Discovering Key Concepts in Verbose Queries

Michael Bendersky
bemike@cs.umass.edu

W. Bruce Croft
croft@cs.umass.edu

Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

ABSTRACT

Current search engines do not, in general, perform well with longer, more verbose queries. One of the main issues in processing these queries is identifying the key concepts that will have the most impact on effectiveness. In this paper, we develop and evaluate a technique that uses query-dependent, corpus-dependent, and corpus-independent features for automatic extraction of key concepts from verbose queries. We show that our method achieves higher accuracy in the identification of key concepts than standard weighting methods such as inverse document frequency. Finally, we propose a probabilistic model for integrating the weighted key concepts identified by our method into a query, and demonstrate that this integration significantly improves retrieval effectiveness for a large set of natural language description queries derived from TREC topics on several newswire and web collections.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query Formulation

General Terms

Algorithms, Experimentation, Theory

Keywords

Information retrieval, verbose queries, key concepts extraction

1. INTRODUCTION

Automatic extraction of concepts of interest from a larger body of text have proved to be useful for summarization [16], keyword extraction [15], content-targeted advertising [33], named entity recognition [4] and document clustering [11]. In this paper, we describe an extension of automatic concept extraction methods for the task of extracting key concepts from verbose natural language queries.

Information retrieval research is generally more focused on *keyword queries*: terse queries that contain only a small selection of key words from a more verbose description of the actual information need underlying the query. TREC topics illustrate the difference between a keyword query and a description query. A TREC topic consists of several parts, each of which corresponds to a certain aspect of the topic. In the example at Figure 1, we consider the title (denoted <*title*>) as a keyword query on the topic, and the description of the topic (denoted <*desc*>) as a natural language description of the information request.

```
<num> Number 829
<title> Spanish Civil War support
<desc> Provide information on all kinds of material
international support provided to either side in the
Spanish Civil War.
```

Figure 1: An example of <*title*> and <*desc*> parts of a TREC topic.

It might appear obvious to the reader that the key concept of the topic in Figure 1 is *Spanish Civil War*, rather than, say, *material international support*, which only serves to complement the key concept. However, there is no explicit information in the description itself to indicate which of these concepts is more important.

A simple experiment illustrates this point. When running the <*desc*> query from Figure 1 on three commercial web search engines, the first page of the results (top ten retrieved documents) for each of the search engines contains six, four and zero documents related to the Spanish Civil War, respectively. Only one of the search engines returns documents mentioning international support during the war. In contrast, running the <*title*> query from Figure 1 results, for all three search engines, in all the documents returned on the first page referring to some aspect of Spanish Civil War, including international support during the war.

A verbose query could also potentially contain two or more equally essential key concepts. For example, consider a query *What did Steve Jobs say about the iPod?*[1], which contains two key concepts, *Steve Jobs* and *iPod*, that must

[1]This example originally appeared on the Powerset blog: http://blog.powerset.com/

SIGIR'08, July 20–24, 2008, Singapore.

	ROBUST04	W10g	GOV2
<*title*>	25.28	19.31	29.67
<*desc*>	24.50	18.62	25.27

Table 1: Retrieval effectiveness comparison (mean average precision) for <*title*> and <*desc*> queries on several TREC collections.

be somehow related in a retrieved document in order for it to be relevant. When examining the top ten documents retrieved by three commercial web search engines in response to this query, we note that some of them contain only information about a single key concept (e.g., a magazine article "What Did the Professor Say? Check Your iPod"), while others contain both concepts, but with no explicit relation between them (e.g., a blog entry "iPod Giveaway: Design a Steve Jobs Movie Poster"). In contrast, when examining the top ten documents retrieved in response to a keyword query *"steve jobs"+iPod*, we note that most of them discuss Steve Jobs in some relation to the iPod (e.g., a link to a video documenting an iPod introduction by Steve Jobs, which did not appear on the first page of results for the more verbose query).

Our goal in this paper is to overcome the difficulty of key concepts detection in verbose natural language queries. We hypothesize that the identification of the key query concepts will have a significant positive impact on the retrieval performance for verbose queries (such as <*desc*> queries), which often mix several key and complementary concepts, as discussed in the above examples. Treating all query concepts equally causes loss of focus on the main topics of the query in the retrieval results, e.g., returning documents that discuss *material international support*, but not the *Spanish Civil War*.

This loss of focus causes a paradoxical situation in that a keyword query that provides *less* information about the topic than its more verbose natural language counterpart attains *better* retrieval effectiveness. Indeed, when comparing the effectiveness of the retrieval using either <*title*> or <*desc*> query types, we note that <*title*> queries consistently perform better on a variety of TREC collections (see Table 1).

In this paper we: (i) present a general probabilistic model for incorporating information about key concepts into the base query, (ii) develop a supervised machine learning technique for key concept identification and weighting, and (iii) empirically demonstrate that our technique can significantly improve retrieval effectiveness for verbose queries.

