Automatic Discovery of Language Models for Text Databases

Jamie Callan, Margaret Connell, and Aiqun Du
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts
Amherst, Massachusetts 01003-4610, USA
callan@cs.umass.edu

Abstract

The proliferation of text databases within large organizations and on the Internet makes it difficult for a person to know which databases to search. Given *language models* that describe the contents of each database, a *database selection* algorithm such as GlOSS can provide assistance by automatically selecting appropriate databases for an information need. Current practice is that each database provides its language model upon request, but this *cooperative* approach has important limitations.

This paper demonstrates that cooperation is not required. Instead, the database selection service can construct its own language models by sampling database contents via the normal process of running queries and retrieving documents. Although random sampling is not possible, it can be approximated with carefully selected queries. This *sampling* approach avoids the limitations that characterize the cooperative approach, and also enables additional capabilities. Experimental results demonstrate that accurate language models can be learned from a relatively small number of queries and documents.

1 Introduction

The proliferation of text databases within large organizations and on the Internet can make it difficult to know which databases to search for desired information. Large corporate networks can provide access to several thousand Lotus Notes databases, Oracle databases, and other corporate document management systems, each containing many text documents. The Internet, via the WorldWideWeb, also provides access to thousands of searchable text databases. How does a person who needs information know where to search?

Content-based *database selection* algorithms such as GlOSS have been proposed as one solution [7, 6, 1, 14]. A content-based database selection algorithm (*'database selection algorithm'*) automatically ranks each text database according to its likelihood of satisfying a given query. Database selection algorithms require no manual effort, scale efficiently to millions of databases, and handle unforseen information needs. They are an appealing solution in many respects.

Database selection algorithms do not interact directly with the databases they rank. Instead, the algorithms interact with an index containing *language models* that describe the contents of each database. The contents of the language models vary according to the selection algorithm, but in general a language model describes the words or indexing terms that occur in the database, and frequency information indicating how often each term occurs.

Language models are perhaps the most essential element of database selection. Significant research activity is directed at studying what they should contain, but that is not our concern in this paper. Instead, we are interested in how a database selection service *acquires* language models for each database.

The state-of-the-art until now was for each database to provide its language model upon request. We call this the *cooperative* approach, because it assumes that all parties are willing and able to provide the information needed by other parties. The cooperative model turns out to be a weak solution in environments containing many databases managed by different parties.

This paper presents *query-based sampling*, a new method of acquiring language models that requires no special cooperation. Query-based sampling assumes only that the database selection service can run simple queries on each database to retrieve a small number of full-text documents. Language models are built automatically from these documents.

Query-based sampling was tested in a series of experiments. The results demonstrate that accurate language models can be learned, and that they can be learned with a reasonably small number of queries and documents. Query-based sampling is robust, and avoids many of the problems associated with cooperative

SIGMOD '99 Philadelphia PA

approaches. It also enables additional capabilities not supported by cooperative methods.

The next section describes content-based database selection in more detail, including the state-of-the-art in language models and their acquisition. Query-based sampling, our new approach to language model acquisition, is described in Section 3. Our experimental methodology and results are described in Sections 4, 5, and 6. Sections 7 and 8 discuss how query-based sampling enables additional capabilities related to database selection, and Section 9 concludes.

2 Content-Based Database Selection

People who use commercial search services such as Dialog, Westlaw, or Lexis-Nexis have always faced the *database selection* problem. People must either search all databases or select a subset. The simplest choice is to search *all* of the available databases, but this choice is neither available nor practical in all environments (e.g., the Internet).

Some commercial services group databases into sets with common themes, for example newspaper collections or appellate court decisions. Such grouping was a manual process for many years, but automatic methods are becoming more common [11, 2, 14]. Database grouping is an effective solution for information needs or information access patterns that can be anticipated.

When information needs are diverse, content-based database selection is a more effective solution. Content-based database selection algorithms rank databases by their similarity to a query [7, 6, 1, 14]. Each query yields a ranking that is tailored for that particular query. Typically the top n databases or all databases exceeding some threshold similarity are selected for search.

Content-based database selection algorithms are appealing because they handle a diverse range of queries, are automatic, and have moderate computational requirements (i.e., comparable to ordinary full text Information Retrieval systems). Their effectiveness has been demonstrated on $O(100)$ databases [15], and researchers are currently studying their effectiveness on $O(1,000)$ databases.

2.1 Language Models

It would be prohibitively expensive in computation and communication costs to compare each query to each database. Instead, content-based database selection algorithms compare each query to an index that only partially represents the contents of each database. We call this partial representation of a full-text database its *language model.*

Language models can be arbitrarily complex and detailed, but they have so far tended to be simple, for example, consisting of a list of the terms that occur in the database, and their frequencies of occurrence [7, 1, 14, 6, 4, 15]. More complex language models might include information about phrases or other term co-occurrence information, but this type of information is less common, and its value unclear [15].

2.2 Acquiring Language Models Cooperatively: The STARTS Protocol

One technique for acquiring language models is the STARTS Protocol proposed by Gravano, et al. [5], and considered for inclusion in the ISO's Z39.50 standard interface for Information Retrieval systems [13]. STARTS is a protocol that allows the exchange of language models for full-text databases.

Although the protocol is detailed, the basic idea is simple. Each database provides a list of its index terms, along with information about their frequencies of occurrence. A small amount of corpus information is also included, for example, the number of documents contained, whether suffixes have been stripped from words ("stemming"), and whether extremely frequent words have been removed ("stopword removal").

The STARTS Protocol is an excellent solution when a single party controls all of the databases. However, it relies upon cooperation, which makes it a poor solution when databases are controlled by many parties. It can fail when database providers *can't* cooperate, *won't* cooperate, or *lie* (misrepresent their contents).

