Mining the Web to Create Minority Language Corpora

Rayid Ghani
Carnegie Mellon Univ., and
Accenture Technology Labs
Rayid.Ghani@cs.cmu.edu

Rosie Jones
Carnegie Mellon University,
Pittsburgh, PA, USA
Rosie.Jones@cs.cmu.edu

Dunja Mladenić
J. Stefan Inst., Slovenia and
Carnegie Mellon Univ.
Dunja.Mladenic@ijs.si

ABSTRACT

The Web is a valuable source of language specific resources but the process of collecting, organizing and utilizing these resources is difficult. We describe CorpusBuilder, an approach for automatically generating Web-search queries for collecting documents in a minority language. It differs from pseudo-relevance feedback in that retrieved documents are labeled by an automatic language classifier as relevant or irrelevant, and this feedback is used to generate new queries. We experiment with various query-generation methods and query-lengths to find inclusion/exclusion terms that are helpful for retrieving documents in the target language and find that using *odds-ratio* scores calculated over the documents acquired so far was one of the most consistently accurate query-generation methods. We also describe experiments using a handful of words elicited from a user instead of initial documents and show that the methods perform similarly. Experiments applying the same approach to multiple languages are also presented showing that our approach generalizes to a variety of languages.

1. INTRODUCTION

Electronic text corpora are used for modeling language in many language technology applications, including speech recognition [12], optical character recognition, handwriting recognition, machine translation [2], and spelling correction [10]. Language corpora are also useful for linguistic and sociolinguistic studies, as they are readily searchable and statistics can easily be computed.

The Linguistic Data Consortium (LDC) has corpora for twenty languages [13] while Web search engines currently perform language identification on about a dozen of the languages they index, allowing language-specific searches in those languages. Documents in many other languages are also indexed, though no explicit labeling of the language they are written in is available.

The WWW has been gaining popularity as a resource for multilingual content. Resnik [16] use the Web to automatically constructed a parallel corpus of English and French. We describe techniques for automatically collecting language specific resources from the Web and present a system which automatically generates Web search queries to construct corpora for minority languages. Here we refer to languages which are in the minority on the web, but which have speakers reading and creating web pages.

While search-engines are an invaluable means of accessing the Web for users, automated systems for learning from the Web have primarily been installed in crawlers, or spiders. A new generation of algorithms is seeking to augment the set of search capabilities by combining other kinds of topic or target-directed searches. Glover and colleagues [9] use machine learning to automatically augment user queries for specific documents with terms designed to find document genres, such as home-pages and calls for papers. Rennie and McCallum [15] use reinforcement learning to help a crawler discover the right kinds of hyperlinks to follow to find postscript research papers. Diligenti et al. [6] make use of hyperlink structure to learn naive Bayes models of documents a few back-links away from target documents to give their crawler more information about the types of hyperlinks to follow. Chen et al.'s WebSail [5] uses reinforcement learning based on feedback from the user about the relevance of a query match. Boley et al. [1] propose using the most-frequent words for query generation for their WebACE system, generating these from clusters, seeking to maximize term-frequency and document frequency of the terms selected. They use stemmed version of words as query terms, and show by example that automatically generated queries can be used to find more related documents. They use queries that used a combination of conjunctive and disjunctive terms. However, they do not evaluate a system employing automatic query-generation.

Ghani and Jones [7] describe an algorithm for building a language-specific corpus from the Web using single-word queries. However, their experiments are limited to a small closed corpus of less than 20,000 documents, vastly limiting the generalization power of their results to the Web. The approach of using single words from a language as queries to a Web search engine results in very low precision when finding documents in that language. On the Web, a single word from one language can also appear in another as an acronym. In addition, a single word from one language may be present as a vocabulary item in another language. Romance languages, for example, share many high-frequency word-strings, including "de", "la" "si" and "se". Slavic languages also share many word-strings, for example "in", "na", "pa", "ali". This means that we cannot count on simple word-inclusion of small number of terms to find us documents in a particular language. In general, to quickly find documents in a specific language, we need to be able to construct queries that both find a wide range of documents in the target language, and that filter out a large proportion of more and less closely related languages. Our hypothesis is that by selecting appropriate inclusion and exclusion terms from documents already collected, and using the results of classification by a high-precision language filter, we can construct very high-precision queries automatically. Our experiments show that this approach works well and also does not require specialized knowledge of which languages are related.

2. CORPUSBUILDER ARCHITECTURE

In this section we describe the CorpusBuilder architecture, query-generation methods and the language-filter used. Our approach differs from pseudo-relevance feedback [17], [18] in that retrieved documents are labeled by an automatic language classifier as relevant or irrelevant, and this feedback is used to generate new queries.

2.1 General Algorithm

CorpusBuilder iteratively creates new queries, in order to build a collection of documents in a single language. The target language is defined by one or more initial documents provided by the user, and the language filter.

