Viewing Morphology as an Inference Process

Robert Krovetz
Department of Computer Science
University of Massachusetts, Amherst, MA 01003

Abstract

Morphology is the area of linguistics concerned with the internal structure of words. Information Retrieval has generally not paid much attention to word structure, other than to account for some of the variability in word forms via the use of stemmers. This paper will describe our experiments to determine the importance of morphology, and the effect that it has on performance. We will also describe the role of morphological analysis in word sense disambiguation, and in identifying lexical semantic relationships in a machine-readable dictionary. We will first provide a brief overview of morphological phenomena, and then describe the experiments themselves.

1 Introduction

Morphology is the area of linguistics concerned with the internal structure of words. It is usually broken down into two subclasses: *inflectional* and *derivational*. Inflectional morphology describes predictable changes a word undergoes as a result of syntax - the plural and possessive form for nouns, and the past tense and progressive form for verbs are the most common. These changes have no effect on a word's part-of-speech; a noun still remains a noun after pluralization. In contrast, derivational morphology may or may not affect a word's part-of-speech, and may or may not affect its meaning (e.g., '-ize', '-ship').

The different forms of a word can have a strong impact on the effectiveness of a retrieval system. English has relatively weak morphology and does not suffer from these problems as much as other languages (e.g., Hungarian or Hebrew, which may have thousands of variants on any given word; experiments with a Hebrew retrieval system have shown that a failure to process morphological variants resulted in retrieving only 2%-10% of the documents retrieved with such processing [Choueka 92]).[1] However, even in English, retrieval based on conflating word forms leads to significant improvements in performance. Such conflation, which is the norm in IR systems, is referred to as 'stemming'.

Stemming can be viewed from several different perspectives. It can be thought of as a mechanism for query expansion - as a way of enhancing the query with terms that are not the literal word-forms given by the user; from this view it is similar to using a thesaurus (cf. [Stevens 70]). From another perspective it can be viewed as clustering, in which the clusters are based on the rules for conflation. From a third perspective it is a way of normalizing the concepts used in the query. The concepts are the senses of the query words, and the rules for deciding which word forms are related can be considered as inferences. This view will be elaborated in the section on word-sense disambiguation.

Various stemming algorithms have been discussed in the literature. They range from simply removing plural endings (and also perhaps other inflectional forms such as the past participle '-ed' and the gerund or present participle '-ing'), to approaches that handle a variety of suffixes. The two most common stemmers are the Lovins stemmer ([Lovins 68]) and the Porter stemmer ([Porter 80]). The Lovins stemmer removes over 260 different suffixes using a longest-match algorithm, and the Porter stemmer removes about 60 suffixes in a multi-step approach; each step successively removes suffixes or makes some transformation of the stem (e.g., *-y* to *-i*).

One of the problems with stemming is that it does not pay attention to the differences caused by a word's meaning. For example, the word 'gravitation' is related to the force-of-gravity sense of the word 'gravity' rather than the sense meaning 'serious'. If this word

[1] English and Hebrew form two extremes; English typically has only three or four variants per word. Experiments with Slovene support the argument that stemming is very important for highly inflected languages [Popovic and Willet 92].

were simply stemmed we might conflate it with senses of 'gravity' that do not have the appropriate meaning.

Recognizing the ties between morphological variants is also an important component of an algorithm for word-sense disambiguation. Our hypothesis is that retrieval performance can be improved by indexing documents not by words, but by word *meanings*. Our previous experiments have shown that word senses are strongly correlated with judgments of relevance [Krovetz 92a]; when the sense of a query word does not match the sense of the word in a document, the document is unlikely to be relevant with respect to that word.

Most approaches to stemming not only ignore word meanings, they usually operate in the absence of any lexicon at all. The Lovins stemmer removes the suffix which is the longest-match, and the Porter algorithm iteratively removes suffixes from a pre-defined set. The aim is not to produce a linguistic root, but to improve performance. However, the absence of a lexicon results in improper conflations (and an associated loss in precision). There are also errors of omission because the rules they use do not provide for the full range of morphological variation. A representative sample of these errors for the Porter stemmer is given in Table 1.

Note that an IR system would normally not index words like 'doing' (because they are part of a stop-word list - a list of words that are not considered important for retrieval). The reduction of 'doing' to 'doe' interferes with this process, and so it would be indexed.