Databases that *can't* cooperate are often older, or "legacy" systems that have not been upgraded. These are inaccessible to systems that rely on cooperative solutions such as STARTS.

Cooperative protocols enable a database provider to choose *not to cooperate* with some database selection services. A database provider might be hostile to some selection services, for example based on corporate alliances, or it might simply have no incentive to cooperate with every selection service.

It is not uncommon for information providers on the Internet to *misrepresent* their services, in order to increase the number of people visiting their sites. The STARTS Protocol offers no protection against misrepresentation.

Although these problems are serious, perhaps the most serious problem is the assumption that vocabulary and frequency information provided by different databases is in some way comparable. In practice, it is difficult to know how a database containing 1,000 occurrences of *'apple'* compares to databases containing 2,000 occurrences of *'apple'* or no occurrences of *'apple'*. Full-text IR systems use a variety of word stemming algorithms, stopword lists, and case-conversion techniques, as well as specialized indexing for common phrases, names, locations, and dates. A database selection algorithm would find it nearly impossible to compensate for these differences among systems.

The STARTS Protocol is ideally suited to environments controlled by a single party, because cooperation can be enforced, deliberate misrepresentation is not a problem, and databases can be indexed uniformly. These characteristics are common, for example, within small and medium-sized organizations. However, within large organizations, and on the Internet, another approach is required.

3 Acquiring Language Models By Sampling

A database selection service recommends which databases should be searched to find documents that satisfy an information need. This observation implies that each database is capable of running queries and returning documents that match the queries. These are minimal criterion that we assume any database can satisfy.[1]

If queries can be run and documents retrieved, then it is possible to sample the contents of each database. Documents returned in response to a query necessarily constitute a biased sample, so we call this technique *query-based sampling*. It is well-known that the characteristics of a large population can be estimated from a relatively small random sample of the population. Our *hypothesis* is that query-based sampling can provide a sample of documents sufficiently random for learning an accurate language model of the entire database.

It is an open question how large a sample is required to construct language models of a specified accuracy. Word occurrences follow a highly skewed distribution, with a few words occurring very often, and most words occurring rarely [16]. Words in the middle of the frequency range are thought to be the most useful for distinguishing among documents within a single database [10]. There is also evidence that highly frequent words may be useful for distinguishing among databases [3]. These bits of evidence suggest that the important vocabulary occurs frequently in a database, and might therefore be acquired by sampling. The resource requirements, measured in queries run and documents examined, are likely to be reasonable.

The algorithm for query-based sampling is simple.

1. Select an initial query term.
2. Run a one-term query on the database.
3. Retrieve the top N documents returned by the database.
4. Update the language model based on the characteristics of the retrieved documents.
5. If a stopping criterion has not been reached yet,
   (a) Select a new query term; and
   (b) Go to Step 2.

The algorithm involves a number of specific choices, for example how query terms are selected, how many documents to examine per query, and when to stop sampling the database. We defer discussion of these choices to later sections of the paper.

Query-based sampling has a number of advantages as a technique for learning language models. It requires no special cooperation from a database beyond running queries and retrieving documents, which we assume is a minimum level of service required for a database to be useful. The absence of special cooperation enables query-based sampling to be used for learning language models from databases that *can't* or *won't* cooperate. It also makes misrepresenting database contents more difficult, because language models are learned as a consequence of normal database behavior.

Building language models directly from sampled documents also provides the database selection service with control over the content of the language model. It frees the selection service from the nearly impossible task of reconciling the different approaches to text indexing (word stemming, stopword removal, case conversion, name recognition, etc) taken in each database. The selection service can determine the degree of sophistication and detail applied in creating language models, is able to enforce consistency among language models, and is able to match the details of the language models to the characteristics of the selection algorithm.

Document samples s_1, s_2, ..., s_n taken from databases d_1, d_2, ..., d_n are also a rich resource that enables the selection service to provide additional capabilities beyond database selection. It enables more complex analysis of the contents of each database, for example to drive browsing or visualization aids (Section 7). The union of samples s_1, s_2, ..., s_n is a sample of the universe of databases served by the selection algorithm, which enables a different set of capabilities, such as co-occurrence-based query expansion (Section 8).

One important piece of information that appears difficult to acquire by sampling is the size of the database. Zipf's law and empirical evidence show that vocabulary growth slows, but does not stop, as additional documents are seen [16, 9], and that this rate is independent of database size. Hence it is unclear how to estimate database size by sampling.

Database size is primarily used by selection algorithms to scale the word frequencies in language models provided for databases of varying sizes. It is likely that a similar effect can be obtained by scaling the frequencies in learned language models by the sizes of the samples they are based upon. This would make database size

[1] We do not assume that every database will necessarily run queries and return documents *for free*. There could be a financial cost to acquiring language models for some databases, but that issue lies outside the scope of this paper.

Name	Size, in bytes	Size, in documents	Size, in unique terms	Size, in total terms	Variety
CACM	2MB	3,204	6,468	117,473	homogeneous
WSJ88	104MB	39,904	122,807	9,723,528	heterogeneous
TREC-123	3.2GB	1,078,166	1,134,099	274,198,901	very heterogenenous

Table 1: Test corpora.

a less necessary piece of information, although it would still be desirable.

4 Experimental Methodology

The hypothesis motivating our work was that accurate language models can be learned by sampling a text database with simple 'free-text' queries. This hypothesis was tested by comparing language models learned by sampling known databases (*'learned language models'*) with the *actual language models* for those databases. We also recorded the number of queries and documents required for the learned language models to achieve a given level of accuracy.

4.1 Language Models

Experiments were conducted on language models consisting of index terms (usually words) and their frequencies. Frequency was measured as the number of documents containing a term (*'document frequency'* or *df*).