At a high level, CorpusBuilder works by taking as initial input from the user two sets of documents, relevant and non-relevant. Given these documents, it uses a term selection method to select words from the relevant and non-relevant documents to be used as inclusion and exclusion terms for the query, respectively. This query is sent to the search engine and the highest ranking document is retrieved, passed through the language filter and added to the set of relevant or non-relevant documents according to the classification by the filter. The process is then iterated, updating the set of documents that the words are selected from at each step. Here we refer to a URL returned by the search engine as a *hit*. When querying the search engine with a new query, only the first hit is used but the remaining hits are stored to a file for efficiency, so that we avoid re-querying the search engine for the same query. If we have used the same query before, we take the next unseen hit from our cached results. If all hits have been seen, no hit is returned. The general algorithm is as follows:

1. Initialization
2. Generate query terms from relevant and non-relevant documents
3. Retrieve next most relevant document for the query
4. Language Filter, assign document to relevant or non-relevant set
5. Update frequencies and scores based on relevant and non-relevant documents
6. Return to step 2.

We retrieve a single document in step 3 to allow the algorithm maximum opportunities to improve performance using the language filter at every step. Interesting future work would involve investigating how the number of documents retrieved before updating models could be optimized, possibly by examining the number of positive documents returned so far by the current query.

2.2 Query Generation

Given a collection of documents classified into relevant and non-relevant, the task of a query generation can be described as follows: examine current relevant and non-relevant documents to generate a query which is likely to find documents that are *similar* to (but not the same as) the relevant documents (i.e. also relevant) and not similar to the non-relevant documents. This is analogous to sampling without replacement. The Web contains different documents in a variety of natural languages. We use query generation and a search engine to sample from this document collection. Ghani and Jones experimented with sampling with and without replacement and found that both strategies performed similarly for their dataset [7].

We generate queries using conjunction and negation of terms only. A query is defined to consist of a set of terms required to appear in the documents retrieved (inclusion terms), and a set of terms forbidden from appearing in the documents retrieved (exclusion). Consequently, each query can be described by four parameters: the number of inclusion terms, the number of exclusion terms, the inclusion term selection method, and the exclusion term selection method. This contrasts with full Boolean queries, which give greater expressive power by also employing disjunction, and negation with a greater variety of scope . We chose to use only conjunction to simplify the experiments, and for all experiments we set the number of inclusion terms equal to the number of exclusion terms, and used a fixed term selection method throughout the entire experiment. Using a less restrictive querying language may permit more precise queries, but we would then need to learn when to use disjunction, and over what scope in the terms. We have done related experiments using machine learning to choose the term selection method at each iteration [8]. The term selection method selects k inclusion and k exclusion terms using the words that occur in relevant and non-relevant documents, with different methods using different term weighting schemes to select terms. Query term selection methods were: *uniform*, *term-frequency*, *probabilistic term-frequency*, *rtfidf*, *odds-ratio*, *probabilistic odds-ratio*. Each is described below for inclusion terms. The operation for exclusion terms is analogous, swapping relevant and non-relevant documents where appropriate.

- *uniform* (UN) selects k terms from the relevant documents, with equal probability of each term being selected.

- *term-frequency* (TF) selects the k most frequent terms from the relevant documents.

- *probabilistic term-frequency* (PTF) selects k words from the relevant documents according to their unsmoothed maximum-likelihood probabilities, that is, with probability proportional to their frequency. More frequent words are more likely to be selected.

- *rtfidf* (RTFIDF) selects the the top k words ranked according to their rtfidf scores. The rtf score of a term is the total frequency of that term calculated over all relevant documents as classified by the language filter. The idf score of a term is calculated over the entire collection of documents retrieved, and is given by $log(\frac{total\ number\ of\ documents}{number\ of\ documents\ containing\ the\ term})$. rtfidf for a term is the product of rtf and idf.

- *odds-ratio* (OR) selects the k terms with highest odds-ratio scores. The odds-ratio score for a word w is defined as

$$log_2\left(\frac{P(w|relevant\ doc) * (1 - P(w|nonrelevant\ doc))}{P(w|nonrelevant\ doc) * (1 - P(w|relevant\ doc))}\right)$$

- *probabilistic odds-ratio* (PO) selects words with probability proportional to their odds-ratio scores.

The simplest measure used as a baseline is random selection of terms, i.e. a uniform probability distribution is imposed over all words in the vocabulary (UN). Scoring terms according to their frequency (TF) has been used as a simple measure known to give good results for feature scoring in document categorization [14], [20]. Using a multinomial distribution over frequency of terms (PTF - *probabilistic-term-frequency*) has been shown to perform better than simple frequency on a similar problem [7]. Haines and Croft [11] show that *rtfidf* is a good scoring mechanism for information retrieval. Mladenic and Grobelnik [14] show that scoring using *odds-ratio* (OR) achieves very good results on document categorization when dealing with a minority concept, which is similar to our problem scenario since Slovenian is a minority language on the Web. Motivated by the superior performance of *probabilistic term-frequency* over *term-frequency*, we derived a variant of *odds-ratio*, which is selecting terms according to a multinomial distribution over odds-ratio scores of terms (PO – *probabilistic-odds-ratio*).

The query generated at each step may be a novel query, or one we have already issued, either because the method is probabilistically selecting terms or because the addition of new documents did not change word distributions in a way which influences the term selection.

2.3 Recovery from Empty Query Results

In the case of a deterministic term-selection method, such as *term-frequency*, *rtfidf* and *odds-ratio*, query terms selected can change only when the underlying document statistics change through the addition of a new document. When a query adds no new documents, we need a method of altering the query to recover. We took the approach of successively incrementing a counter i, first through inclusion terms, then through the exclusion terms, taking the i through $i + k$th highest scoring terms till a query is found which returns a URL.