Other problems with the Porter stemmer involve the difference between a root and a stem. *Iteration* is turned into *iter*, and *general* is turned into *gener*. This not only results in errors of commission, it makes it difficult and often impossible to relate it to the entry in the dictionary. This is essential for our research on word-sense disambiguation; without knowing what word we are working with, disambiguation is impossible. In addition, it creates difficulties in interacting with the user. If the system were to provide the user with a list of terms for query expansion, the user would not always know which concepts they referred to.

2 Collection Statistics

In order to ensure generality, we conducted experiments with four test collections. Each collection covered a different domain (computer science, newspaper stories, physics, and law), and they represent a wide range in terms of average document length and overall number of documents. The statistics for the collections are given in Table 2; each collection corresponds to the domain just mentioned, respectively.

The statistics indicate that the WEST documents are much longer than those in the other collections. There are approximately the same number of documents as in the NPL collection, and about the same number of relevant documents per query. In contrast, the TIME collection contains a small number of medium length documents, and a small number of relevant documents per query, but the queries are about the same length as for WEST. Although the CACM collection contains many more documents than TIME, it contains about the same amount of text because each document is about nine times larger in TIME.

3 Revised Stemmer

Because of the difficulties we encountered with the Porter stemmer, we modified the algorithm to use a machine-readable dictionary.[2] The Porter algorithm is a five stage process in which different classes of inflectional and derivational suffixes are removed at each step. We modified the algorithm to check the word against the dictionary prior to each step. For example, given the word *generalizations*, the modified algorithm would first check to see if it was in the dictionary. Since it is not, it would remove the 's' in the initial step. It would then look up *generalization* at the next step and return that as the result (since generalization *is* in the dictionary).

The modified algorithm sometimes performed worse than the original (see Tables 4 and 5; the modified algorithm is indicated under the heading 'rev-stem'). We found that this was due (at least in part) to words ending in 'e'. If the original stemmer was given the word *distribute*, it would remove the 'e' during one of the stages and return *distribut* as the stem. This would allow the word to be conflated with *distributing* and *distributed*. With the modified algorithm, *distribute* would be found in the dictionary prior to the first stage, and would be returned as the stem. Given the word *distributed*, however, it would not find that form in the dictionary, and the 'ed' would be removed. Thus, *distributed* and *distributing* were no longer being conflated with *distribute*. An effort was made to fix this problem, but similar difficulties occured at other points in the algorithm, and with other endings, and we felt it was better to develop a new algorithm entirely. The following sections will describe the development of that algorithm, and how it performs compared to the existing stemmer. We will then describe some of the relationships between morphology and semantics, and how stemming can be used for word sense disambiguation. Finally, we will describe our experiments to identify related senses in the dictionary; this will be used to test our hypothesis that unrelated senses will be more effective than related senses at separating relevant from nonrelevant documents.

[2] The dictionary we are using in our research is the *Longman Dictionary of Contemporary English (LDOCE)*.

Errors of Commission	**Errors of Omission**
organization/organ	european/europe
doing/doe	analysis/analyzes
generalization/generic	cylinder/cylindrical
numerical/numerous	matrices/matrix
policy/police	urgency/urgent
university/universe	create/creation
easy/easily	decompose/decomposition
addition/additive	machine/machinery
negligible/negligent	useful/usefully
execute/executive	noise/noisy
define/definite	route/routed
past/paste	search/searcher
ignore/ignorant	sparse/sparsity
special/specialized	explain/explanation
arm/army	resolve/resolution
head/heading	triangle/triangular

Table 1: Errors made by the Porter Stemmer

	CACM	**TIME**	**NPL**	**WEST**
Number of queries	64	83	93	34
Number of documents	3204	423	11429	11953
Mean words per query	13.0	8.9	7.1	9.6
Mean words per document	62	581	43	3262
Mean relevant documents per query	15.84	3.90	22.3	28.9
Number of words in collection	200,000	250,000	490,000	39,000,000

Table 2: Statistics on information retrieval test collections

Ending type	Instances (CACM)	Instances (NPL)	Instances (TIME)	Instances (WEST)
Plural	178 (44%)	345 (58%)	286 (54%)	373 (57%)
Tense	109 (27%)	128 (21%)	123 (23%)	145 (22%)
Aspect ('ing')	115 (29%)	125 (21%)	122 (23%)	136 (21%)
Total	402 (100%)	598 (100%)	531 (100%)	654 (100%)

Table 3: Statistics on Inflectional Endings

4 An Inflectional Stemmer

The first step in the development of the new algorithm was inflectional morphology. These variations always occur after derivational forms, and a small number of endings account for most of the occurrences of different word forms. We were also curious how much they accounted for the improvement in retrieval performance.