Stopwords were not discarded when language models were constructed. During controlled testing, learned and actual language models were compared only on words that appeared in both language models, which effectively discarded from the learned language model any word that was considered a stopword by the database. The databases each used the default stopword list of the Inquery IR system, which contained 418 very frequent and/or closed-class words.

Suffixes were not removed from words ('stemming') when language models were constructed. During controlled testing, suffixes were removed prior to comparison to the actual language model, because the actual language models (the database indexes) were stemmed.

These choices are consistent with the language models explored most often in the research literature on database selection [7, 1, 6, 5, 4, 15].

4.2 Databases

Three full-text databases were used to test the effects of corpus characteristics and size on how quickly and accurately language models are learned. They were:

CACM: a small, homogeneous set of titles and abstracts of scientific articles from the *Communications of the ACM*;

WSJ88: the 1988 *Wall Street Journal*, a medium-sized corpus of American newspaper articles; and

TREC-123: a large, heterogeneous database consisting of TREC CDs 1, 2, and 3, which contains newspaper articles, magazine articles, scientific abstracts, and government documents [8].

These are standard test corpora used by many researchers. Table 1 summarizes their characteristics.

4.3 Metrics

The language models consisted of two types of information: a *vocabulary*, and *frequency* information for each term in the vocabulary. The correspondence between the learned vocabulary and the actual vocabulary was measured with two metrics, *percentage learned* and *ctf ratio*. The correspondence between the learned frequency information and the actual frequency information was measured with the *Spearman Rank Correlation Coefficient*. Each metric is described below.

4.3.1 Percentage Learned

The learned vocabulary is necessarily a subset of the actual vocabulary, because it is learned by examining a subset of documents in the database. It is natural to ask what proportion of the set is covered by the subset, or in this case, what proportion of the terms in the actual vocabulary are found in the learned vocabulary. We call this metric *percentage learned*.

The *percentage learned* metric is actually a poor match for text data, because of the skewed distribution of terms in a text database. In general, about 50% of the unique terms in a text database occur just once in the database; another 17% occur twice, and 8% occur three times [16, 9]. About 75% of the vocabulary of a text database is words that occur three times or less.

The *percentage learned* metric treats the terms in a language model as if they were all equally important. In reality most of them convey very little information about the contents of the database.

4.3.2 *Ctf* Ratio

The skewed distribution of terms in a text database is caused by a large number of essentially irrelevant terms, and a small number of frequent terms. The frequent and

moderately-frequent terms convey the most information about the contents of a database. A more appropriate metric for measuring the quality of a learned vocabulary is one that is weighted by the importance of each term.

Ctf ratio is such a metric. It measures the proportion of database term occurrences that are covered by terms in the learned language model. For a learned vocabulary V' and an actual vocabulary V, *ctf* ratio is:

$$\frac{\sum_{i \in V'} ctf_i}{\sum_{i \in V} ctf_i}$$

where ctf_i is the number of times term i occurs in the database (collection term frequency, or *ctf*).

A *ctf* ratio of 80% means that the learned language model contains the words that account for 80% of the word occurrences in the database. For example, if the database consists of 99 occurrences of *'apple'* and 1 occurrence of *'bear'*, and if the learned language model contains just *'apple'*, its *ctf* ratio is 99 / (99 + 1) = 0.99, or 99%.

4.3.3 Spearman Rank Correlation

The second component of a language model is term frequency information, which indicates the relative importance or descriptive power of each term in the database. Information Retrieval (IR) algorithms typically use frequency information as a component of a statistical ranking procedure [7, 6, 1]. For example, a term that occurs often in a particular database might be considered representative of its contents.

A sampling algorithm can estimate the *proportion* of documents in a database that contain the term. However, it is not known yet how to estimate the *size* of a database by sampling, so it is impossible to estimate the actual number of documents containing a term (*df*).

The estimated proportion of documents containing a term could be compared with the actual proportion of documents containing the term, but such a comparison is biased by the number of documents examined. For example, if the true proportion is 86%, the most accurate estimate possible after seeing 10 documents is 90%, hence a certain amount of error is built into this type of comparison.

It is more accurate to rank terms by their frequency of occurrence and then compare the rankings of terms that occur in both the database and the learned language model. Zipf's Law indicates that there is a predictable relationship between a term's rank and its frequency in the database [16]. Given a term's rank, its frequency can be estimated relatively accurately, and vice versa.

The Spearman Rank Correlation Coefficient is an accepted metric for comparing two rankings [12]. The Spearman Rank Correlation Coefficient is defined as:

$$R = 1 - \frac{6}{n^3 - n} \Sigma d_i^2$$

where d_i is the rank difference of common term i, and and n is the number of terms. Two rankings are identical when the correlation coefficient is 1. They are uncorrelated when the coefficient is 0, and in reverse order when the coefficient is -1.

Database selection does not require a rank correlation coefficient of 1.0. It is sufficient for the learned language model to represent the relative importance of index terms in each database to some degree of accuracy; for example, it might be sufficient to know the ranking of a term ±5%. Although most database selection algorithms are likely to be relatively insensitive to small ranking errors, it is an open question how much error a given algorithm can tolerate before selection accuracy deteriorates. That question, although clearly important, lies outside the scope of this paper. In this paper we simply study the degree of correlation between the learned and actual rankings under a variety of experimental conditions.

4.4 Setting Parameters

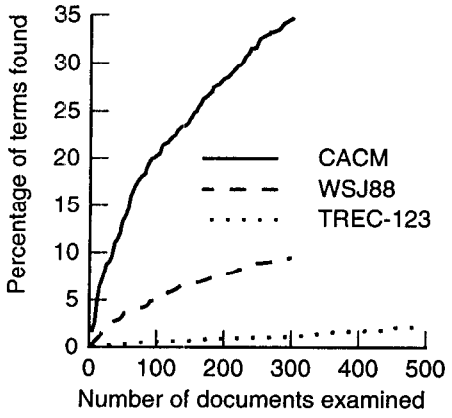
Experiments with query-based sampling require making choices about how query terms are selected and how many documents are examined per query.

