Information Retrieval Based on Context Distance and Morphology

Hongyan Jing
Department of Computer Science
Columbia University
New York, NY 10027, USA
hjing@cs.columbia.edu

Evelyne Tzoukermann
Bell Laboratories - Lucent Technologies
700 Mountain Ave.
Murray Hill, NJ 07974, USA
evelyne@research.bell-labs.com

Abstract

We present an approach to information retrieval based on context distance and morphology. Context distance is a measure we use to assess the closeness of word meanings. This context distance model measures semantic distances between words using the local contexts of words within a single document as well as the lexical co-occurrence information in the set of documents to be retrieved. We also propose to integrate the context distance model with morphological analysis in determining word similarity so that the two can enhance each other. Using the standard vector-space model, we evaluated the proposed method on a subset of TREC-4 corpus (AP88 and AP90 collection, 158,240 documents, 49 queries). Results show that this method improves the 11-point average precision by 8.6%.

1 Introduction

Information Retrieval typically measures the relevance of documents to a query based on word similarity. The basic underlying assumption is that the same word form carries the same semantic meaning. Stemming, as a recall enhancing engine, reduces morphological variants to the same root. On one hand, stemming builds more links between words and as a result, retrieves more related documents; on the other hand, it can also build links between irrelevant words. In contrast, sense-based retrieval is a precision enhancing procedure; it links words based on their semantics. The problem we address in this paper consists of integrating these two opposite approaches so that they can enhance each other. As the result of the integration, more links are added between morphologically related words, and at the same while, false links between morphological relevant but semantic irrelevant words are avoided.

We present a context distance and morphology based retrieval strategy. The context distance model aims to tackle word polysemy problem in retrieval so that we can correlate words based on their meanings rather than surface forms. A linguistic morphological analyzer is used to replace a traditional stemmer [Porter1980, Lovins1968]. The semantic-based context distance model and the morphological processing are integrated in the retrieval stage for determining the semantic equivalence between words in a query and words in documents. The experiments on a sub-collection of TREC-4 corpus show an improvement of 8.6% on the 11-point average precision by the proposed method.

In section 2, we present the proposed approach, describing the context distance model and its integration with morphology. We also introduce how global corpus information and local document information is used in the proposed approach. In section 3, we describe the experiment on TREC corpus and present case studies. Then, we discuss related work in stemming and sense-based retrieval and analyze the problems in traditional techniques. In the last section, we conclude by discussing future work.

2 The retrieval model based on context distance and morphology

In our proposed approach, a word is assumed to have a dominant meaning in a document [Gale et al.1992, Yarowsky1993]. We represent this meaning in the form of a context vector. The semantic closeness of two words is indicated by the distance between their context vectors. The distance is computed using a model based on global lexical co-occurrence information.

2.1 The context vector

We encode the semantic meaning of a word in a document using a context vector. The context vector in our model is based on all the occurrences of the same word in the document and the assumption is that a word has a dominant meaning through the document. This is in contrast to the context vector model used in Schütze and Pedersen (1995) and Schütze (1998), where the context vector represents the context of a single occurrence of a word in the document.

To compute the context vector for a target word, all candidate words which can possibly be included in the context vector are first collected. This is accomplished by extracting all the words which appear at least once in the local contexts of the target word in the document. In the experiment, a window of 10 words (five words on either side of the target word) is considered as local context. Then a weight is assigned to each collected word based on the following formulae: $Fr(I|T)/Fr(T)$, the frequency of the word I appearing in the window with the target word T divided by the term frequency of the target word. The purpose of this step is to measure the importance of each collected candidate word in the context of the target word. If there are more than 10 candidate words, the final context vector for the target word includes 10 candidate words with the highest weights. In the case of a tie score, a context word with a higher term frequency is selected. If these are less than 10 words in the candidate list, as in the case of short queries, all candidate words are included in the context vector. Therefore, the size of the vector is 10 or less. The context vector is then normalized. As a result, each of the words in the context vector will acquire a weight between 0 and 1. The more frequently a word co-occurs with the target word in the local contexts, the larger the weight.