2. MODEL

In this section we present our model of key concept selection for verbose queries. We start by developing a formal probabilistic model for the utilization of key concepts for information retrieval. We then proceed to detail the supervised machine learning technique used for key concept identification and weighting.

2.1 Ranking Principle

We start by ranking a document d in response to query q by estimating the probability $p(q|d)$, which is a standard approach in the *language modeling* retrieval model [25, 10]. Next, we consider *all* possible implicit concepts c_i (we defer the treatment of concept identification to Section 2.2) that could potentially generate the actual query q, and get that

$$p(q|d) = \sum_i p(q|d, c_i)p(c_i|d). \quad (1)$$

A common way to estimate a joint conditional probability is using a linear interpolation of the individual conditional probabilities [20, 18, 30]. Accordingly, we use a linear interpolation of $p(q|d)$ and $p(q|c_i)$ to estimate $p(q|d, c_i)$. Applying some probability algebra, we can rank a document d in response to a query q using an estimate

$$rank(d) = \lambda' p(q|d) + (1 - \lambda') \sum_i p(q|c_i)p(c_i|d),$$

where λ' is a free parameter in $[0, 1]$. Assuming a uniform distribution for both $p(q)$ and $p(c_i)$ (which is reasonable, given no prior knowledge), the above is rank-equivalent to

$$rank(d) \propto \lambda p(q|d) + (1 - \lambda) \sum_i p(c_i|q)p(c_i|d), \quad (2)$$

where λ is normalized such that $\lambda = \frac{\lambda'}{\lambda' + \frac{p(q)}{p(c_i)}(1-\lambda')}$.

We note that Equation 2 takes a general form, where each document is ranked according to the combination of its probability of generating the query itself, and a weighted sum of its probabilities of generating each implicit concept c_i. The concept weights are determined by how well they "represent" the query q. As using all possible implicit concepts for ranking a document is infeasible, and moreover the probability $p(c_i|q)$ will be close to zero for all but very few query-related concepts, one may approximate the ranking above by using only a fixed number of concepts with the highest weights. Thus, Equation 2 may be interpreted as a *query expansion* technique such as local and global document analysis [32], latent concept expansion [23] or query expansion using random walk models [9] among others.

Alternatively, we may only consider the explicit concepts, i.e., the concepts that appear in the actual query q. This is the approach we take in this paper, as we are interested in discovering key concepts in verbose natural language queries. Thus, Equation 2 at above reduces to

$$rank(d) \propto \lambda p(q|d) + (1 - \lambda) \sum_{c_i \in q} p(c_i|q)p(c_i|d). \quad (3)$$

We use Equation 3 for examining the benefits for retrieval performance attained by our technique for concept identification and weighting discussed in the following sections.

2.2 Concept identification

We note that the ranking principle presented in Section 2.1 is not committed to any particular definition of a concept. A concept can be a single word, an idiom, a restricted collocation or a free combination of words [3]. With this in mind, we use noun phrases extracted from the queries as concepts. Noun phrases have proven to be reliable for key concept discovery in some past work on information retrieval [32, 7, 2] and natural language processing [15], and are flexible enough to naturally distinguish between words, collocations, entities and personal names among others. For instance, the description query presented in Figure 1 can be split into the following noun phrases: *[information, kinds, material international support, side, Spanish Civil War].*

Feature Name	Feature Description
$is\_cap(c_i)$	A Boolean indicator. Set to TRUE iff all the concept words are capitalized.
$tf(c_i)$	Concept term frequency in the corpus.
$idf(c_i)$	Concept inverted document frequency in the corpus.
$ridf(c_i)$	Concept residual inverted document frequency in the corpus.
$wig(c_i)$	Concept weighted information gain.
$g\_tf(c_i)$	Concept term frequency extracted from Google n-grams counts [5].
$qp(c_i)$	Number of times a concept was used as a part of a query, extracted from a large query log.
$qe(c_i)$	Number of times a concept was used as an exact query, extracted from a large query log.

Table 2: A summary of features used as an input for the concept classification task.

These phrases represent the different aspects present in the actual query.

2.3 Concept weighting

This section focuses on concept weighting. Given Equation 3, we treat the probability $p(c_i|q)$ as a weight assigned to each query concept that reflects how well concept c_i represents the query at hand.

Instead of estimating $p(c_i|q)$ directly, we take a different approach, which allows us to leverage some non-query specific information towards key concept detection. We make the following assumptions:

Assumption A. Each concept c_i can be assigned to one of the mutually exclusive classes: **KC** (key concepts class) or **NKC** (non-key concepts class)

Assumption B. A global function $h_k(c_i)$ exists, such that $h_k(c_i)$ indicates the "confidence" that concept c_i belongs to class **KC**[2].

Following the assumptions at above, we use a normalized variant of $h_k(c_i)$, to derive an estimate

$$\hat{p}(c_i|q) = \frac{h_k(c_i)}{\sum_{c_i \in q} h_k(c_i)}. \quad (4)$$

In other words, concepts for which we have the highest confidence in membership in class **KC** are regarded as the best "representations" for the query among other query concepts.