In our experiments, the first query run on a database was always determined by selecting a word randomly from the actual TREC-123 language model. The initial query could be selected using other criteria, for example selecting a frequent term, or selecting a term from another language model. Several informal experiments found that the choice of the initial query term had little effect on the quality of the language model learned.

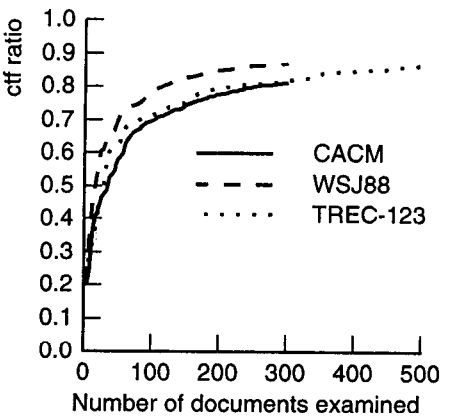
Subsequent query terms were chosen by a variety of methods, which are described in the following sections. However, in all cases the terms chosen were subject to certain requirements, in order to avoid selecting terms that would be likely to retrieve few or no documents. A term selected as a query term could not be a number and was required to be 3 or more characters long. These requirements are similar to the requirements often placed on index terms in text retrieval systems.

We had no hypotheses to guide the decision about how many documents to sample per database query. Instead, experiments were conducted to determine the effect of varying this parameter.

The CACM and WSJ88 experiments presented in this paper were ended after examining 300 documents. The TREC-123 experiments presented in this paper were ended at 500 documents. These stopping criteria were chosen empirically after running several initial experiments, and were biased by our interest in learning language models from small (ideally, constant) sized samples. Several experiments with each database were continued until several thousand documents were sampled, to ensure that nothing unusual happened.



(a)



(b)

Figure 1: Measures of how well a learned language model covers the vocabulary of a full-text database. **(a)** Percentage of database terms covered by the learned language model. **(b)** Percentage of database word occurrences covered by terms in the learned language model. (Four documents examined per query.)

5 Experimental Results

A series of experiments was conducted in which the method of generating queries and the number of documents examined per query were varied. The goals of these experiments were to determine whether effective language models were learned at all, and if so, what combination of parameters produced either the fastest learning or the most accurate language models.

In all experiments, the first query run on a database was determined by selecting a word randomly from the TREC-123 database, as described above.

The baseline experiment consisted of choosing the second and subsequent query terms randomly from the language model being learned, and examining four documents per query. The number four was determined empirically; later in this paper we present results for other numbers of documents. Results for the baseline experiment are shown in Figures 1 and 2.

The graphs in Figures 1a and 1b demonstrate why the percentage of terms learned is a poor metric for judging the quality of a language model. About 250 documents had to be seen in order to discover about a third of the CACM vocabulary, but that vocabulary represented about 80% of the word occurrences in the CACM database. The contrast was more dramatic on the TREC-123 database; in that test, only about 1% of the vocabulary was discovered after 250 documents, but it represented about 81% of the term occurrences.

Recall that 418 *stop words*, including such frequent terms as "the", "and", and "a", were discarded from the language models before these comparisons were done. Stop words are usually words that are necessary to the language syntax but that convey little information. If stop words were not discarded from the vocabulary, the *ctf* ratio would grow and converge even more rapidly.

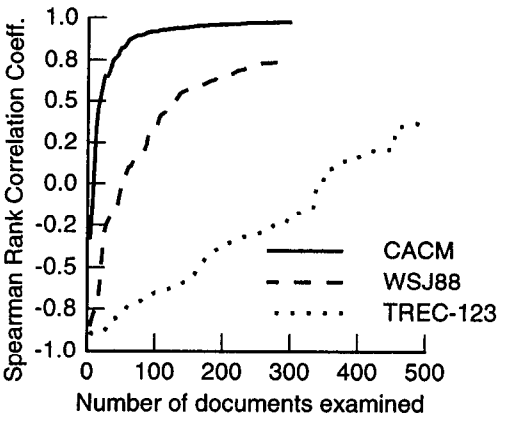


Figure 2: A measure of how well a term ranking by document frequency (*df*) in the learned language model agrees with a term ranking by document frequency in the full-text database. (Four documents examined per query.)

The percentage of terms found after examining a given number of documents is a function of database size, hence it varies widely among the three databases examined. However, the *ctf* ratios in these experiments indicate that most of the frequent terms are found after examining a fixed number of documents, no matter how big the database is. At 250 documents, *ctf* ratio is greater than 80% for all three databases, and the curves are leveling, implying that after a certain point most of the new terms occur rarely. This is consistent with Zipf's law [16].

Figure 2 shows the rate at which the ranking of terms in the learned language model begins to match the ranking of terms in the actual language model. The model for the small homogeneous CACM database converges

rapidly; at 82 documents, the correlation coefficient exceeds 0.9. The model for the larger, more heterogeneous WSJ88 database converges more slowly, reaching a correlation coefficient of 0.76 at 300 documents. The even larger, more heterogeneous TREC-123 database converges the most slowly, reaching a correlation coefficient of 0.4 at 500 documents.

Unlike the *ctf* ratio, which appeared to converge at a constant rate despite database size, the correlation of term rankings appears to be influenced by database size. The model for the CACM database reached a high degree of correlation, but had sampled 2.6% of the database (82 documents). The model for WSJ88 reached a lesser degree of correlation, but had sampled just 0.8% of the database (300 documents). The model for TREC-123 sampled only 0.04% of the database (500 documents).