We show below the context vector for the word *bank* from a sample document (AP881231-0128 in the TREC corpus):

Target word : bank
Context vector :
{ savings(0.44) federal(0.44) million(0.44)
loan(0.33) company(0.22) farmington(0.22)
board(0.22) agreed(0.22) billion(0.22)
nationwide(0.22) }

In this example, the target word *bank* appears 9 times in the document. The words *savings*, *federal*, and *million* have a higher weight than others in the context vector since they appear 4 times in the local contexts of the target word, while most of the other words occur 2 times. The words in the context vector are important for distinguishing the semantic meaning of the target word. For example, the words *(savings, million, loan, ...)* help to disambiguate the target word *bank* as the money *bank* rather than the river *bank*. The weight associated with each word in the context vector indicates the importance of the word in the context vector.

2.2 The distance between context vectors

The computation of context distance is based on the mutual information between words in context vectors. To measure the mutual information between two given words, we rely on their co-occurrence information in the corpus. The corpus we used for this computation is the TREC AP88 collection (79,919 documents, 0.24 GB). We use a measure called *corpus relevance* to represent the mutual information between two words. The corpus relevance of two words is precomputed before retrieval, as shown in the following:

$$R(I_1 I_2) = \frac{DF(I_1 I_2)}{DF(I_1) + DF(I_2) - DF(I_1 I_2)}$$

that is, the frequency of two words appearing in the same document in the corpus divided by the frequency of at least one of the two words appearing in the corpus. DF here represents document frequency. The purpose is to use co-occurrence information in the corpus to measure the mutual information between two words. The corpus relevance between two words is a value between 0 and 1. A value of 1 indicates that two words always occur in the same documents in the corpus; a value of 0 indicates they never occur in the same document. Figure 1 shows some sample word pairs with high corpus relevance scores and also some sample pairs with low corpus relevance scores.

Word Pairs	Corpus Relevance
gaza, palestinians	0.600
nyse, dow	0.571
composite, dow	0.537
wheat, grain	0.443
south, year	0.117
food, told	0.052
miles, people	0.051

Table 1: Sample word pairs and their corpus relevance

We consider this corpus relevance score as an indication of relatedness between two words. Words that are more related tend to have a high corpus relevance score, for example, *NYSE* and *Dow*. This information from the corpus analysis provides us with useful word correlations which may not even be present in lexicons.

The distance between two context vectors are computed in two steps. First, we determine the corresponding relations between words in two context vectors. Suppose context vector CV_1 consists of 10 words: $A_1, ..., A_{10}$, and context vector CV_2 consists of 10 words: $B_1, ..., B_{10}$, we look up the pre-computed corpus relevance value and find the corpus relevance for every pair (A_i, B_j), for i, j=1..10. We then sort the 100 corpus relevance values in descending order. The selection of the corresponding pairs starts from the pair with the highest corpus relevance and each word is matched only once. When this step is finished, each word in one context vector will be matched to a word in the other context vector. We represent this matching as $A_i \rightarrow B_{m(i)}$, where $m(i)$ means the match of i, for i=1..10. For example, if A_1 is matched to B_3, then $i = 1, m(i) = 3$.

In the second step, we compute the distance between the two context vectors based on the matching in the first step and the pre-computed corpus relevance. If we represent the two context vectors as:

$$CV_1 = \{A_1(W_{1,1}), ..., A_{10}(W_{1,10})\}$$

$$CV_2 = \{B_1(W_{2,1}), ..., B_{10}(W_{2,10})\}$$

where A_1 to A_{10} and B_1 to B_{10} are the 10 words in the two context vectors respectively, $W_{i,j}$ is the weight for the j-th word in the i-th vector. Suppose A_i is paired with $B_{m(i)}$ in the first step, the context distance is computed as follows:

$$Dist(CV_1, CV_2) = \sum_{i=1}^{10} (R(A_i, B_{m(i)}) \times W_{1,i} \times W_{2,m(i)})$$

where $R(A_i, B_{m(i)})$ means the corpus relevance of A_i and $B_{m(i)}$. The computed context distance is a value between 0 and 1. The higher the value, the closer the two contexts vectors. We then compute the *average distance* between two context vectors by dividing the computed distance with the vector size, which is 10 in this case.