2.4 Language Filter

We pass each document fetched in response to a search query through a language filter. We used van Noord's TextCat implementation [19] of Cavnar and Trenkle's character n-gram based algorithm [4] which was shown to be over 90% accurate on a variety of languages and document lengths. We considered a document to belong to the target language if that language was top-ranked by the language filter. To estimate the performance of the filter for Slovenian, we asked a native speaker to evaluate 100 randomly selected Web-pages from a list of several thousand classified as Slovenian by the language filter. 99 of these were judged to in fact be in Slovenian, giving a precision of 99%. An analogous evaluation for Web pages judged to be negative shows that 90-95% of the pages classified as non-Slovenian were actually non-Slovenian. All our results are reported in terms of this automatic language classification and no further manual evaluation was carried out.

3. EXPERIMENTAL SET-UP

Our queries can be described by four parameters as detailed in Section 2.2. We set inclusion and exclusion parameters to the same value, i.e., the number of include terms is equal to the number of exclude terms. In addition the same method is used to select include terms as to select exclude terms, though scoring is based on relevant or non-relevant documents as appropriate. We varied the number of include/exclude terms from 1 to 10 and compared the following six term selection methods: *uniform*, *term-frequency*, *probabilistic term-frequency*, *rtfidf*, *odds-ratio*, *probabilistic odds-ratio*.

3.1 Using Existing Corpus-construction Techniques

Ghani and Jones [7] showed that single word queries were sufficient for finding documents in Tagalog, and that selecting the query-words according to their probabilities in the current documents performed the best. It is important to note that their experiments were run on a small corpus[1] of Tagalog documents and other distractor documents collected from the web and stored on disk. We compared their best-performing methods against other query generation methods and lengths, on the tasks of finding both Tagalog and Slovenian documents on the Web.

3.2 Using Search Engine's Built-in Features - "More Like This or Related Pages"

In order to establish whether search engines already incorporate a functionality which subsumes CorpusBuilder, we ran an experiment using AltaVista's "Related Pages" function.

We start with the same initial documents as in the systematic Slovenian experiments, and use CorpusBuilder query generation, setting include and exclude lengths to 5 and the term selection method to *odds-ratio*. Then we retrieve the highest ranking AltaVista hit classified as relevant by our language filter. We then query AltaVista using like:URL to find documents "like" the one we retrieved and add all of the relevant ones to our initial set.

We used *odds-ratio* with query length 5 as this method had been performing well in other experiments.

3.3 Initial conditions

For each document-based experimental condition, CorpusBuilder had access to a single example of a positive document in the target language, except in the user keyword experiment, described in more detail in section *3.3.2*. For most experiments we supplied four negative example documents, one each in English, Czech, Serbian and Croatian, though for the user keyword experiment 10 English stopwords constituted the initial negative document. The language model for the language filter was also supplied.

3.3.1 Choice of Initial Document

Since all the query generation methods described in this paper derive the query terms from documents already found, it is important to consider the effect of varying the initial document used. We used three different initial positive documents in Slovenian. The properties of these initial docu-

[1]the corpus consisted of 500 Tagalog documents and 15000 documents mostly in English and Brazilian Portuguese

ments are shown in Table 1. Document 2 was the document used to generate TextCat's Slovenian language model.

	Type	Topic	Length	Vocab Size
1	Formal	Horticulture	568	354
2	News	Politics	716	446
3	Informal	Personal Web-Page	90	74

Table 1: Description of the 3 different initial documents used in experiments

3.3.2 *Initializing with User-supplied Keywords*

It may not always be easy to find entire documents in a language to supply to our system. As an alternative to starting with a positive document in the target language, we may start with a few words in the target language obtained from the user or a native speaker. To this end we asked two native speakers of Slovenian to supply us with words. We asked three questions of each native speaker, to elicit initial words of varying types. We asked for ten *common* words, to obtain words a Slovenian may be likely to think of first. We then asked for ten *unique*ly Slovenian words, to get at a native speaker's intuitions about what words are particular to their language. Croatian and Czech, for example, share many strings and words in common with Slovenian. A native speaker of Slovenian may know words which are shared with other languages. They may, however, not know that some are shared with other languages. Finally we asked for ten words *useful* for finding texts in Slovenian on the web.

The words native Slovenian speakers supplied us with can be seen in Table 2. We used these words as mock initial documents, and 10 English stop-words as the initial negative document.

3.4 Comparison of Methods

In order to compare the term selection methods, we used each in isolation for a number of different query lengths. We compared them in terms of both proportion of documents in the target language, and number of target documents as a proportion of Web accesses. For these experiments Slovenian was the target language.

3.5 Generalizing across Languages

Systematic experiments with Slovenian by themselves do not provide any evidence that our techniques can be generalized to other languages. To test the hypothesis that our approach will indeed perform well on many languages, we repeat some of the experiments for Tagalog, Croatian and Czech.