Table 3 shows the frequencies of the inflectional endings in the different collections. They are based on the endings for words used in the queries, and for the additional words that are conflated by the stemmer; the statistics for the words in the entire collection vocabulary are similar. They indicate that plural inflections account for 44-58% of the inflectional endings, and that past tense and aspect ('ing') are evenly divided.

Because of the problems we encountered with the Porter algorithm, we wanted to be very careful with words that end in 'es' or 'ed'; should we remove two letters, or just one? The Porter algorithm bases this decision on the number of vowel/consonant sequences in the resulting stem. Rather than use such a measure, we noticed that if it was possible for a word to end in 'e', that was usually the correct root. As with the modified Porter stemmer, we check the dictionary prior to stemming any word, and if it is found then that is returned as the result. If it is not found, we replace 'ing'/'es'/'ed' with 'e', and check the the dictionary again. If this fails the entire suffix is removed and the dictionary is checked once more.

There are, however, words that are exceptions. If we always prefer a stem that ends in 'e', then *attached* would be stemmed to *attache*. Part-of-speech can help with this, but it will not handle all of the cases. We wrote a routine to identify all of the words in the dictionary that ended in 'e', and that were also in the dictionary if that letter was removed. This resulted in a list of 229 words, and the words were manually examined to determine the exceptions (60 words). The exception list is only examined for the 'ed' and 'ing' endings; most of the words are nouns, and if the word-form is a plural, then the "exception" is in fact the correct root. For example, *suited* is reduced to *suit* because the word is on the exception list, but *suites* is reduced to *suite*.

The algorithm for the inflectional stemmer has three parts: converting plurals to singular form, converting past tense to present tense, and removing '-ing'. Plurals are subdivided into three cases - endings in 'ies', 'es', or 's'. In the first case the 's' is removed and the stem is checked against the dictionary; if it is found then it is accepted as the root, and if not the ending is replaced by 'y'. This prevents *calories* from being converted to *calory*. In the second case we first try removing 's', and if the stem is found in the dictionary it is accepted as the root; otherwise the 'es' is removed, and the dictionary is checked again; if this fails the default is to assume that the root ends in 'e'. Finally, if the word ends in 's', and the ending is not 'ous' or a double 's', the 's' is removed. The routines for past tense and 'ing' endings are similar to those for the plural, except for the need to consider the exception for words ending in 'e', and the need to convert a doubled letter to a singular one (e.g., *controlling*); this conversion is always done if the stem including the doubled letter is not found in the dictionary.

Tables 4 and 5 give a comparison of the Porter stemmer, the Revised Porter stemmer, and the inflectional stemmer. The baseline for comparison is no stemming at all. We find that there is an improvement in performance in all of the collections, but that there are substantial differences between them. The improvement is minor in the TIME collection, greater in WEST and CACM, and extremely high in NPL. The relative lack of improvement in TIME can be partially explained by an examination of the query vocabulary; many of the query words refer to locations, and the Porter stemmer does not conflate them with their variants (e.g., *Europe* and *European*). The improvement in the other collections is a reflection of the average document length; stemming becomes increasingly important as the documents become shorter. This is just what we would expect - the probability of matching the exact word-form used in the query decreases if the documents are shorter. The results with the WEST collection indicate that morphology is still important even in documents that are long.

Two of the collections (CACM and WEST) include a modified set of queries that contain phrase and proximity operators.[3] The use of these operators provides an increase in performance, but the degree of improvement varies depending on the collection and on the interaction with stemming. The correct formulation of phrase-based queries is still unclear, and we are in the process of conducting further experiments with other formulations and with queries for the other two collections.

Finally we note that the improvements from stemming increase as we go to higher levels of Recall, and that derivational morphology is responsible for a greater percentage of the improvement at high levels of Precision. That is, although inflectional morphology is important at all levels of Recall, derivational morphology accounts for the greatest difference in the documents that will actually be seen by the user.