These experiments demonstrate that representative term frequencies are learned by sampling only a small fraction of the contents of a database. However, they also raise questions that we cannot yet answer.

It is likely that the ranking errors are not distributed evenly throughout the rankings, because there is far more opportunity to gather information about the proper ranking of frequent terms than rare terms. How best to measure the distribution of error is the subject of current research.

It is also an open problem how correlated the rankings need to be for accurate database selection. A correlation coefficient of 0.9 (CACM) is almost certainly not required. Whether a correlation coefficient of 0.4 (TREC-123) is sufficient is not known. If it is not, sampling can be continued to reach whatever level of correlation is required. For example, sampling for another 100 queries (400 documents) would greatly improve the degree of correlation, while raising the size of the sample from 0.04% of the database to 0.08%.

5.1 Number of Documents Examined Per Query

The baseline experiments retrieved from the database, or sampled, the four most highly ranked documents for each query. The number four is used here as a baseline because it produced good results empirically. However, the number four is a parameter that can be varied, and it is reasonable to investigate the effects of varying it.

There is a cost to running queries; the database provider must do computation, and the process building the language model must wait. Costs might be reduced by examining more than four documents per query. For example, searchable databases on the Web often return 10 document titles in response to a full-text query, with additional documents available upon request. This characteristic might suggest examining as many as 10 documents per query.

Docs/ Qry	CACM		WSJ88		TREC-123	
	Docs	SRCC	Docs	SRCC	Docs	SRCC
1	267	0.97	123	0.40	193	-0.27
2	251	0.97	123	0.43	185	-0.23
4	248	0.97	114	0.43	211	-0.35
6	231	0.97	135	0.47	288	-0.43
8	194	0.97	141	0.44	216	-0.39
10	229	0.98	107	0.48	248	-0.40

Table 2: Effect of varying the number of documents examined per query on how long it takes a sampling method to reach a *ctf* ratio of 80%. *Docs* is the number of documents that had to be examined. *SRCC* is the Spearman Rank Correlation Coefficient.

Conversely, the process of running queries and retrieving documents is intended to approximate a random sampling of the database. However, a query necessarily returns a biased sample of documents. If the documents returned by a query have similar vocabularies and term frequency patterns, then little is gained by examining many of them. This consideration might suggest examining a small number of documents per query, perhaps as few as one per query.

A series of experiments was performed to determine the effect of varying N, the number of documents examined per query. Values of $N = 1, 2, 4, 6, 8$, and 10 were tested.

It is difficult to see from graphs that there is any difference in the accuracy of the language models learned by examining different numbers of documents per query, hence they are not included in this paper. The differences are small; query-based sampling produces stable results over a range of parameter settings.

For example, consider the point at which the *ctf* ratio reaches 80% for each database (Table 2). The language model for the small, homogeneous CACM database is learned most quickly when 8–10 documents are examined per query. Four documents per query is better for the larger and heterogeneous WSJ88 database, and 2 documents per query is best for the even larger and more heterogeneous TREC-123 database.

It appears to make little difference whether 1, 2, or 4 documents are examined per query. The differences are sufficiently small that the decision can be based on other criteria, for example, the relative costs of running queries and examining documents. (Hence our choice of 4 as a baseline, which requires fewer queries to reach a given number of documents.) However, the results with the TREC-123 database show that there can be a significant cost to examining too many documents per query, presumably because the samples are less diverse.

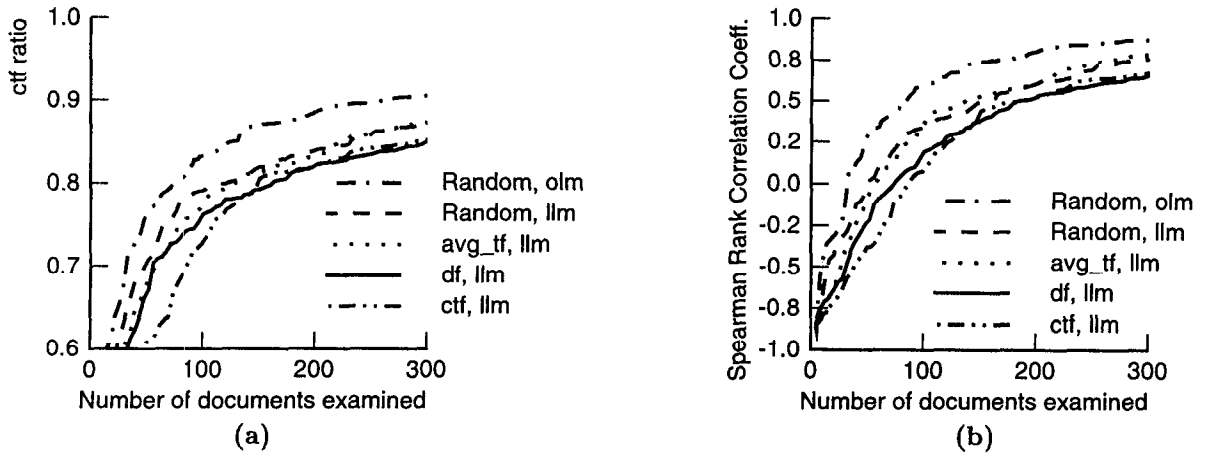


Figure 3: Measures of how different query selection strategies affect the accuracy of a learned language model. **(a)** Percentage of database word occurrences covered by terms in the learned language model. **(b)** Spearman rank correlation coefficient between the term rankings in the learned language model and the database. (1988 Wall Street Journal database. Four documents examined per query.)

	Random, olm	Random, llm	avg_tf, llm	df, llm	ctf, llm
Number of queries	167	78	107	155	153

Table 3: The number of queries required to retrieve 300 documents using different query selection criteria.