4. RESULTS

The goal of our system is to collect as many documents in the target language as possible while minimizing the cost. The evaluation measures we used were (a) percentage of documents retrieved in the target class (`PosDocs`) and (b) the number of documents in the target class per unique web query (`PosQueries`). The number of queries issued is defined as the number of times a new query is issued to the search engine AltaVista. We compared the term selection methods according to these two performance measures for each length independently.

Speaker 1	
common	da, pa, in, je, bi, si, bo, a, se, ki
unique	jaz, hisa, ogenj, dez, gozd, hlod, zlikrofi, struklji, okno, cevelj, najin, vajin, njun
useful	splet, stran, podjetje, vsebina, vsak, ker, miza, hisa, kazalo, povezava
Speaker 2	
common	janez, delo, hribi, omara, promet, sonce, papir, stena, ker, hrana
unique	hrepenenje, karkoli, strani, enajst, zoprno, trkanje, uporabljati, splet, kozolec, navodilo
useful	izmenjava, zoprno, karkoli, cvetje, velikanski, zmeda, gostilne, prehajanje, obsijal, gozdar

Table 2: Words supplied by native Slovenian speakers for Use in Automatic Query Construction. Speaker 1 provided function words as *common*, while Speaker 2 provided common nouns and names such as "John" (janez), and "work" (delo). For *unique* words, both speakers provided the names of traditional Slovenian foods and artifacts. Additionally Speaker 2 also provided uniquely Slovenian words that are not likely to appear in other languages, such as "desire" (hrepenenje), "disagreeable" (zoprno) and "knocking" (trkanje). For *useful* words, Speaker 1 provided internet and computer-related terms, such as Slovenian for "Web" (splet), "page" (stran), "content" (vsebina) and "directory" (kazalo), while Speaker 2 provided words useful for finding different topics on the Web, such as Slovenian for "exchange" (izmenjava), "flowers" (cvetje), "restaurants" (gostilne).

4.1 Using Existing Corpus-construction Techniques

Applying single-word *term-frequency* and *probabilistic term-frequency* queries to the Web for Slovenian results in relatively low precision, as shown in Figure 1. Using the *odds-ratio* query generation method described in section 2.2 outperforms the *probabilistic term-frequency* approach with single include and exclude-word queries. Furthermore, using more words in the query (3 for inclusion and 3 for exclusion) performs better than the single word queries previously used. To establish whether the better performance of *odds-ratio* was due to the choice of language, we performed the same evaluation on Tagalog. Although fewer documents were found overall, the trend remained the same, with OR-3 (Odds-Ratio with 3 inclusion and 3 exclusion terms) finding over 600 documents in Tagalog after examining 1000 documents, while OR-1, PTF-1 and TF-1 found just over 300, 122, and 17 documents respectively.

4.2 "More Like This"

As we can see in Figure 2, our proposed odds-ratio method outperforms "Related pages" function of Altavista. It finds a document in the target language around 33% of the time, while using a query based on *odds-ratio* with query-length 5 finds a document in the target language around 90% of the time.

4.3 Choice of Initial Documents

Figure 3 shows the number of documents in Slovenian found, plotted against the number of documents examined. It can be seen that the method performs comparably for all three initial documents. At the time the results diverged, over 1000 positive documents had been found.

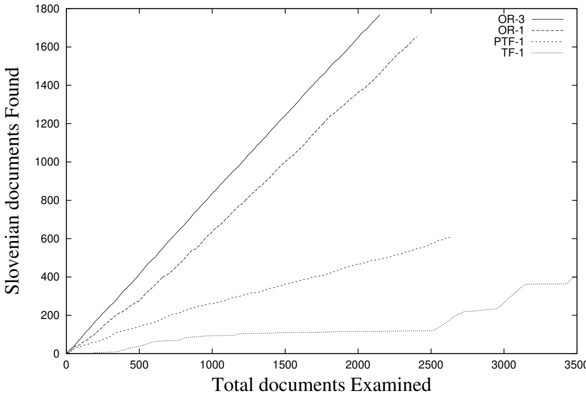

Figure 1: On Slovenian *term-frequency* (TF) and *probabilistic term-frequency* (PTF) term selection methods with one inclusion and one exclusion term were outperformed by *odds-ratio* (OR) with one inclusion and one exclusion term, and an even better-performing *odds-ratio* with 3 inclusion and 3 exclusion terms

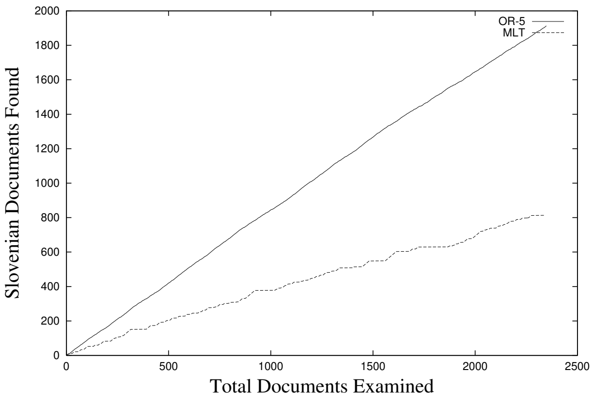

Figure 2: Using "More Like This" finds a document in the target language around a third of the time, while using a query based on *odds-ratio* with query-length 5 (OR-5) finds a document in the target language around 90% of the time.