[3]The proximity operator specifies that the query words must be adjacent and in order, or occur within a specific number of words of each other; the phrase operator is a generalization of the proximity operator so that a partial match of the phrase is acceptable.

CACM

	Precision (% change) – 50 queries								
Recall	no-stem	std-stem		rev-stem		inflect-stem		deriv-stem	
25	49.0	54.1	(+10.4)	55.3	(+12.7)	52.7	(+7.5)	55.5	(+13.1)
50	32.7	36.7	(+12.2)	37.2	(+13.7)	36.8	(+12.5)	37.8	(+15.7)
75	15.4	19.4	(+26.0)	20.9	(+35.0)	19.8	(+27.9)	20.4	(+32.4)
average	32.4	36.8	(+13.5)	37.8	(+16.6)	36.4	(+12.4)	37.9	(+17.1)

NPL

	Precision (% change) – 93 queries								
Recall	no-stem	std-stem		rev-stem		inflect-stem		deriv-stem	
25	24.3	35.3	(+45.4)	34.2	(+41.1)	32.9	(+35.7)	33.1	(+36.2)
50	13.8	20.3	(+47.9)	18.1	(+31.6)	18.0	(+31.2)	18.4	(+33.9)
75	6.5	9.1	(+39.9)	8.0	(+23.9)	8.2	(+26.8)	8.2	(+26.6)
average	14.8	21.6	(+45.3)	20.1	(+35.6)	19.7	(+33.0)	19.9	(+34.1)

TIME

	Precision (% change) – 83 queries								
Recall	no-stem	std-stem		rev-stem		inflect-stem		deriv-stem	
25	71.0	73.0	(+2.8)	71.7	(+1.0)	71.5	(+0.7)	73.9	(+4.1)
50	66.6	67.8	(+1.8)	68.3	(+2.6)	68.4	(+2.7)	70.2	(+5.5)
75	57.7	57.0	(−1.3)	58.8	(+1.8)	59.0	(+2.2)	60.1	(+4.1)
average	65.1	65.9	(+1.3)	66.3	(+1.8)	66.3	(+1.8)	68.1	(+4.6)

WEST

	Precision (% change) – 34 queries								
Recall	no-stem	std-stem		rev-stem		inflect-stem		deriv-stem	
25	71.9	71.6	(−0.5)	72.7	(+1.0)	71.0	(−1.4)	72.8	(+1.2)
50	51.3	53.7	(+4.7)	53.4	(+4.1)	53.6	(+4.5)	53.0	(+3.4)
75	26.6	30.4	(+14.2)	27.8	(+4.7)	30.6	(+15.2)	30.5	(+14.9)
average	49.9	51.9	(+3.9)	51.3	(+2.7)	51.7	(+3.6)	52.1	(+4.4)

Table 4: Comparative Performance of Different Stemmers - Word Based

CACM

	Precision (% change) – 50 queries								
Recall	no-stem	std-stem		rev-stem		inflect-stem		deriv-stem	
25	53.2	55.7	(+4.6)	53.6	(+0.8)	54.1	(+1.6)	55.8	(+4.8)
50	34.3	37.0	(+7.6)	37.3	(+8.6)	37.9	(+10.3)	38.8	(+12.9)
75	17.6	20.8	(+18.2)	20.1	(+14.6)	19.6	(+11.9)	20.9	(+18.7)
average	35.0	37.8	(+7.9)	37.0	(+5.6)	37.2	(+6.2)	38.5	(+9.8)

WEST

	Precision (% change) – 34 queries								
Recall	no-stem	std-stem		rev-stem		inflect-stem		deriv-stem	
25	72.8	76.0	(+4.4)	76.4	(+5.0)	74.4	(+2.2)	76.0	(+4.5)
50	53.5	60.3	(+12.6)	58.6	(+9.5)	58.7	(+9.7)	58.9	(+10.1)
75	32.7	38.0	(+16.1)	34.1	(+4.2)	34.9	(+6.8)	35.1	(+7.2)
average	53.0	58.1	(+9.6)	56.4	(+6.4)	56.0	(+5.7)	56.7	(+6.9)