There are several well-known word weighting techniques we could use for deducing $h_k(c_i)$. However, instead of giving a preference to any single technique, we choose a supervised machine learning approach, and use different weightings as input features for the weight assignment algorithm. Similar approaches were shown to work well in some previous work on keyphrase detection [12, 29, 15].

Formally, we consider a training set of labeled instances

$$(\mathbf{x}_1, l_1), \ldots, (\mathbf{x}_n, l_n),$$

where $\mathbf{x}_i$ is a feature vector representing the concept c_i, and l_i is a binary label, indicating whether $c_i \in$ **KC**. Given the training set, we seek to learn a ranking function of the form $h_k \colon \mathbf{X} \rightarrow \mathbb{R}$, such that $h_k(\mathbf{x}_i) > h_k(\mathbf{x}_j)$ entails that concept c_i has a higher confidence in membership in class **KC** than concept c_j. Once the learning process completes, we can proceed to directly use $h_k(\mathbf{x}_i)$ to calculate an estimate $\hat{p}(c_i|q)$, as shown in Equation 4.

[2]A possible interpretation for $h_k(c_i)$ is a conditional probability $p(\mathbf{KC}|c_i)$, but we do not require $h_k(c_i)$ to be a proper probability function.

What is left in order to complete the derivation of the machine learning method is the feature generation process for each instance $\mathbf{x}_i$. We use a mix of both novel weighting features and weighting features used in previous work on keyphrase detection [12, 29, 15] to represent each concept. Table 2 presents the summary of the features used for concept weighting. More detailed explanation of each feature is given below.

Features.

$is\_cap(c_i)$ This feature is a Boolean indicator that is set to TRUE iff all the concept words are capitalized (e.g. *International Criminal Court*).

$tf(c_i)$ Concept *term frequency* in the corpus. We assume that key concepts will tend to have a higher term frequency than non-key concepts.

$idf(c_i)$ Concept *inverse document frequency* in the corpus. Inverse document frequency is commonly used in information retrieval as a weighting function [27]. The form of IDF used in this paper is

$$idf(c_i) = \log_2 \frac{N}{df(c_i)},$$

where N is the number of documents in the corpus and $df(c_i)$ is concept *document frequency*.

$ridf(c_i)$ *Residual IDF* is the difference between the observed IDF and the value predicted by a Poisson model [8]:

$$ridf(c_i) = idf(c_i) - \log_2 \frac{1}{1 - e^{\theta_i}},$$

where $(1 - e^{\theta_i})$ is the probability of a document with at least one occurrence of concept c_i under a Poisson distribution, and $\theta_i = \frac{tf(c_i)}{N}$ (the average number of occurrences of c_i per document). Residual IDF is based on the assumption that the Poisson distribution only fits the distribution of non-content concepts [8]. Therefore, deviation from Poisson model is potentially a good predictor of the key concepts.

$wig(c_i)$ *Weighted Information Gain* (WIG) [34] measures the change in information about the quality of the retrieval (in response to concept c_i) from a state where only average document is retrieved to a state where the actual results are observed. In our paper, a normalized version of WIG is used

$$wig(c_i) = \frac{\frac{1}{M} \sum_{d \in T_M(c_i)} \log p(c_i|d) - \log p(c_i|\mathcal{C})}{-\log p(c_i|\mathcal{C})},$$

where $T_M(c_i)$ is a set of top M documents[3] retrieved in response to c_i from collection $\mathcal{C}$, and $p(c_i|\cdot)$ is calculated using a maximum likelihood estimate [25]. WIG was found to have a positive correlation with the retrieval performance [34], and we hypothesize that it serves as an indicator of key concepts, as using such concepts as queries should provide a cohesive set of top retrieved documents.

$g\_tf(c_i)$ We use the *Google n-grams* [5] (English word n-grams frequency counts generated from approximately 1 trillion word tokens of text from publicly accessible Web pages) to estimate the concept term frequency in a large web collection. We expect these counts to provide a more accurate frequency estimator, especially for smaller corpora, where some concept frequencies may be underestimated due the collection size.

$qp(c_i)$, $qe(c_i)$ We use a large query log consisting of approximately 15 million queries[4] to estimate the concept *query frequency*. We extract two features for each concept c_i: $qp(c_i)$ indicates how many times a concept c_i was used *as a part* of some query in the log, and $qe(c_i)$ indicates how many times an *exact concept* c_i was used as a query in the log. We assume that for the key concepts the ratio $\frac{qe(c_i)}{qp(c_i)}$ will be higher than for the non-key concepts.

3. RELATED WORK

Using supervised machine learning techniques for an automatic extraction of key concepts from documents was first proposed by Turney [29], and later explored by several other researchers [12, 15]. Similar machine learning techniques have also proved beneficial for other tasks such as named entity recognition [4], content-targeted advertising [33] and summarization [16].