If CV_1 or CV_2 has less than 10 elements, the computation is basically similar to the above process except that the vector size equals to the minimal of CV_1 size and CV_2 size rather than the standard size of 10. The average distance is computed by dividing the computed distance with the actual vector size.

There are several reasons why we designed this model to compute the distance between context vectors. First of all, we observed that the semantic closeness of two contexts is not always demonstrated by the presence of the same words, but often by the presence of related words. For example, the word *bank* may occur with the word *money* in one context, and with the word *loan* in the other. If we can capture the close relatedness of *money* and *loan*, we can deduce that *bank* probably has similar meanings in the two occurrences. A model which relies on exact word repetition will fail in this case since it will miss the relations between *money* and *loan*. The kind of lexical relations that exist between the words such as *money* and *loan* or *eat* and *fork* is often not present in existing lexicons. However, lexical co-occurrence information to some degree indicates such correlations. The co-occurrence information has been successfully used for sense disambiguation and query expansion [Schütze and Pedersen1995, Schütze1998, Li and Abe1998, Buckley et al.1994].

2.3 Integrating context distance with morphology

Relating morphological variants enhances recall while sense based retrieval improves precision. To make the best of the two, we integrate the semantic-based context distance model with morphological information. For a word in a query, we not only compare it with the same word form in the document, but also with the other morphologically related words in the document. If the context vectors of two morphologically related words are close enough, then the two related words will be equated. This brings us the benefit of a stemmer but avoids the problem of over-generalizing.

Morphological relations are extracted from the CELEX [CELEX1995] lexical database. Inflectional variants were acquired from the CELEX English morphology lexicon for words directly, and derivational variants were extracted from the CELEX English morphology lexicon for lemmas by grouping words with the same lemma in a derivational family. A total of 52,447 words (not including inflected forms) were grouped into 17,963 derivational families. A sample derivational family is *(adjust, adjustment, adjuster,...)* or *(private, privacy, privateer, privatize, privatization,...)*.

2.4 The retrieval algorithm

The system consists of the preparation stage and the retrieval stage. In the preparation stage, we build the morphological databases, pre-compute context vectors, and pre-compute corpus relevance. In the retrieval stage, the documents and the queries are first indexed as usual. Before computing the similarity of two documents using a traditional vector model as in SMART system [Buckley et al.1994], we compute the context distances between a word in a query and its morphologically related words in a document, using the algorithm we have introduced above. If the context vector of the word in the query is close enough to that of its morphologically related word in the document, the two words will be considered equivalent; otherwise, they will be considered different even if they have the same word form. We show some examples in the next section. If the context vector for the word in the query has a very small size, as in the case of short queries, the query word is considered equivalent to its morphologically related words in the document regardless of the context distance between them, since the context vectors have too few words to reliably indicate the meaning of the query word. The algorithm is summarized as follows:

Preparation:

Step 1: Build morphological databases using the CELEX database.

Step 2: Compute context vectors for words in a document.

Step 3: Compute lexical co-occurrences in the corpus and corpus relevance values for each word pair in the corpus.

Retrieval:

Step 4: Index the corpus.