Table 5: Comparative Performance of Different Stemmers - Phrase Based

Ending	**Count**	**Other Endings**
-ate	66	(-ation 67) (-ative 8) (-ator 31)
-er	54	
-ion	41 [108]	
-ly	38 [64]	
-ity	34 [44]	
-ic	33	(-ical 14) (-ics 5)
-ize	31	(-ise 4)
-al	29 [43]	(-ally 26)
-ive	19 [27]	(-iveness 1)
-able	18	(-ability 8) (-ible 3) (- ibility 2)
-or	14 [45]	
-ent	12	(-ence 8) (-ency 4) (-ant 7) (-ance 14)
-ary	11	
-ment	10	
-ify	10	
-ure	9	
-ism	8	
-th	7	
-ine	6	
-ar	6	
-ous	5	
-ess	5	(-ness 3)

Other endings: 4 times or less
Numbers in brackets indicate the total count, which includes the endings covered by larger groups (e.g., ly = 38 + 26 (ally)).

Table 6: Frequencies for Derivational Endings in the NPL Collection

CACM

	Number of Terms	Words found in LDOCE
Porter Stemmer	5318	2027 (38 %)
Inflectional Stemmer	6848	4266 (62 %)
Derivational Stemmer	6301	4367 (69 %)
Unstemmed	8978	3810 (42%)

NPL

	Number of Terms	Words found in LDOCE
Porter Stemmer	7688	2365 (31 %)
Inflectional Stemmer	9420	4677 (50 %)
Derivational Stemmer	8753	4800 (55 %)
Unstemmed	11921	4159 (35 %)

TIME

	Number of Terms	Words found in LDOCE
Porter Stemmer	13387	5752 (43 %)
Inflectional Stemmer	16016	10641 (66 %)
Derivational Stemmer	15262	10846 (71 %)
Unstemmed	20560	9171 (45 %)

WEST

	Number of Terms	Words found in LDOCE
Porter Stemmer	104911	9769 (9 %)
Inflectional Stemmer	118592	19208 (16 %)
Derivational Stemmer	112135	19474 (17 %)
Unstemmed	137401	18329 (13 %)

Table 7: Proportion of Collection Vocabulary in the Longman Dictionary of Contemporary English (LDOCE)

	CACM	NPL	TIME	WEST
Algorithm	3	4	2	1
Spelling Errors	1	1	0	92
Irregular Form	2	2	1	0
Dictionary Omission	2	8	11	2

Table 8: Reasons for Failure to Produce a Root

	Words	Senses
Derivational Variants	7632	7927
Spelling Variants	87	87
Clipped Forms/Abbreviations	89	94
Compounds	1435	1446
Unrelated Words	1347	1410

Table 9: Analysis of Trigram Groups

5 A Derivational Stemmer

Inflectional variation is typically associated with syntax, and has relatively little impact on a word's meaning (see the discussion on word-sense disambiguation in the following section for exceptions to this). There is also psychological evidence that inflectional variants are processed differently by the brain than derivational variants; derivational variants appear to be stored as a whole in the mental lexicon, but inflectional variants are processed dynamically ([Aichison 88], [Badecker and Caramazza 89]). We took a conservative approach to derivational variants and only stem them if we could show that they were related to the meaning of the root form. Dictionaries will usually list a word form separately if it has a meaning that is distinct from the root, so we did not stem a word-form if it appeared in the dictionary.[4] This was only an initial approximation, and was intended to reduce the set of word forms to those that were either unrelated, or in which the relationship depended on the word's meaning.

Determining whether word-senses are related is a major problem both practically and theoretically. As we mentioned in the Introduction, our hypothesis is that retrieving documents on the basis of word senses (instead of words) will result in better performance. Our approach is to treat the information associated with dictionary senses (part of speech, subcategorization, subject area codes, etc.) as multiple sources of evidence (cf. [Krovetz 91]). This process is fundamentally a divisive one, and each of the sources of evidence has exceptions (i.e., instances in which senses are *related* in spite of being separated by part of speech, subcategorization, or morphology). Identifying related senses will help us to test the hypothesis that unrelated senses will be more effective at separating relevant from nonrelevant documents than senses which are related.