5.2 Query Selection Strategies

The baseline experiments select query terms randomly from the learned language model. Random selection from the learned language model is used as a baseline in this paper because it produced good results empirically. However, query terms could be selected from the learned language model using other criteria, or they could be selected from another language model. The query selection strategy is another parameter, and it is reasonable to investigate the effects of varying it.

An early hypothesis was that frequent terms in a database would produce a relatively random sample of documents, because they would be more likely to occur in a variety of contexts. There are many metrics for measuring term frequency in a database, but the three most common in Information Retrieval are document frequency (*df*), collection term frequency (*ctf*), and average term frequency ($avg\_tf = ctf \,/\, df$). Although these metrics are related, they have differing characteristics, and tend to be useful for different purposes. Each was tested as a method of selecting query terms from the learned language model.

An early concern was that the learned language model would initially be biased strongly towards the documents that just happened to be sampled first, and that that bias would be reinforced by continuing to select query terms from the learned language model. A solution would be to select terms from another, more complete language model, in the hopes of getting a more random set of query terms. This hypothesis was designated the 'other language model', or *olm*, hypothesis. It, too, was tested, and compared with the 'learned language model', or *llm*, technique used throughout the previous sets of experiments.

The 'other' language model used in these experiments was the full TREC-123 language model. This choice clearly creates an unfair bias in favor of the TREC-123 database. However, we were interested in seeing whether an 'other' language model that was well-matched with the actual language model would provide any advantage; if it appeared promising, it could be investigated more carefully.

A series of experiments was conducted, using the same methodology used in the previous sets of experiments. The number of documents examined per query was four. Query terms were selected either from the learned language model using one of the term frequency metrics described above, or randomly from the 'other' language model. All three databases (CACM, WSJ88, and TREC-123) were tested. The results were similar for each database, so only results for WSJ88 are presented here (Figure 3 and Table 3).

The Random olm (other language model) experiments learned the important vocabulary and term fre-

quency information more quickly than the Random llm (learned language model) (Figure 3). However, they also required twice as many queries to create a sample of 300 documents (Table 3). This difference is caused by selecting query terms that either don't occur in the sample database, or that occur in fewer than $N = 4$ documents.

Although the results using the learned and other language models are relatively similar in this experiment, the 'other' language model was an exact match to one sampled database (TREC-123) and a superset of another (WSJ88). We view the Random, olm results with caution, because the number of failed queries might have been higher if we had selected query terms from the language model of a less similar database.

The experiments demonstrate that selecting query terms from the learned language model, as opposed to a more complete 'other' language model, does *not* produce a markedly skewed sample of documents. The rate of learning is faster if measured by the number of queries run, and slower if measured by the number of documents examined. Whichever metric is used, a relatively unbiased language model is learned with moderate cost.

The experiments also demonstrate that selecting query terms randomly from the learned language model is more effective than selecting them based on high frequency. This result was a surprise, because our hypothesis was that high frequency terms would either occur in many contexts, or would have relatively weak contexts, producing a more random sample. This hypothesis was not validated by the experiments.

It may be that some high frequency terms tend to co-occur frequently in similar contexts, for example 'stocks' and 'bonds' in the Wall Street Journal. A more careful approach to selecting high frequency terms, for example, based on part of speech (verbs, adjectives), or paying more attention to co-occurrence relationships in the sampled documents, might produce better results. Or, it may simply be that random selection of query terms produces a more random sample of documents than frequency-based selection.

6 Stopping Criteria

The term rankings in a learned language model converge to the term rankings in the actual language model as the number of documents examined increases. However, the differences between the learned and actual rankings are inconsequential long before they disappear completely. A *stopping criterion* is needed to let the sampling system decide when the learned language model is sufficiently accurate.

Our hypothesis was that the learned language model did not converge to the actual language model at an even rate. This hypothesis is supported to some extent by the experiments described above. The Spearman Rank Correlation Coefficient initially rises rapidly, but then levels off (Figures 2 and 3). If the rate of convergence slows at a predictable rate, it might be possible to use this information in a stopping criterion.

One simple technique is to compare the learned language model at time t with the learned language model at time $t+\delta$. If the two learned language models are sufficiently similar, one might conclude that the learned language model is indeed converging.

Similarity among the two language models could be determined with the Spearman Rank Correlation Coefficient, but its semantics are not obvious or intuitive. Experiments also showed that it is not well-suited for identifying small improvements in correlation. For example, language models learned at 50 document intervals tended to be highly correlated with each other (correlation coefficient 0.9997), even when there were noticeable differences in how well they correlated with the actual language model.

Instead, we defined a new metric, *rdiff*, which measures the average rank difference of each term t_i in two rankings R_1 and R_2. *rdiff* can be viewed as the average distance, measured as a percentage of the number of ranks, that each term must move to convert one ranking into the other. *rdiff* is defined as:

$$\frac{1}{n^2} \cdot \Sigma \text{abs}(d_i)$$

where d_i is the rank difference of common term i, and and n is the number of terms.

For example, given two rankings of 100 terms that are identical *except* that term t_1 is ranked 4th in one ranking and 5th in the other, while term t_2 is ranked 5th in one ranking and 4th in the other (i.e., the two terms swap rankings), $rdiff = (1/(100 * 100)) * (2) = 0.0002$.

If only one term could occupy each rank, *rdiff* would vary between 0.0 and 0.5. When multiple terms can occupy each rank, as is usually the case in language models, *rdiff* varies between 0.0 and 1.0.

Figure 4 shows the *rdiff* between the language models created at 50-document increments for three databases. The *rdiff* between the language model created from 50 CACM documents and the language model created from 100 CACM documents is 0.01224. This *rdiff* means that the average term in one ranking must move a distance of $1.2\% * n$ to reach its place in the other ranking, where n is the number of terms the two rankings have in common.