Step 5: For each query and each document:

5.1 Compute the average context distance between the context vector of a word in the query and those of its morphological variants in the document

5.2 If ((the average context distance > the distance threshold) or (the size of the vector is too small))
then
consider the two words as the same term
else
consider the two words as different terms

5.3 Compute the similarity of the query and the document.

Recall	**Precision**		
	No-stem	Stem	New Method
0.00	0.419	0.466	0.489
0.10	0.282	0.306	0.333
0.20	0.224	0.256	0.279
0.30	0.183	0.226	0.244
0.40	0.162	0.193	0.212
0.50	0.135	0.169	0.179
0.60	0.107	0.146	0.156
0.70	0.080	0.122	0.129
0.80	0.050	0.081	0.106
0.90	0.029	0.052	0.064
1.00	0.013	0.024	0.035
Average	**0.154**	**0.186**	**0.202 (8.6%)**

Table 2: The evaluation results

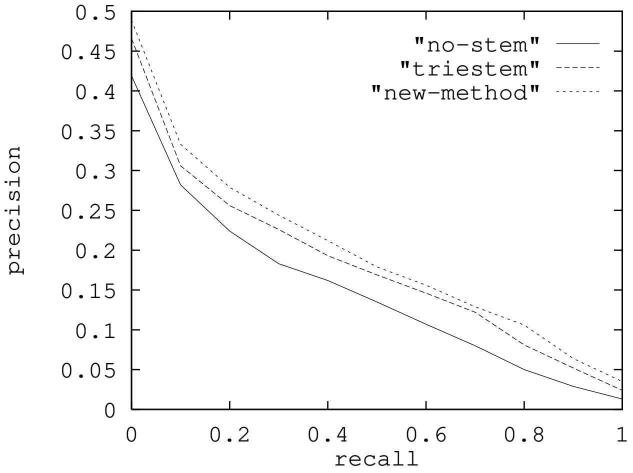

Figure 1: Performance comparison between Lovins's and the new method.

3 Experiments and results

We tested the proposed approach on a sub-collection of TREC-4 corpus. The documents we used are the AP88 and AP90 newswire, consisting of 158,240 documents (0.49GB). We used 49 queries of TREC-4 (query numbers 202-250). Retrieval is based on the standard vector similarity model using SMART from Cornell University (Version 11.0) [Buckley et al.1994]. We used the augmented term frequency/inverse document frequency weighting (augmented tf/idf) for computing document vectors.

There is one parameter to adjust in the proposed approach: the threshold of context distance for relating one word with the other. We used 15% of documents as training set for adjusting this context distance threshold parameter. We then retrieve on the whole collection. To compare with other systems, we also performed the same retrieval task using SMART system. We did two runs: one without stemming and one with stemming. The stemmer used in the second experiment is triestem, which is provided by SMART and is a modified version of Lovins's stemmer [Lovins1968]. Compared with the result using stemming, the proposed method achieved an improvement of 8.6% on average precision for 11 points of recall (from an average precision of 0.186 to an average of 0.202). Compared with the no-stemming baseline, the proposed method achieved an improvement of 31.2% on average precision. Table 2 shows the detailed results. Figure 1 shows the results in a graph format.

We also analyzed the performance for individual queries. First, we compared the proposed method and the traditional stemmer on their influence on individual queries. By looking at the results of the two runs on SMART, we can see that traditional stemming using a stemmer like triestem greatly improved the overall performance in this experiment, from 0.154 average precision to 0.186. Although applying traditional stemming greatly improved the overall performance, it also decreased the performance of a number of queries: 15 out of 49 queries(31%) had a decrease in performance after the traditional stemming was used. Our context distance and morphology based model, in contrast, resulted in performance decrease on only 7 queries, half of the number compared with traditional stemming, while improving the overall performance. We also found that the proposed method are more likely to outperform traditional stemming for longer queries.

Compared to the standard retrieval procedure by SMART, the new approach takes up more space by storing context vectors and corpus relevance values. It also takes more time due to three additional actions: the pre-computing of context vectors, the pre-computing of corpus relevance, and the computing of context distance during retrieval. Hopefully, the problems with time and space can be alleviated by more efficient data storing and computing. The investigation of this issue is in our future work.