4.4 Initializing with User-supplied Keywords

Using the sets of 10 keywords supplied by native Slovenian speakers given in Table 2 as initial positive information, and 10 English stop-words as initial negative information performs as well for initialization as using whole documents. It is interesting to note that the 10 English stop-words work as well as using documents from closely related languages as the initial negative document set. Regardless of the starting condition the search resulted with queries bringing in Slovenian documents 80% of the time, when OR-3 was used as the query-generation method.

4.5 Term Selection Methods

We systematically compared the term selection methods for each length independently. The comparison of different term selection methods given in Table 3 gives values after retrieving 3000 documents and values for 1000 issued queries, unless the result is marked by *, where the values are given at much lower number of documents. For instance, for PO

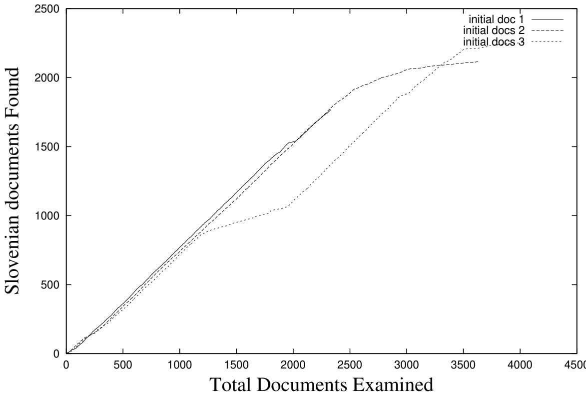

Figure 3: For 3 different initial positive documents, *odds-ratio* with length 3 performs with similar results. Differences emerge long after the initial document is dwarfed by the presence of over 1000 other documents.

and UN on length 5, less than 100 documents and for length 10 less than 5 documents. This shows that the two methods are mostly issuing queries which do not return any document (especially for longer queries) but the few successfully queries usually return the target class documents.

When observing the number of queries issued, *odds-ratio* (OR) finds the greatest number of target documents. In terms of the number of documents examined, *odds-ratio* is again the best (length 1 — 3) except when all methods have about the same performance (length ≥ 4).

The two probabilistic methods (PTF and PO) select among terms randomly with probability proportional to scores assigned by term-frequency and odds-ratio respectively.

Detailed results are shown in Figures 4 and 5. Note the differences in scale on the graphs in Figure 4; longer queries are much more likely to return no documents at all, and so can be costly. *odds-ratio* performed best for all query-lengths except 5, where *term-frequency* found many documents with few queries, contrasting with query-length 1, where it found the least.

When observing performance for the same term selection method with different query lengths, we found differences between the methods. For *term-frequency*, the best performance is achieved with length 4 (when 4 terms are included and 4 terms are excluded). For *probabilistic term-frequency* and *odds-ratio* length 3 gives the best results, while for *probabilistic odds-ratio* using more than 1 include and 1 exclude term gives a very small number of the target language documents while using a very high number of queries.

4.6 Generalizing to other languages

As seen in Table 4 *odds-ratio* found more target documents than both *term-frequency* and *probabilistic term-frequency* for the same number of total documents examined, for all of Slovenian, Croatian, Czech and Tagalog. This suggests that the superiority of this query-generation mechanism generalizes across languages. Table 5, however, shows that the number of queries is dependent on the target language and method. This may reflect the number of web-pages which are confusable between Slovenian, Czech and Croatian when queries are made with frequent-words, whereas Tagalog's frequent words are more unique on the Web.

Len	Methods ordered by their performance	
	wrt PosDocs measure after retrieving 3000 docs	wrt PosQueries measure after issuing 1000 queries
1	68.8%(OR)>46%(PO)>39%(UN)>19.1%(PTF)> 8.9%(TF)	1.6(OR)>0.43(PO)>0.36(UN)>0.25(PTF)>0.09(TF)
3	82.3%(OR)>65.8%(UN)*>64.1%(PTF)>33.2%(TF)>9.4%(PO)*	1.77(OR)>0.38(PTF)>0.18(TF)>0.094(PO)>0.035(UN)
5	92.4%(UN)*>92.3%(PO)*>81.5%(OR) >77.4%(PTF)>77.0%(TF)	1.2(OR)>0.53(TF)>0.14(PTF)>0.01(PO)>0.01(UN)
10	100%(PO)*>88.7%(PTF) >79.2%(OR)>50%(UN)*>7.0%(TF)	0.02(OR)>0.01(PTF)>0(TF)>0(PO)>0.001(UN)

Table 3: Comparison of different term selection methods for query length varied from 1 to 10.

