition these features increases absolute correlation performance.

6. CLASSIFYING SEARCH RESULTS

The final task is to identify known items from the returned search results. As the features that distinguish the known items from other returned results are largely shared with the query re-judgment task, we can perform these two classifications in parallel. We use the search result matching features introduced in the previous re-judgment task but calculated on a per-search

result basis, rather than on a query basis. We also introduce features that normalize the results compared to the maximal match for the query. For example, say a query returns 10 results, and that the maximum BLEU score for any of the 10 results is .5. The BLEU score for a particular search result relevant to that query might be .2. In this case, we add the raw BLEU score as a feature and its normalized BLEU score (.2/.5) = .4 as an additional feature. We calculate these scores for all 1,500 annotated search result instances to perform cross validation, as noted in Section 4.1. Table 5 shows the correlation of the different configurations to the annotations.

Table 5: Correlation on the search result classification task.

Configuration	9 point Likert	Two Class
Majority Baseline	0	0
Inter-judge Agreement	.763	.750
J48 (MTscores)	.632	.607
J48 (others)	.655	.648
J48 (both)	.726	.685
SVM (MTscores)	.519	.562
SVM (others)	.524	.560
SVM (both)	.609	.639

In this task, the decision tree classifier outperforms the SVM based classifier in every configuration. Similar to the query re-judgment task, we see that the machine translation features are correlated with the task but not as strongly as our manually distilled features. The two sets of features are complementary and reinforce each other to yield the best performance. In this case, the decision tree classifier yields a performance that is .726 / .763 = 95% of the level of performance of human participants on the raw 9-point scale.

7. USING THE CLASSIFIERS

Having known item query and title classifiers allows for more sophisticated query handling in a library catalog. For example, if a user's query is clearly a known item query and produces no results in the catalog, it is clear that the item is not present in the library catalog. Instead of ending the search in frustration for the user, the OPAC can cooperatively help by providing useful alternatives. These would include recommending the user to try external catalogs (*e.g.*, interlibrary loan) or recommending that the library consider purchasing the title. Note that these actions are specific to known item queries, not unknown item and area searches, because the search implies that the item does exist but is not owned by the library. Conversely, when the query is a known item query and the catalog does retrieve it, we can zoom directly to title's entry rather than waste the user's time with a display of the remaining search results.

As important is the opportunity to teach the user how to use the catalog more effectively. We can automatically "rewrite" any known query to use advanced operators (*e.g.*, "t:" in INNOPAC to match only title fields) to teach users how to utilize the operators to make more precise searches. An alternative is to direct users to alternative title, author and subject query interfaces. Either way enhances the literacy rate of OPAC users in a cooperative manner that does not require explicit demonstration and time commitment, unlike formal training.

These possibilities are shown in screen mock-ups in Figure 2. We display these screen prototypes to show the spirit of the cooperative model and not as finished products; suitable user interface engineering and design would need to be done to tune the screens appropriately.

As our models include different levels of confidence though the use of the raw 9-point Likert scores, we can have different models of interaction for different levels of confidence. For example, if the system suspects that a title is the known item that the user is searching for but is not certain, it can re-order the search results to place the title in the first position or place circulation information as part of the search results. Availability / circulation information is particularly important in known item search, as it is likely that the user will want to access or borrow the title if it is available.

8. CONCLUSION

Automatic query analysis in general is gaining in interest recently. The IR community has mostly focus on topical search in the past decade with TREC work, although there have been some attempts in categorizing known item search in the outputs of TREC related research as well [10][12]. Recently, work in the web search community has also called attention to query analysis, as query logs show that queries are more heterogeneous than just topical search alone [16][17].

Known item search is an important factor in both automated library catalogs and digital libraries. An example from [5] cites researchers ourselves: when looking for references and citations, expert researchers tend to look for specific papers, authors or research groups, rather than do topical search. While known item search has been shown to be the most straightforward and simple type of search [17], this does not mean we cannot improve upon this and that they are unimportant. On the contrary: as known item queries represent a sizeable percentage of catalog queries and can easily be resolved, providing cooperative handling for both successful and failed known item queries can lead to better usability of library catalogs.

Our study has investigated how a system can automatically determine whether a query is a known item query, and whether the known item(s) exist in a library's holdings. We have analyzed both known item queries and the titles retrieved by them in a large academic catalog. From this data, we have distilled a number of characteristics that can distinguish known items queries from other queries, and identify when the known item is correctly retrieved by the catalog. We have demonstrated that an automatic approach can be constructed and achieves up to 85% and 95% correlation with human annotations, for each of the two tasks respectively. Our study also indicates that knowledge of the queries' search results makes the query classification task more consistent and polarizes judgments among human annotators. This source of information can assist the automatic classifier as well improving performance over approaches that examine the query alone.