The *rdiff* values observed were generally small, indicating that the language models did not change dramatically over 50 document increments.

However, the more interesting results are that *rdiff* values between different snapshots of the language model fell as more documents were examined, and that they appeared to do so independently of database size.

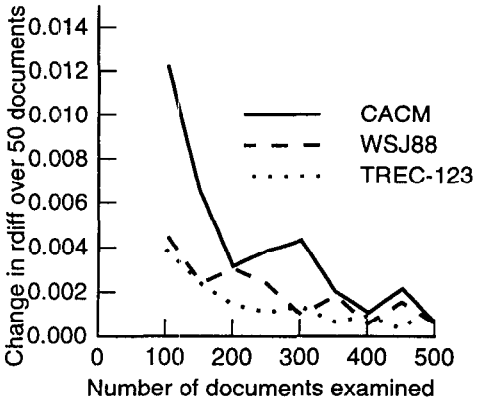


Figure 4: Average distance a term must move to convert the ranking after $D-50$ documents to the ranking after D documents. (Four documents examined per query.)

The former result suggests that it is possible to establish a stopping criterion based only on information that can be observed as language models are being learned. For example, a language model might be accurate enough when $rdiff \leq 0.004$ over 2 consecutive 50 document spans.

The latter result, that *rdiff* falls independently of database size, relates to a discrepancy between what the *ctf* ratio and Spearman Rank Correlation Coefficient indicate about how language models converge. *ctf* ratio suggests that they converge in a nearly constant number of documents (Figures 1 and 3a). The Spearman coefficient suggests that the rate of convergence is influenced by database size (Figures 2 and 3b). *rdiff* appears to agree with *ctf* ratio that a constant number of documents is sufficient.

The experiment with *rdiff* is significant because it suggests that a stopping condition can be created that depends only upon information that is observable while learning a language model. It also raises the question of whether such a stopping condition is even required, or whether it is sufficient to sample a constant number of documents, irrespective of database size and characteristics.

7 A Peak Inside: Summarizing Database Contents

Our interest is primarily in an automatic method of learning language models that are sufficiently accurate and detailed for use by automatic database selection algorithms. However, a language model can also be used to indicate to a person the general nature of a given text database.

A simple and well-known method of summarizing database contents is to display the terms that occur frequently and are not stopwords. This method is effective because databases are in some sense guaranteed to be about the words that occur most often.

We illustrate this point with a 'real world' example based on the Microsoft Customer Support database, available on the Web. We chose this database because its contents are well-known. Although this example is presented late in this paper, it is based on our earliest sampling research, when it was unclear what were the best parameter settings. The database was sampled as in other experiments described above, except that 25 documents were examined per query. Subsequent research indicated that fewer documents per query is more efficient (Section 5.1).

Terms in the learned language model were ranked for browsing by *df*, *ctf*, and *avg_tf*. The rankings produced by all three metrics allow one to see easily that the database contains documents about Microsoft software. *avg_tf* produced the most informative ranking, because words such as "excel", "foxpro", "microsoft", "nt", "access", and "windows" are ranked highly, and because the list contains more content words (Table 4). However, one cannot draw strong conclusions about how to summarize database contents from this one test.

Although simple word lists are effective for summarizing database contents, they are not necessarily the most effective techniques. Frequent phrases, and common relationships among words or concepts, are known to be better.

Indeed, one consequence of creating language models from sampled documents is that it makes more powerful summarizations possible. The sampling process is not restricted just to words lists and frequency tables, nor is it restricted to just the information the database chooses to provide.

Instead, it has a set of several hundred documents from which to mine frequent phrases, names, dates, relationships, and other interesting information. This information enables construction of more powerful and more informative summaries than is possible with the simple language models used by cooperative methods.

8 Query Expansion

Query expansion is a process in which terms are added to a query during document retrieval, to make it longer and more representative of a person's information need. Query expansion can occur automatically, without user assistance or knowledge, or it can be done interactively.

The state-of-the-art in document retrieval is query expansion based on co-occurrence. Expansion terms are words and phrases that tend to occur in the database often with query terms, but that are not necessarily synonyms. For example, the phrase *'illegal alien'* might be added to the query *'immigration'* if they tend to co-occur in documents in the database.

term	*avg_tf*	term	*avg_tf*	term	*avg_tf*	term	*avg_tf*	term	*avg_tf*
project	10.924	microsoft	5.736	access	4.554	set	3.948	command	3.504
excel	8.750	object	5.637	print	4.550	application	3.919	following	3.387
office	8.565	user	5.297	data	4.322	product	3.890	windows	3.369
works	7.389	visual	5.273	internet	4.268	menu	3.840	new	3.369
server	7.271	beta	4.986	error	4.217	text	3.717	settings	3.317
word	7.221	service	4.983	box	4.213	software	3.621	example	3.152
table	6.639	basic	4.903	articles	4.121	code	3.617	version	3.147
printer	6.507	file	4.867	setup	4.094	name	3.611	message	3.119
foxpro	6.486	nt	4.845	mail	4.067	system	3.544	information	3.076
database	6.117	field	4.729	users	4.042	dialog	3.515	select	3.072

Table 4: The top 50 words found by sampling the Microsoft Customer Support Database (ranked by *avg_tf*).

A recent paper showed that co-occurrence query expansion can also significantly improve automatic database selection [15]. Indeed, it is arguably more important in database selection, because language models contain no information about which documents in a database contain which terms. A database may contain many occurrences of *'white'* and *'house'*, but they may not occur in the same documents. If it also contains many occurrences of *'president'*, *'clinton'*, *'oval'* and *'office'*, it is far more likely to contain documents about U.S. politics.