To examine whether the computed distance between context vectors provides useful results, we studied a random set of cases. The program correctly identified the semantic closeness between the following two context vectors (the two context vectors have a distance of 0.03012 – the relative large value means they are close):

bank : { savings(0.44) federal(0.44) million(0.44)
loan(0.33) company(0.22) farmington(0.22)
board(0.22) agreed(0.22) billion(0.22)
nationwide(0.22) }
banks { fdic(0.56) depression(0.42) number(0.28)
failed(0.28) post(0.28) fund(0.28) year(0.28)
fslic(0.28) loan(0.14) deposits(0.14) }

Note that the two contexts have only one overlapping words. A vector model solely based on word similarities will fail to find the high relevance between the above two context vectors, while our context distance model does capture such relatedness.

The program also correctly identified the non-relatedness of the following two context vectors (the context vectors have a distance of 0.0002 – the small value means that they are not related):

bank : { savings(0.44) federal(0.44) million(0.44)
loan(0.33) company(0.22) farmington(0.22)
board(0.22) agreed(0.22) billion(0.22)
nationwide(0.22) }
bank : { west(0.45) gaza(0.45) strip(0.45)
state(0.22) boasted(0.22) called(0.22)
hailed(0.22) israeli(0.22) plo(0.22)
occupied(0.22) }

4 Discussion on stemming and sense-based retrieval

Our work of integrating context distances and morphology is related to the research on stemming and sense-based retrieval. In this section, we discuss related work in the two areas and analyze why the integration of the two methods improves retrieval. We also analyze some drawbacks present in the current approaches and discuss how the proposed approach tries to overcome these problems.

4.1 Stemming

Stemming conflates morphologically related words to the same root, either by a traditional stemmer such as Porter's [Porter1980] or Lovins's [Lovins1968], or by a linguistically-based morphological analyzer. Different studies showed inconsistent results of the effect of using stemmers. Harman (1991) showed that stemming provides no improvement over no-stemming at all, and different stemming algorithms make no difference either; Krovetz (1993) showed that stemming does help, and that the improvement is between 1.3% to 45.3% for different test collections and stemmers; a more recent large-scale analysis by Hull (1996) concluded that "some form of stemming is almost always beneficial, but the average absolute improvement due to stemming is small, ranging from 1 to 3%."

Two useful observations have been made in these studies. First, although the overall improvement of stemming seems insignificant, all experiments showed that it does greatly help certain individual queries; however, degradation in other queries may cancel out such improvement in overall results [Hull1996, Krovetz1993]. This implies to us that correlating morphologically related words could be potentially very useful, if we can somehow distinguish the cases in which it helps and in which it degrades, and therefore apply stemming only to positive cases.

The second observation is that, although stemmers are applied to words, semantic correlations exist only between particular meanings of morphological variants [Krovetz1993, Church1995]. For example, given the sentence *The waiter served us dinner*, it would be better to link the word *served* with the word *server* in the sentence *The server brought us the food*, but not the word *server* in the sentence *Client-Server architecture is promising.*

This second observation partially explains to us why the phenomena in the first observation happened. Traditional stemmers stem words without considering the specific words and the specific senses of the words involved. In some cases, queries are retrieved with more accuracy because the morphological variants happen to be also semantically related. In other cases, queries are retrieved with less accuracy because the morphological variants are not semantically related, thus stemming introduces noises in the statistical count. Since the semantic relatedness of morphological variants depends on the specific corpus studied, this might be one of the reasons why different studies showed very inconsistent results.

This analysis leads us to think that if we can integrate traditional stemming with a sense-based retrieval strategy, we may achieve a better result than the case when either method is used alone. This is the reason why we pursued the proposed approach. Corpus-based stemming [Xu and Croft1998] in some way is in the same direction, in that words are stemmed based on their correlations in the corpus rather than considering only their word forms.