To our knowledge, this is the first work that seeks to characterize known item queries and identify their targets in an automated manner. Kilgour [7] prescribes how informed library users can issue effective known item queries, by including the author's surname and specific words from the title of the item. However, many users will not have the luxury or inclination to learn best

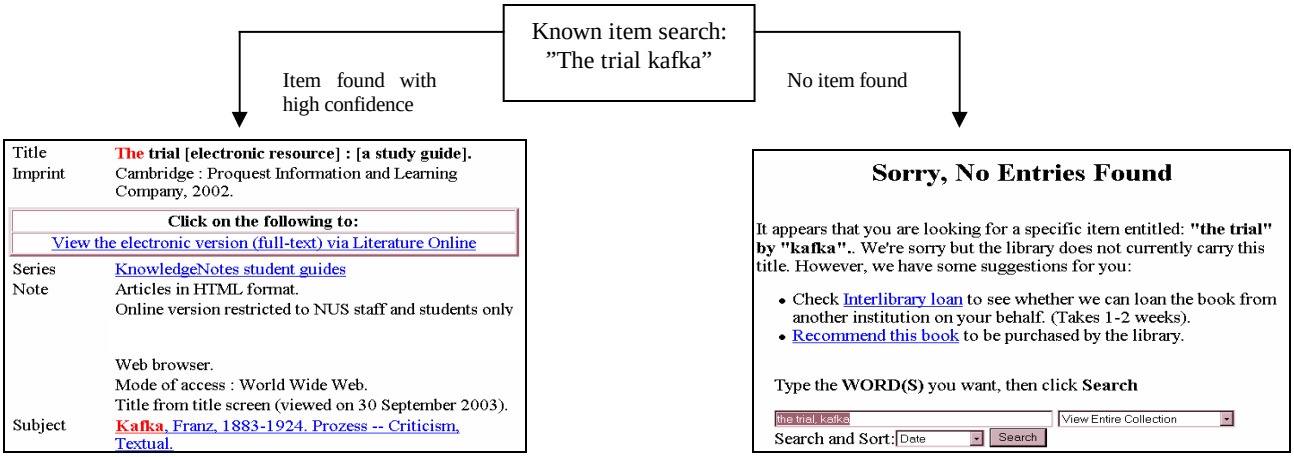

Figure 2: Prototype cooperative user interface for known item query handling, in successful (l) and unsuccessful (r) cases.

practices. As such, we see our work as complementary to Kilgour's in tackling the same problem – support of known item queries – from two different perspectives.

We have focused on cooperative strategies for OPAC searching. As patrons are increasingly using a single search text box as the sole user interface to a DL, we need to expend more efforts into intelligently process their queries and to assist them with their searches. Our work here is one step in this direction. In future work, we plan to examine the effects of using a larger collection of known item titles (e.g., *Books in Print*) to further improve the query classifier. We also plan to study author searchers in greater detail as this type of query also plays a large role in OPAC searches.

9. REFERENCES

[1] B. Allen. Recall Cues in Known-Item Retrieval. *J. Amer. Society for Info. Science.* 40(4). 1989. pp. 246-252

[2] M. J. Bates. Where should the person stop and the information search interface start? *Info. Proc. And Management*. 26(5) 1990. pp. 575-591.

[3] D. Harman (ed.) *The Fourth Text REtrieval Conference (TREC-4).* NIST Special Publication 500-236. 1995.

[4] Innovative Interfaces – Innopac <http://www.iii.com/>

[5] K. Jarvelin and P. Ingwersen. Information seeking research needs extension towards tasks and technology. *Info. Research.* 10(1). 2004.

[6] F. Jelinek. *Statistical Methods for Speech Recognition.* Bradford Books, 1998.

[7] F. G. Kilgour. Known-Item Online Searches Employed by Scholars Using Surname plus First, or Last, or First and Last Title Words, *J. Amer. Society for Info. Sci. and Tech.* 52(14). 2001. pp. 1203-1209.

[8] F. G. Kilgour, B. B. Moran and J. R. Barden. Retrieval Effectiveness of Surname-Title-Word Searches for Known Items by Academic Library Users. *J. Amer. Society for Info. Science.* 50(3). 1999. pp. 265-270.

[9] R. Larson. The Decline of Subject Searching: Long-Term Trends and Patterns of Index Use in an Online Catalog. *J. of Amer. Society for Info. Science.* 42(3). 1991. pp. 197-215

[10] M.-K. Leong. Concrete Queries in Specialized Domains: Known Item as Feedback for Query Formulation.

[11] National Institute of Science and Technology. Readme documentation for the MT evaluation kit, version 10. 2003. Available from: http://www.nist.gov/speech/tests/mt/resources/scoring.htm

[12] P. Ogilvie and J. Callan. Combining Document Representations for Known-Item Search. In *Proc. Of SIGIR '03.* 2003.

[13] K.A.Papineni, S. Roukos, T. Ward and W. J.Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL '02*. 2002.

[14] F. Peng, D. Schuurmans and S. Wang. Language and Task Independent Text Categorization with Simple Language Models. In *Proc. of HLT-NAACL '03*, 2003. pp. 110-117.

[15] A. Ratnaparkhi. *A Maximum Entropy Part-Of-Speech Tagger*, In *Proc. of the Empirical Methods in Natural Language Processing Conference (EMNLP '96).* Pennsylvania, USA. 1996.

[16] D. E. Rose and D. Levinson. Understanding user goals in web search. In *Proc. Of WWW '04*. 2004. pp. 13-19.

[17] A. Spink, D. Wolfram, M. Jansen and T. Saracevic. Searching the Web: The Public and Their Queries. *J. of Amer. Society for Info. Sci. and Tech.* 52 (3). 2001. pp. 226-234.

[18] D. J. Slone, Encounters with the OPAC: On-line Searching in Public Libraries, *J. of Amer. Society for Info. Sci. and Tech.* 51 (8). 2000. pp. 757-773.

[19] I. H. Witten and F. Eibe. Data Mining: Practical machine learning tools with Java implementations, Morgan Kaufmann, San Francisco. 2000.