Key concept detection in verbose queries has been a subject of some previous work in information retrieval. Allan et al. [2] use a set of linguistic and statistical methods and a proximity operator to discover *core terms* in <*desc*> queries. According to Allan et. al [2], a core term is a term that *must* be present in a document for the document to be relevant. Callan et al. [7] use noun phrases, named-entities recognition, exclusionary constraints and proximity operators to convert <*desc*> queries into structured INQUERY queries. Although similar to our work, the key concepts extraction and weighting techniques discussed in these papers are focused on queries derived from TREC topics (e.g., removing "stop-phrases" common in TREC queries, or assigning higher weights to concepts that appear in the title of the topic), while our key concept extraction method is more general and is not biased towards any specific query type.

Some previous work on query expansion [32, 6, 9] uses expansion term groupings to balance the various aspects of the original query in the expanded query. Single terms [6], noun phrases [32] or n-grams [9] are used to determine query aspects. These methods, however, do not assign explicit weights to aspects.

Recent work by Kumaran and Allan [17] addresses the issue of extracting the optimal (in terms of retrieval effectiveness) *sub-query* from the original long query. Their approach involves extracting a short list of candidate sub-queries using the mutual information measure and presenting this list to the user, allowing her to replace the original query by one of the candidates from the list. This approach resulted in significant improvements over retrieval with the original <*desc*> queries.

In most of the previous work on retrieval the differentiation between key and non-key concepts is done via statistical methods such as term and document frequency weighting [27] or term-specific smoothing [14, 21]. However, these methods are usually based on single word collection statistics, and do not always capture the key *concepts*, rather than words. In contrast to previous work, we do not constrain ourselves to a specific weighting scheme to detect the key query concepts. Instead, we adopt the supervised machine learning approach [29, 12, 15] for key query concept extraction, and use a diverse mix of features as inputs for the supervised machine learning algorithm.

4. EXPERIMENTAL RESULTS

In this section, we describe the experimental results of our work. Section 4.1 focuses on the concept classification and weighting experiments, Section 4.2 is dedicated to the analysis of the features used in the machine learning algorithm, and Section 4.3 details the retrieval experiments based on the weighted concepts.

We provide a summary of the corpora used for our experiments in Table 3. We note that collections vary both by type (ROBUST04 is a newswire collection, while W10g and GOV2 are web collections), number of documents and number of available topics, thus providing a diverse experimental setup for assessing the robustness of our classification, weighting and retrieval methods.

Name	# Docs	Topic Numbers
ROBUST04	528,155	301-450, 601-700
W10g	1,692,096	451-550
GOV2	25,205,179	701-850

Table 3: Summary of TREC collections and topics used in Section 4

Noun phrases were extracted using MontyLingua natural language processing tool [19]. All concept classification and weighting experiments were performed using the algorithms implemented in Weka [31], a collection of machine learning algorithms for data mining tasks. Indexing and retrieval was done using Indri [28], which is a part of the Lemur Toolkit [24]. All indexes and topics were stopped using a standard INQUERY stopwords list [1] and stemmed using a Porter stemmer [26]. Either <*desc*> or <*title*> portions of the TREC topics were used to construct the queries. In all retrieval experiments, Dirichlet smoothing with $\mu = 1500$ is used.

4.1 Concept Classification Results

In order to assess the effectiveness of the supervised machine learning approach outlined in Section 2.3, we employ the *AdaBoost.M1* meta-classifier with *C4.5 decision trees* as base learners [13]. AdaBoost.M1 "boosts" the repeated runs $1, \ldots, T$ of the base learners on various distributions over the training data into a single composite learner, which is

[3]In our experiments $M = 50$.

[4]Live Search 2006 search query log excerpt.

often more effective than using any of the individual base learners. The AdaBoost.M1 method was selected for several reasons. First, it consistently outperformed other classification methods such as C4.5 decision tree or Naive Bayes classifier in the preliminary experiments we have conducted. Second, its output for a single input instance $\mathbf{x}_i$ can be interpreted not only as a binary classification decision ($c_i \in \mathbf{KC}$ or $c_i \in \mathbf{NKC}$), but also as a weighted combination of base hypotheses $\sum_{j=1,\ldots,T} w_j h_j(\mathbf{x}_i)$ [13]. This combination naturally translates into a confidence function $h_k(c_i)$, presented in Section 2.3.

In the first suite of experiments, we examine how well our proposed approach separates the key and the non-key concepts. To this end, all the noun-phrases are extracted from the <*desc*> queries, and a single noun phrase, which is deemed to be the most suitable candidate, is selected as a key concept for each query (e.g., for the example at Figure 1, *Spanish Civil War* is selected as a key concept).

The AdaBoost.M1 classifier is trained using a set of labeled instances. At the test phase, for each tested query, the extracted noun phrases are ranked according to their confidence level of belonging to class **KC** — $h_k(c_i)$. The highest ranked noun phrase is then selected as a key concept for the query.