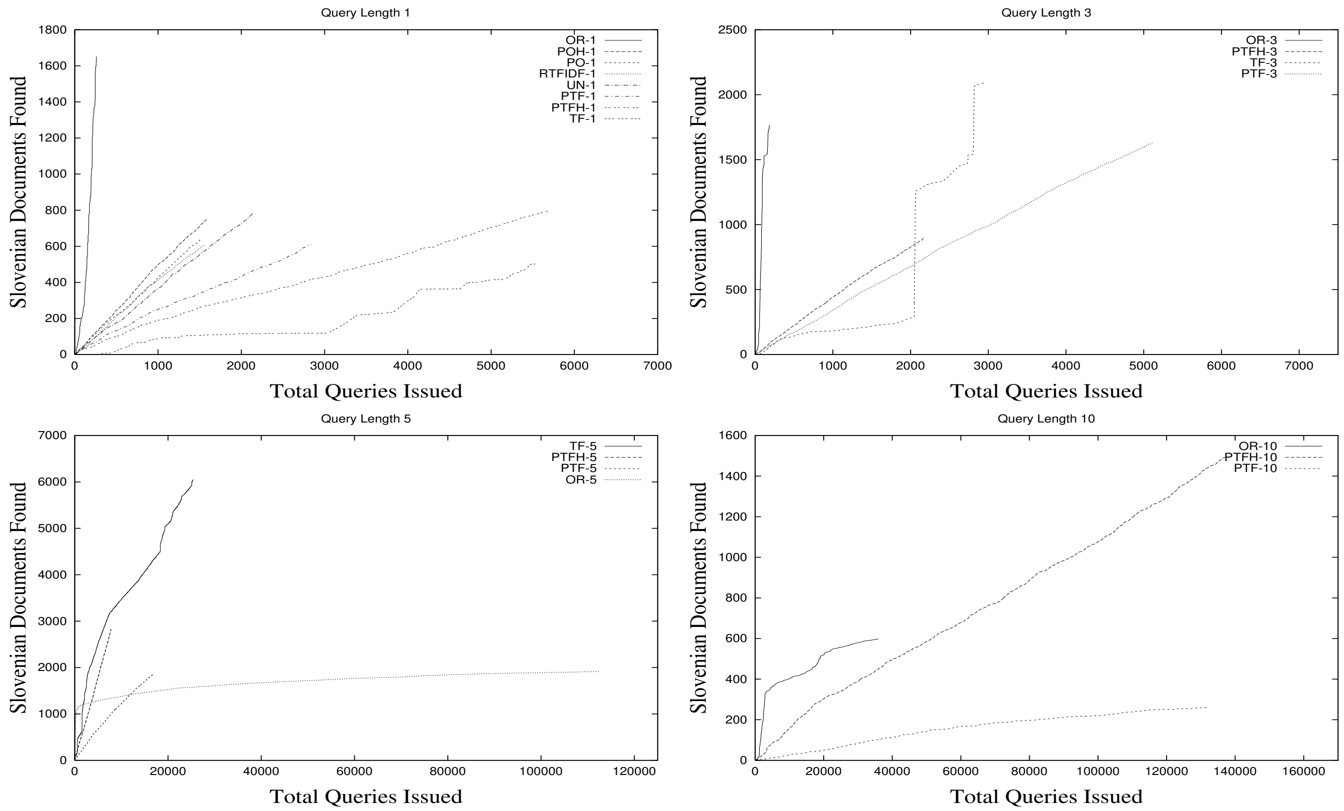

Figure 4: Comparison of different term selection methods for query lengths of 1, 3, 5, and 10 include and exclude terms, with cost function the number of queries. *odds-ratio* consistently finds more documents given the same number of queries, except for length 1, where *most-frequent* finds high yield queries. Note the differences in scales on the graphs; longer queries are much more likely to return no documents at all, and so can be costly. Methods are shown only if the number of documents found exceeds 200.

Method	Language	Target Docs at 1000 Total Docs
TF-3	Slovenian	178
PTF-3	Slovenian	646
OR-3	Slovenian	**835**
TF-3	Croatian	39
PTF-3	Croatian	410
OR-3	Croatian	**677**
TF-3	Czech	385
PTF-3	Czech	451
OR-3	Czech	**743**
TF-3	Tagalog	440
PTF-3	Tagalog	359
OR-3	Tagalog	**664**

Table 4: *odds-ratio* found more target documents than both *term-frequency* and *probabilistic term-frequency* for the same number of total documents examined, for all of Slovenian, Croatian, Czech and Tagalog.

Method	Language	Docs	Typical Query
TF-3	Slovenian	182	+in +v +za -html -a -re
PTF-3	Slovenian	342	+v +bolj +podrocju -carapama -den -jest
OR-3	Slovenian	**1409**	+torej +bomo +nisem -attila -laszlo -kerdes
TF-3	Croatian	61	+i +u +za -de -o -a
PTF-3	Croatian	231	+su +u +osobito -vozil -iso -pripad
OR-3	Croatian	93	+allah +takodjer +uzviseni -ief -mgtf -summary
TF-3	Czech	286	+na +v +pro -re -the -a
PTF-3	Czech	198	+v +na +s -zakroky -za -ga
OR-3	Czech	**680**	+vldy +mt +tden -found -davky -dite
TF-3	Tagalog	**793**	+sa +ng +ang -the -to -a
PTF-3	Tagalog	262	+unang +ang +dati -obey -optik -beef
OR-3	Tagalog	236	+niya +walang +siya -i -za -server

Table 5: Number of target documents found at 1000 queries. *odds-ratio* found more documents than both *term-frequency* and *probabilistic term-frequency* for the same number of queries, for Slovenian and Czech. Typical queries shown are those that most commonly found positive documents in our experiments. For PTF-3 every query was unique; a randomly selected one is shown.