4.2 Sense-Based Retrieval

The role of word sense disambiguation in information retrieval has been studied by several researchers. Krovetz and Croft [Krovetz and Croft1992] showed that sense disambiguation does not result in as much improvement in the top ranked documents, when we have moderate length queries and documents. The reason is that word collocation has partially reduced word ambiguities in the top ranked documents. Voorhees [Voorhees1993] showed that a simple word disambiguation technique based on taxonomic relations is not sufficient for retrieval; it unfortunately caused a decrease in performance. Schütze and Pedersen [Schütze and Pedersen1995] demonstrated that their clustering-based disambiguation model can effectively improve the retrieval performance by 7% and 14% on average. The work on Latent Semantic Indexing [Deerwester et al.1990] also uses semantic structures to improve terms found in the queries. It deals with the synonymy problem and offers a partial solution to the polysemy problem.

We believe that the kind of sense disambiguation we need for retrieval is different from the general sense disambiguation task as studied in many previous work [Yarowsky1992, McRoy1992, Ng and Lee1996]. The fundamental reason is that the underlying assumptions of traditional sense disambiguation do not fit for retrieval applications. Two underlying assumptions of traditional sense disambiguation are: fixed number of senses per word, and one sense per occurrence. We believe these assumptions have detrimental effects for retrieval.

4.2.1 Fixed number of senses per word

Traditional sense disambiguation uses predefined word senses as standards. The senses usually come from a static, pre-compiled lexicon such as WordNet or LDOCE (Longman Dictionary Of Contemporary English). A word is assumed to have a fixed number of senses as defined in the lexicon. This is problematic in two ways. First, a word in a collection could be used in a sense not covered by the lexicon. For example, *Java* is only listed in the sense of *coffee* in WordNet, while its meaning as a programming language which is frequently used in computer science related articles is missing. In this case, a disambiguation program dooms to fail even before it starts. Second, a word tends to be invoked only in one or a few specific meanings in a particular domain. A large percent of word senses predefined in a lexicon might not be used at all. Considering all predefined senses not only consumes resources but also complicates the disambiguation task by raising the chances of making wrong decisions. A corpus based approach is more useful for unrestricted text retrieval since it avoids the above two problems.

4.2.2 One sense per occurrence.

The most damage to retrieval, however, comes from the second assumption of traditional disambiguation: one sense per occurrence. For the same word, different lexicons may provide different number of senses. The corresponding relations between the senses defined in different lexicons for the same word are not clear. One sense in lexicon A may correspond to two senses in lexicon B, or it may correspond to part of sense (a) and part of sense (b) in lexicon B. This shows that the distinctions between senses are not absolute. Two different senses of a word may be semantically distinctive, as *bank* in the sense of river bank and *bank* in the sense of a place for money. They could also be very semantically close. For example, the verb *train* has 10 senses in WordNet, and the first two senses are defined as follows:

Sense 1:
  train, develop, prepare, make prepared, educate
    (definition: prepare for a future task or career
     example: The hospital trains the new doctors.)
Sense 2:
  train, prepare
    (definition: undergo training or instruction
     example : He is training to be a doctor.)

A semantic lexicon like WordNet makes important semantic distinctions; some of which may be more finely grained than needed for IR. For IR purposes, it would be better to relate the two senses rather than considering them as distinctive and losing such links. While for the case of *bank*, we do need to separate the two senses.

The above examples indicate that the sense distinctions predefined in a lexicon are not suitable for IR. [Kilgarriff1993] argued that word senses should be decided by the special task involved. Prior studies which have reported improvements [Schütze and Pedersen1995, Schütze1998] also abandoned the notion of predefined word senses but decided senses based on the corpus.

Our context distance model uses the same strategy. We abandon predefined word senses but represent senses using context vectors, based on the actual usage of the word in a document. We do not assume absolute sense distinctions but compute the relative distance, based on corpus information. Through these measures, we avoid the problems with traditional sense-based retrieval.