We note that although some queries might contain more than one key concept, selecting a single key concept per query has two important advantages. First, it potentially simplifies the manual key concept selection task, as the assessor is not required to determine the optimal number of key concepts for each query. Second, it establishes a *lower bound* on the accuracy of the key concept selection process defined above, as it provides a minimal amount of training data.

We use a cross-validation approach; for each of the collections, queries are divided into subsets of 50 queries each. Each subset, in turn, is used as a test set, while the rest of the queries serve as a labeled training set. Experiments are run separately for each collection, and average results over all test sets are reported.

Our second experiment suite is designed to test whether our key concept classification approach indeed outperforms a simple non-supervised weighting approach where $idf(c_i)$ weights are directly used to rank the extracted noun phrases, and, as before, the highest ranked noun phrase is selected as a key concept for the query.

Table 4 reports the *accuracy* and the *mean reciprocal rank* (MRR) results when either AdaBoost.M1 or $idf(c_i)$ ranking are used for key concept classification. Accuracy is simply the percentage of the correctly identified key concepts. MRR is the mean of the reciprocal ranks at which the key concepts were returned. A higher MRR score indicates a higher confidence in membership in **KC** class for key concepts, compared to other concepts in the query.

Table 4 shows that for all the tested collections AdaBoost.M1 outperforms $idf(c_i)$ ranking. We note that the difference in the performance is inversely proportional to the collection size, which confirms our hypothesis (see Section 2.3) that for smaller corpora, features other than collection term and document counts should be used in order to prevent concept frequency underestimation.

Misclassification case analysis.

In a posterior analysis, we discovered that most of the classification errors were a result of an ambiguity in a key concept selection process. E.g., for the description query *How are pets or animals used in therapy for humans and what are the benefits?* (topic 794), the *therapy* concept was marked as a key concept, while the classification algorithm assigned the highest rank to *pets or animals* concept, and the *therapy* concept was ranked second. Clearly, neither of the concepts describes the query in its entirety, and a key concept is better expressed by a combination of the two (which is reflected by the title of the topic *pet therapy*).

Multiple key concepts were more common in the ROBUST04 collection than in the other two collections, as its topics tend to contain more verbose and grammatically complex description queries (e.g., consider the description query *A relevant document would discuss how effective government orders to better scrutinize passengers and luggage on international flights and to step up screening of all carry-on baggage has been.* (topic 341), for which the concept *international flights* was marked as a key concept for a lack of a better choice). This might offer an explanation to the fact that the classification accuracy is lower for ROBUST04, compared to the GOV2 and W10g collections.

Another common issue is the case when a true key concept is "masked" by a non-key concept exhibiting the traits of a key concept. For example, in the description query *What violent activities have Kurds, or members of the Workers Party of Kurdistan (PKK), carried out in Germany?* (topic 611), a key concept *Kurdistan PKK* is masked by a strong collocation *violent activities*.

In Section 4.3 we use the fact that although we do not attain a perfect accuracy in our classification experiments, the confidence levels assigned to each concept are generally reliable, and integrate the weighted concepts into the original <*desc*> query to improve the retrieval performance.

	AdaBoost.M1		**idf(c_i)**	
	Acc	**MRR**	**Acc**	**MRR**
ROBUST04	73.20	83.68	56.40	74.22
W10g	81.00	85.33	66.00	78.58
GOV2	82.67	88.00	74.67	85.67

Table 4: Comparison of accuracy and MRR results for ROBUST04, W10g and GOV2 collections, when using either the AdaBoost.M1 algorithm with the features detailed at Table 2, or a single $idf(c_i)$ feature for concept classification.

4.2 Feature Analysis

In this section, we analyze the utility of the various features used in the classification task described at above (refer back to Table 2 for the summary of the features).

We repeat the key concept classification experiments using AdaBoost.M1, as described in the previous section, while varying the features. In each iteration, we remove a single feature from the full feature set. Assuming independence between the different features, a decrease in accuracy indicates how much does the removed feature contribute to the overall accuracy reported in Table 4. Table 5 reports the feature analysis experiments results.

We note that feature contribution to the overall accuracy varies in different collections. For GOV2, a large web collection, external features such as $g\_idf(c_i)$, $qp(c_i)$ and $qe(c_i)$

	$is\_cap(c_i)$	$tf(c_i)$	$idf(c_i)$	$ridf(c_i)$	$wig(c_i)$	$g\_tf(c_i)$	$qp(c_i)$	$qe(c_i)$
ROBUST04	-6.00	-1.60	-2.40	+3.20	**-7.60**	-4.80	-6.40	-2.40
W10g	-5.00	-6.00	**-10.00**	-8.00	-8.00	-4.00	-8.00	-7.00
GOV2	+1.32	**-5.34**	-1.34	+1.32	-1.34	-0.00	-3.34	-0.67

Table 5: Feature analysis. % of change in accuracy when a single feature is removed. Negative value for a feature indicates that accuracy has decreased after feature removal and vice versa. Features that contribute the most to classification performance are marked in bold.