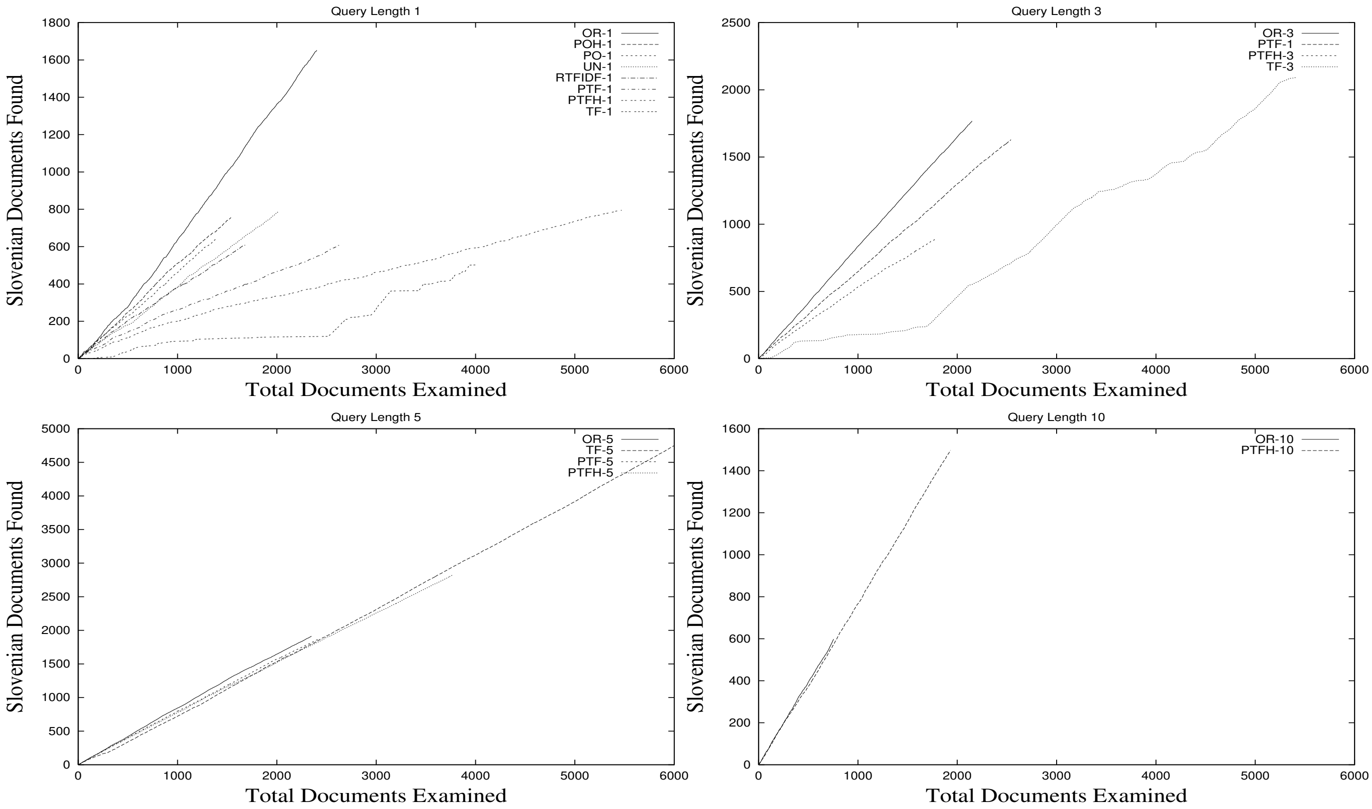

Figure 5: Comparison of different term selection methods for query lengths 1, 3, 5, and 10 include and exclude terms, measured against total documents examined. *odds-ratio* is consistently the most precise in finding Slovenian documents. Precision increases with the number of query terms, though many query-methods are able to find very few total target-language documents when using long queries. Methods finding less than 300 Slovenian documents are omitted.

5. DISCUSSION

Our approach performs well at collecting documents in a minority language starting from a few words or documents but it does require a language filter for that minority language. There are filters available for quite a few languages (van Noord's TextCat [19] has models for over eighty languages), but this is potentially a limitation of our approach. In earlier work [7], we experimented with constructing a filter on-the-fly, starting from the initial document and bootstrapping, and our experiments with Tagalog yielded encouraging results.

We evaluate the corpus collected through our approach by sampling the corpus and verifying the decisions made by the language filter. This gives us information about the precision of our corpus. In general on the web we cannot assess coverage, though we could measure the rate at which we find new Slovenian documents as our experiments progress and a decreasing rate would give us a bound on the number of documents we can find using our methods. Callan et al. [3] and Ghani and Jones [7] use the measures percent vocabulary coverage and cumulative term frequency to evaluate the coverage of their language models, as they have access to the entire experimental corpus. Although our task is not to construct a language model for Slovenian and we do not have the "true" model to compare against, we can still use these measures and calculate of rate at which we add new vocabulary words and the distribution of the words we have in our vocabulary. Since we are sampling in the space of Slovenian words, convergence of the sampled distribution would indicate a reasonable coverage.