The method we proposed is somewhat similar to the work in [Schütze and Pedersen1995, Schütze1998] in that both use context vectors to represent word meanings and both use global corpus and local document information. The significant differences are that while their approach still assigns a *sense* to each word occurrence, we only compute the relative distances. In this sense, we do not assume absolute sense distinctions as they do. While they build context vectors for each occurrence of a word in the document, we compute a context vector for all the occurrences of a word in a document. Their clustering model is also very different from our way of computing context distances.

5 Conclusion and Future Work

We showed in this paper how we integrate context distance model with morphology for retrieval. Our proposed context distance model avoids the problems encountered in traditional sense-based retrieval and uses both global corpus and local document information. We tested the new technique on a sub-collection of TREC-4 corpora and the results showed a measurable improvement: 8.6% on 11-point average precision.

In the future, we plan to explore alternative ways to compute the distance between context vectors, better ways to use the context distance in determining word similarity in retrieval, and methods for improving the speed of the system. Additionally, we will study how the proposed context distance model can be used for other applications.

Acknowledgments

We are very grateful to Dr. Judith Klavans for her valuable comments on the draft of this paper.

References

Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. 1994. Automatic query expansion using smart: Trec-3. In *Proceedings of The third Text REtrieval Conference (TREC-3)*, Gaithersburg, Maryland.

CELEX. 1995. The CELEX lexical database—Dutch, English, German. CD-ROM. Centre for Lexical Information, Max Planck Institute for Psycholinguistics, Nijmegen.

Kenneth W. Church. 1995. One term or two? In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95)*, pages 310–318, Seattle.

Scott Deerwester, Susan T. Dumais, George Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

A. William Gale, Kenneth W. Church, and David Yarowsky. 1992. One sense per discourse. In *Proceedings of the 4th DARPA Speech and Natural Language Workshop.*

David A. Hull. 1996. Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society for Information Science*, 47(1):70–84.

Adam Kilgarriff. 1993. Dictionary word sense distinctions: An enquiry into their nature. In *Computers and the Humanities*, volume 26, pages 365–387.

Robert Krovetz and W. Bruce Croft. 1992. Lexical ambiguity and information retrieval. *ACM Transactions on Information Systems*, 10(2):115–141.

Robert Krovetz. 1993. Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'93)*, pages 191–203, Pittsburg, PA.

Huang Li and Naoki Abe. 1998. Word clustering and disambiguation based on co-occurrence data. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics(COLING-ACL'98)*, pages 749–755, Montreal, Canada.

Julie Beth Lovins. 1968. Development of a stemming algorithm. *Translation and Computational Linguistics*, 11(1):22–31.

Susan McRoy. 1992. Using multiple knowledge sources for word sense discrimination. *Computational Linguistics*, 18(1):1–30.

Hwee Tou Ng and Hian Beng Lee. 1996. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 40–47, Santa Cruz, CA, USA, June.

M. F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14:130–137.

Hinrich Schütze and Jan O. Pedersen. 1995. Information retrieval based on word senses. In *Symposium on Document Analysis and Information Retrieval (SDAIR)*, Las Vegas, NV. University of Nevada.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1), March.

M. Ellen Voorhees. 1993. Using WordNet to disambiguate words senses for text retrieval. In *Proceedings* *of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'93).*

Jinxi Xu and Bruce W. Croft. 1998. Corpus-based stemming using cooccurrence of word variants. *ACM Transactions on Information Systems*, 16(1):61–81.

David Yarowsky. 1992. Word sense disambiguation using statistical models of Roget's thesaurus categories trained on a large corpus. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING'92)*, Nantes, France.

David Yarowsky. 1993. One sense per collocation. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 266–271, Plainsboro, New Jersey, March. ARPA Software and Intelligent Systems Technology Office, Morgan Kaufmann, San Francisco, California.