	AdaBoost.M1	
	Acc	**MRR**
ROBUST04	76.40	84.54
W10g	81.00	85.33
GOV2	84.00	88.88

Table 6: Accuracy and MRR results for ROBUST04, W10g and GOV2 collections, when using the AdaBoost.M1 algorithm with the subset of features that attains the best classification accuracy according to Table 5. Feature $ridf(c_i)$ is not used for classification on collections ROBUST04 and GOV2.

have little or no impact on the overall accuracy, while accuracy on ROBUST04 and W10g, which are smaller, degrades with the removal of these features. On the other hand, collection-dependent features such as $tf(c_i)$, $idf(c_i)$ and $ridf(c_i)$ have little positive (or negative, as in the case of $ridf(c_i)$) impact on the overall accuracy for the ROBUST04, the smallest collection among the three. Surprisingly, both $is\_cap(c_i)$ and $ridf(c_i)$ have a negative, albeit small, impact on the overall accuracy for a GOV2 collection.

Following the analysis at Table 5 we remove the feature $ridf(c_i)$ from the feature sets for collections ROBUST04 and GOV2, and report the classification accuracy and MRR attained by using an optimal combination of features for all the collections in Table 6.

4.3 Retrieval Results

In this section, we explore the retrieval benefits of using the concept classification and weighting technique discussed in the previous sections. We treat the normalized concept confidence function $h_k(c_i)$ obtained by the concept ranking by AdaBoost.M1 presented in Section 4.1 as an estimate for the probability $\hat{p}(c_i|q)$, and rank documents according to Equation 3. (In this, and all the subsequent experiments, the optimal feature combination as detailed in Table 6 is used for concept classification and ranking.)

First, we calibrate the number of weighted concepts added to the base <*desc*> query. The results are presented in Figure 2. We note that an addition of the single highest ranked concept to the original query, i.e., the integration of a concept identified as a key concept in Section 4.1, provides a substantial performance boost (in terms of mean average precision) on all the tested collections. In two out of three collections, performance is further increased when a second-highest ranked concept is also integrated into the base query. We note that adding more than two concepts attains no further retrieval performance gain. This can be interpreted as the fact that, on average, the description queries used for our experiments may be expressed by a combination of two key concepts. This is evident from an example at Figure 1:

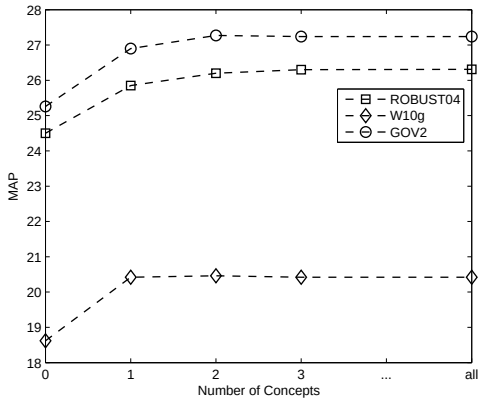

Figure 2: Changes in MAP, when varying the number of concepts added to the original <*desc*> query.

description query can be rewritten as a combination of concepts *"Spanish Civil War"+"material international support"* without noticeable loss in query expressiveness. We note, however, that although this observation holds for a large portion of tested queries, there are cases in which relying entirely on the extracted concepts and discarding the original query causes a degradation in query expressiveness. (For instance, consider the description for Topic 714: *What restrictions are placed on older persons renewing their drivers' licenses in the U.S.?* for which the two highest ranking concepts are *older persons* and *drivers license*. Clearly, these two concepts do not fully express the entire query.)

Due to the above observations, we use the combination of the base <*desc*> query and the two highest-ranked terms weighted by $\hat{p}(c_i|q)$, which results in optimal retrieval performance for most collections (see Figure 2), for our next suite of retrieval experiments. (We have also conducted experiments with the unweighted variant of this combination, but it yielded slightly inferior performance to that of the combination of the weighted concepts, and thus is not considered in the remainder of this section.)

We compare the retrieval results (in terms of *precision at 5* and *mean average precision*) obtained by this setting, denoted *KeyConcept[2]*<*desc*>, to the results obtained using either title or description query alone (denoted <*title*> and <*desc*>, respectively). In addition, we compare the effectiveness of this method to retrieval using *sequential dependency* model [22], denoted *SeqDep*<*desc*>, which integrates the base <*desc*> query with sequential bi-grams derived from query words, and uses ordered and un-ordered proximity operators. We present examples of Indri structured queries [28] representing each of these retrieval techniques