Co-occurrence query expansion algorithms require a large representative database of documents in which to analyze co-occurrence patterns. In ordinary document retrieval, the database being searched is also the database that provides the co-occurrence information.

However, when the task is database selection, it is not clear what database can be used to expand the query for *selecting databases*. Query expansion from any specific database introduces a bias towards selecting that database. It has been an open problem which query expansion database to use for general database selection tasks.

The sampling method of building language models solves that problem. In the course of building language models, it acquires database samples s_1, s_2, ..., s_n from databases d_1, d_2, ..., d_n. The union of these samples s_1, s_2, ..., s_n contains vocabulary, frequency of occurrence, and frequency of co-occurrence patterns that occur in the *set* of databases served by the selection algorithm. The union of samples favors no specific database, but reflects patterns that are common to them all. *It* is the appropriate database to use for query expansion during database selection.

We view this as an advance in making query expansion a common part of automatic database selection.

9 Conclusions

Few people used Information Retrieval systems a decade ago. Now IR systems are used by millions of people every day, in the form of Internet Web-search systems. However, much of the information available on computer networks is not stored in commercial Web-search databases, and never will be. It is scattered across thousands of other searchable text databases, managed by a variety of large and small content providers. A challenge for the research community is to make information in these many databases as accessible as is information within a single database today.

Automatic database selection is a solution with many advantages, but it has been an incomplete solution for large-scale, multi-party environments such as the Internet. The cooperative nature of language model dissemination protocols such as STARTS do not address a variety of problems encountered in 'real world' environments.

The *query-based sampling* approach to language model acquisition presented in this paper avoids the problems of cooperative protocols. It can be applied to older ('legacy') systems, it applies to systems that have no specific incentive to cooperate, and it is not as easily defeated by intentional misrepresentation. It also avoids the problem of reconciling the different word stemming, stopword, and customized indexing representations used within each text database. The representation problems are arguably the most serious problems with the cooperative approach, because they apply even when database providers *intend* to cooperate.

This paper shows that query-based sampling can produce reasonably accurate language models for text databases of varying size and contents from just a few hundred documents. The documents can be acquired by running about one hundred single-term queries. The resource requirements, measured in queries, amount of computation, or amount of network traffic, is low.

An additional benefit of sampling database documents directly is that the selection service can construct for each database a set of language models of varying complexity. A relatively simple word and frequency language model might be used for database selection.

A more detailed language model identifying people, places, and relationships might be used for browsing.

The selection service can also use the documents acquired by sampling to build a database for co-occurrence-based query expansion. It is known that this type of query expansion significantly improves the accuracy of database selection, but it was an open problem how such a database could be acquired. This paper presents a solution.

The work reported in this paper is an important step in the direction of large-scale automatic database selection, but it is only a first step. Some database selection algorithms need to know the number of documents in a database, for scaling purposes, but it is an open problem whether database size can be estimated by sampling its contents. The criteria for recognizing when a language model is sufficiently accurate are more ad-hoc than we would prefer; a more principled criterion is the subject of current research. Finally, it would not be surprising if the query selection criteria could be improved, leading to more rapid learning of language models.

Acknowledgements

This paper is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement EEC-9209623, and by the U.S. Patent and Trademark Office and Defense Advanced Research Projects Agency/ITO under ARPA order D468, issued by ESC/AXS contract F19628-95-C-0235. Any opinions, findings, conclusions, or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsors.

References

[1] J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–28, Seattle, 1995. ACM.

[2] P. B. Danzig, J. Ahn, J. Noll, and K. Obraczka. Distributed indexing: A scalable mechanism for distributed information retrieval. In *Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 220–229, Chicago, IL, October 1991. ACM.

[3] S. T. Dumais. Latent semantic indexing (LSI) and TREC-2. In D. K. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, pages 105–115, Gaithersburg, MD, 1994. National Institute of Standards and Technology, Special Publication 500-215.

[4] J.C. French, J.C. Powell, C.L. Viles, T. Emmitt, and K.J. Prey. Evaluating database selection techniques: A testbed and experiment. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 1998.

[5] L. Gravano, K. Change, H. García-Molina, and A. Paepcke. STARTS Stanford proposal for Internet meta-searching. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, 1997.

[6] L. Gravano and H. García-Molina. Generalizing GLOSS to vector-space databases and broker hierarchies. In *Proceedings of the 21st International Conference on Very Large Databases (VLDB)*, pages 78–89, 1995.

[7] L. Gravano, H. García-Molina, and A. Tomasic. The effectiveness of GLOSS for the text database discovery problem. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, pages 126–137. ACM, 1994.

[8] D. Harman, editor. *Proceedings of the Third Text REtrieval Conference (TREC-3).* National Institute of Standards and Technology Special Publication 500-225, Gaithersburg, MD, 1995.

[9] H. S. Heaps. *Information Retrieval: Computational and Theoretical Aspects.* Academic Press, New York, 1978.

[10] H.P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research*, 2:159–165, 1958.

[11] R. S. Marcus. An experimental comparison of the effectiveness of computers and humans as search intermediaries. *Journal of the American Society for Information Science*, 34:381–404, 1983.

[12] M.J. Moroney, editor. *Facts from figures.* Penguin, Baltimore, 1951.

[13] National Information Standards Organization. *Information Retrieval (Z39.50): Application Services Definition and Protocol Specification (ANSI/NISO Z39.50-1995).* NISO Press, Bethesda, MD, 1995.

[14] E.M. Voorhees, N.K. Gupta, and B. Johnson-Laird. Learning collection fusion strategies. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 172–179, Seattle, 1995. ACM.

[15] J. Xu and J. Callan. Effective retrieval of distributed collections. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 112–120, Melbourne, 1998. ACM.

[16] G. K. Zipf. *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology.* Addison-Wesley, Reading, MA, 1949.