We utilize the resources of a search engine and focus on the task of generating queries that are highly accurate and cover different parts of the language space we are dealing with. There are several other ways that we could collect corpora from the Web. Using a dictionary of the minority language and manually selecting function words to query and then iterating a few times to eliminate function words that are shared with other languages is one approach that would work well but would require domain knowledge in the form of knowledge about function words for the particular language. Our approach learns from examples in that it only requires a handful of documents and requires very little domain knowledge. We could also use a spider that crawls all pages under the domain names of the countries where that language is spoken. For example, in the case of Slovenian, we could spider all web pages in the .si domain. This would not result in good coverage or precision since not all the pages under the .si domain are in Slovenian and there are Web pages outside the .si domain that are in Slovenian. In our experiments, we found approximately 31,000 Web pages under the .si domain, 24% of them being classified as not in Slovenian. Similarly, out of the 30,000 Slovenian documents we found, 22% of them were not under the .si domain.

Another way of collecting a Slovenian corpus would be to crawl all the pages starting from the Regional → Countries

→ Slovenia category in Yahoo (or the category of the country where the language is spoken). This alone would not be accurate because Yahoo does not have an exhaustive list of Slovenian Web sites in general and of Web pages in Slovenian in particular. However, a combined approach, where the current method is used to locate seed pages for a crawler might be interesting to consider.

One promising alternative we compared our approach with was using the built-in functionality of looking up "similar" web-pages that some search engines provide and as shown in Section 4.2, our approach outperformed this alternative.

6. CONCLUSIONS

We have presented a system for automatically generating Web search queries to build a corpus for a minority language and showed that by using a variety of term selection methods and query lengths, we can generate queries that perform well at collecting corpora for specific languages. In particular, we show that selecting terms according to their *odds-ratio* scores works well for several languages and a wide variety of initial documents or keywords supplied by the user result in similar performance.

Our approach for automatic query-generation is useful for any application where we are able to build a very accurate, but expensive classifier. We can then use the search engine as a simpler filtering mechanism, bringing in only those documents likely to satisfy our classifier, thus saving us the expense of running the classifier on all documents retrieved by the crawler. The kinds of high-precision classifiers which would benefit from this approach include systems with parsers, systems which fetch more web-pages to decide on the classification for a given web-page and systems which involve user input thus making our approach applicable in various useful and important areas.

Acknowledgments

The authors were supported in part by the NSF LIS Grant REC-9720374.

7. REFERENCES

[1] D. Boley, M. Gini, R. Gross, E.-H. S. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moor. Document categorization and query generation on the world wide web using webace. *AI Review*, 13(5).

[2] P. Brown, S. D. Pietra, V. D. Pietra, and R. Mercer. The mathematics of statistical machine translation. *Computational Linguistics*, 19(2), 1993.

[3] J. Callan, M. Connell, and A. Du. Automatic discovery of language models for text databases. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 479–490, Philadelphia, 1999. ACM.

[4] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *Proceedings of Third Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, NV, 11-13 April 1994.

[5] Z. Chen, X. Meng, B. Zhu, and R. H. Fowler. Websail: From on-line learning to web search. In *Proceedings of the 2000 International Conference on Web Information Systems Engineering*, 2000.

[6] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori. Focused crawling using context graphs. In *26th International Conference on Very Large Databases, VLDB 2000*, pages 527–534, Cairo, Egypt, 10–14 September 2000.

[7] R. Ghani and R. Jones. Learning a monolingual language model from a multilingual text database. In *Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM 2000)*, 2000.

[8] R. Ghani, R. Jones, and D. Mladenić. Building minority language corpora by learning to generate web search queries. Technical Report Technical Report CMU-CALD-01-100, Carnegie Mellon University, Center for Automated Learning and Discovery, 2001.

[9] E. Glover, G. Flake, S. Lawrence, W. P. Birmingham, A. Kruger, C. L. Giles, and D. Pennock. Improving category specific web search by learning query modifications. In *Symposium on Applications and the Internet, SAINT*, San Diego, CA, January 8–12 2001.

[10] A. R. Golding and D. Roth. A winnow-based approach to context-sensitive spelling correction. *Machine Learning*, 34(1-3):107–130, 1999.

[11] D. Haines and B. Croft. Relevance feedback and inference networks. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1993.

[12] F. Jelinek. *Statistical Methods for Speech Recognition.* MIT Press, 1999.

[13] M. Liberman and C. Cieri. The creation, distribution and use of linguistic data. In *Proceedings of the First International Conference on Language Resources and Evaluation*, 1998.

[14] D. Mladenić and M. Grobelnik. Feature selection for unbalanced class distribution and naive bayes. In *Proceedings of the 16th International Conference on Machine Learning (ICML 1999)*, 1999.

[15] J. Rennie and A. McCallum. Using reinforcement learning to spider the web efficiently. In *Proceedings of the 16th International Conference on Machine Learning*, 1999.

[16] P. Resnik. Mining the web for bilingual text. In *Proceedings of 34th Annual Meeting of the Association of Computational Linguistics*, Maryland, 1999.

[17] S. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.

[18] J. J. Rocchio, Jr. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall, 1971.

[19] G. van Noord. Textcat. http://odur.let.rug.nl/ vannoord/TextCat/.

[20] Y. Yang and J. Pederson. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML 1997)*, 1997.