<title>	`#combine( Spanish Civil War support )`
<desc>	`#combine( information kinds material international support provided side Spanish Civil War )`
SeqDep<desc>	`#weight(0.85 #combine(information kinds material international support provided side Spanish Civil War) 0.10 #combine(#1(information kinds) #1(kinds material) #1(material international) #1(international support) #1(support provided) #1(provided side) #1(side Spanish) #1(Spanish Civil) #1(Civil War)) 0.05 #combine(#uw8(information kinds) #uw8(kinds material) #uw8(material international) #uw8(international support) #uw8(support provided) #uw8(provided side) #uw8(side Spanish) #uw8(Spanish Civil) #uw8(Civil War)))`
KeyConcept[2]<desc>	`#weight( 0.8 #combine( information kinds material international support provided side Spanish Civil War ) 0.2 #weight( 0.99994 #combine ( Spanish Civil War ) 0.00006 #combine ( material international support )))`

Table 7: Examples of the Indri query types used for the retrieval experiments detailed in Table 8. Weights for *SeqDep<desc>* were set according to the optimal setting reported in the original paper. λ for *KeyConcept[2]<desc>* (see Equation 3) was set to 0.8 across all the collections. Concept weights were assigned according to the AdaBoost.M1 algorithm output.

	ROBUST04		W10g		GOV2	
	prec@5	MAP	prec@5	MAP	prec@5	MAP
<title>	47.80	25.28	30.73_d	19.31	56.75	$\mathbf{29.67}_d$
<desc>	47.26	24.50	39.20^t	18.62	52.62	25.27^t
SeqDep<desc>	**49.11**	25.69_d	39.80^t	19.28	$\mathbf{56.88}_d$	27.53^t_d
KeyConcept[2]<desc>	48.54	$\mathbf{26.20}_d$	$\mathbf{40.40}^t$	$\mathbf{20.46}^t_d$	56.77_d	27.27^t_d

Table 8: Retrieval results for ROBUST04, W10g and GOV2 collections. Boldface indicates the best performing retrieval method per collection. Significant differences with *<title>* and *<desc>* are marked by t and d, respectively (statistical significance was measured using the two-tailed Wilcoxon test with $p < 0.05$).

in Table 7. Comparison of the retrieval results obtained by the different techniques is presented in Table 8.

We note that for all collections *KeyConcept[2]<desc>* outperforms *<desc>* retrieval both in terms of MAP and prec@5, often to a statistically significant degree. For ROBUST04 and W10g collections *KeyConcept[2]<desc>* also outperforms the *<title>* retrieval. We note, however, that for these collections *<title>* queries are more verbose than the *<title>* queries for the GOV2 collection.

Table 8 demonstrates that the retrieval performance (in terms of MAP) of *KeyConcept[2]<desc>* method is slightly inferior to the performance of *SeqDep<desc>* on GOV2 collection, and is better than the performance of *SeqDep<desc>* on ROBUST04 and W10g collections. No statistically significant differences between the retrieval effectiveness of the two methods were found.

In terms of retrieval efficiency, the queries produced by our method are more succinct (see example at Table 7), especially in the case of the verbose *<desc>* queries discussed here, than those produced by *SeqDep<desc>*. This indicates that most of the retrieval performance gain can be attributed to only a few highest ranked concepts. We also note that, compared to the sequential dependency model, we do not incorporate any proximity information in our queries, and treat our concepts as "bag of words" queries. We leave the exploration of the benefits of application of proximity operators to our retrieval methods for future work.

5. CONCLUSIONS

In this paper we address the issue of retrieval using verbose queries. We use several standard TREC collections and corresponding topics to demonstrate that the current retrieval methods perform better, on average, with keyword title queries than with their longer description counterparts.

One of the main issues in processing verbose queries is the difficulty of identifying the key concepts. This difficulty can potentially lead to a lack of focus on the main topics of the query in the retrieved document set. To this end, we propose a supervised machine learning technique for discovering key concepts in verbose queries. We detail the query-dependent, corpus-dependent and corpus-independent features used as inputs for our machine learning algorithm. We use our technique to identify and assign weights to key concepts in natural language description queries derived from TREC topics, and show that using our identification method substantially improves the average accuracy in key concept identification over the standard inverse document frequency measure, even if the size of the available training set is relatively small.

Next, we use the highest-ranked concepts for each query to improve the retrieval effectiveness of the verbose queries on several standard TREC newswire and web collections. We propose a formal probabilistic model for incorporating query and key concepts information into a single structured query, and show that using these structured queries results in a statistically significant improvement in retrieval performance over using the original description queries on all tested corpora. In some cases, our structured queries even attain a better retrieval performance than the title queries on the same topic.

The empirical results of this paper are encouraging, as using a simple query-concept combination model outperforms the original query to a statistically significant degree, and attains a comparable retrieval performance to that of using a more elaborate (and computationally more demanding)

sequential dependency model.

We believe that further gains in retrieval performance might be attained by employing additional syntactic and semantic features of natural language verbose queries and by exploring the utilization of different proximity operators for query-concept combination.

6. ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval, in part by Microsoft Live Labs, and in part by NSF grant #IIS-0534383. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

7. REFERENCES

[1] J. Allan, M.E. Connell, W.B. Croft, F.F. Feng, D. Fisher, and X. Li. INQUERY and TREC-9. *Proceedings of the Ninth Text Retrieval Conference (TREC-9)*, pages 551–562, 2000.

[2] James Allan, Jamie Callan, W. Bruce Croft, Lisa Ballesteros, John Broglio, Jinxi Xu, and Hongmin Shu. INQUERY at TREC-5. pages 119–132. NIST, 1997.

[3] L. Bentivogli and E. Pianta. Beyond lexical units: Enriching wordnets with phrasets. *Proceedings of the Research Note Sessions of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL03)*, pages 67–70, 2003.

[4] D.M. Bikel, R. Schwartz, and R.M. Weischedel. An Algorithm that Learns What's in a Name. *Machine Learning*, 34(1):211–231, 1999.

[5] Thorsten Brants and Alex Franz. Web 1T 5-gram Version 1, 2006.

[6] Chris Buckley, Mandar Mitra, Janet A. Walz, and Claire Cardie. Using clustering and superconcepts within SMART: TREC 6. *Information Processing and Management*, 36(1):109–131, 2000.

[7] James P. Callan, W. Bruce Croft, and John Broglio. TREC and tipster experiments with INQUERY. *Information Processing and Management*, 31(3):327–343, 1995.

[8] Kenneth W. Church and William A. Gale. Poisson mixtures. *Natural Language Engineering*, 1(2):163–190, 1995.

[9] K. Collins-Thompson and J. Callan. Query expansion using random walk models. *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 704–711, 2005.

[10] W. Bruce Croft and John Lafferty, editors. *Language Modeling for Information Retrieval*. Number 13 in Information Retrieval Book Series. Kluwer, 2003.

[11] J.F. da Silva, J. Mexia, C.A. Coelho, and J.G.P. Lopes. Document Clustering and Cluster Topic Extraction in Multilingual Corpora. *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 513–520, 2001.

[12] E. Frank, G.W. Paynter, I.H. Witten, C. Gutwin, and C.G. Nevill-Manning. Domain-specific keyphrase extraction. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 668–673, 1999.

[13] Y. Freund and R.E. Schapire. Experiments with a new boosting algorithm. *Machine Learning: Proceedings of the Thirteenth International Conference*, 148:156, 1996.

[14] Djoerd Hiemstra. Term-specific smoothing for the language modeling approach to information retrieval: the importance of a query term. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 35–41. ACM, 2002.

[15] A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223, 2003.

[16] Kevin Knight and Daniel Marcu. Statistics-based summarization - step one: Sentence compression. In *AAAI/IAAI*, pages 703–710, 2000.

[17] Giridhar Kumaran and James Allan. A case for shorter queries, and helping user create them. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 220–227, 2006.

[18] O. Kurland and L. Lee. Corpus structure, language models, and ad hoc information retrieval. *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 194–201, 2004.

[19] Hugo Liu. MontyLingua: An end-to-end natural language processor with common sense, 2004. Available at: web.media.mit.edu/ hugo/montylingua.

[20] X. Liu and W.B. Croft. Cluster-based retrieval using language models. *Proceedings of the 27th annual international conference on Research and developement in information retrieval*, pages 186–193, 2004.

[21] Q. Mei, H. Fang, and C. Zhai. A study of poisson query generation model for information retrieval. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 319–326. ACM, 2007.

[22] D. Metzler and W.B. Croft. A Markov random field model for term dependencies. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 472–479, 2005.

[23] D. Metzler and W.B. Croft. Latent concept expansion using markov random fields. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 311–318, 2007.

[24] P. Ogilvie and J. Callan. Experiments using the Lemur toolkit. *Proceedings of the Tenth Text Retrieval Conference (TREC-10)*, pages 103–108, 2001.

[25] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR*, pages 275–281, 1998.

[26] M. Porter. The Porter Stemming Algorithm. Accessible at http://www. tartarus. org/martin/PorterStemmer.

[27] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, 1988.

[28] T. Strohman, D. Metzler, H. Turtle, and W.B. Croft. Indri: A language model-based search engine for complex queries. *Proceedings of the International Conference on Intelligence Analysis*, 2004.

[29] P.D. Turney. Learning Algorithms for Keyphrase Extraction. *Information Retrieval*, 2(4):303–336, 2000.

[30] X. Wei and W.B. Croft. LDA-based document models for ad-hoc retrieval. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, 2006.

[31] I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

[32] J. Xu and W.B. Croft. Query expansion using local and global document analysis. *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 4–11, 1996.

[33] Wen T. Yih, Joshua Goodman, and Vitor R. Carvalho. Finding advertising keywords on web pages. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 213–222, New York, NY, USA, 2006. ACM.

[34] Y. Zhou and W.B. Croft. Query performance prediction in web search environments. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 543–550, 2007.