We developed the new algorithm by conducting quantitative studies of the frequency of various endings. We first extracted a list of 106 derivational endings from the Longman dictionary,[5] and then printed the term dictionary[6] for an unstemmed version of each collection. This gave us a list of the unique words along with their frequency of occurrence. This list was processed to identify the words that ended with each suffix, and the number of times each suffix occurred. We also produced a subset of this list consisting of the words used in the queries for the collection along with any words that would have been conflated using the existing stemmer.

Table 6 gives a listing of the frequencies of the various endings for one of the collections; we only provide the data for the words used in the queries (or conflated by the stemmer), but the statistics for the entire collection are similar. The rank of each ending varies somewhat between the different collections, but there is a consensus about which endings are the most common. The ten most frequent endings were used as the basis for modifying the inflectional stemmer. The table also indicates the most common combinations of endings.

The endings identified as most frequent were: -er, -or, -ion, -ly, -ity, -al, -ive, -ize, -ment, and -ble (this incorporated -ible and -able). We modified the inflectional stemmer by writing a separate procedure to handle each ending, and then determined the number of words that were being conflated by it and the words that were not being conflated because they were in the dictionary. We also identified the words that *should* have been conflated, but were not; this was due to spelling errors in the collection and omissions in the Longman dictionary.

As a result of the initial modifications and analysis, five more endings were added (-ism, -ic, -ness, -ncy, and -nce). A few more words might have been conflated by handling additional endings, but the variants for those endings were usually found in the dictionary and would not be conflated with the root even if the ending was recognized.

We tried to be conservative in the conflations by checking the dictionary to see that it contained the proposed root. However, we recognize that any dictionary will be incomplete, and allowed certain transformations to be made even if the resulting form was not found in the dictionary. For example, -ization was always reduced to -ize, and -ally was always reduced to -al; the -ism and -ness endings were always removed even if the root was not found in the dictionary. A small number of these allowances eliminated almost all of the errors caused by dictionary omissions.

We had several aims in creating a new stemmer. First, we wanted the stems to be words rather than truncated word-forms. Second, we did not want to conflate any word-forms unless their meanings were related. Third, we wanted the coverage to be as broad as possible (i.e, to conflate as many word-forms as we could, subject to the constraint that their meanings were related). Fourth, we wanted the performance of the stemmer to be at least as good as the Porter algorithm. Finally, we wanted the stemmer to play a role as one component of an algorithm for word-sense disambiguation.

[4] This was also true of the inflectional stemmer, but most of the morphological variants listed in the dictionary are derivational, and identifying the links between the word-senses becomes a more significant problem.

[5] The Longman dictionary provides definitions for prefixes, suffixes, and combining forms. The type of affix is explicitly indicated. We also note that the number of suffixes for English stands in marked contrast to a highly inflected language; [Popovic and Willet 92] describes a stemmer for Slovene that uses 5276 suffixes.

[6] The term dictionary is a data structure used by the retrieval system; it contains the vocabulary for the collection, and should not be confused with the general-purpose dictionary we are using for our experiments. All references to 'dictionary' refer to the general-purpose dictionary.

The first objective was largely met; Table 7 gives a breakdown of the vocabulary for the various collections after being processed by the different stemmers. It indicates that the percentage of the collection vocabulary which is contained in the Longman dictionary is *much* higher after being processed with the inflectional or derivational stemmers, and is approximately double the number of words produced by the Porter stemmer. The percentage for the WEST collection might seem low relative to the other collections, but it is a reflection of the collection's size. The number of dictionary words in the WEST collection is actually very large; there are only 27855 non-phrasal headwords in the Longman dictionary, so the WEST collection actually contains 70% of the words defined in the entire dictionary.

The second objective was also largely met, but complete success was not possible due to the fact that the dictionary is incomplete, and because the collection contains spelling errors. For example, the dictionary was missing *digitize*, and *factorial*, and these were stemmed to *digit* and *factory*. These are plausible roots given the absence of those words, and they were also words that would be conflated by the Porter stemmer. False conflations were also made because of proper nouns (e.g., *Mooer* was stemmed to *moo*, and *Navier-Stokes* was stemmed to *navy* and *stoke*).[7] Proper nouns must be processed (consider *Pakistani* or *Algerian*), but ideally the stemmer should be tailored to treat proper nouns differently. Finally, some false conflations were due to spelling errors. For example, *properity* (prosperity) was stemmed to *proper*, and *agence* (agency) to *age*.

Spelling errors also caused a problem in meeting the third objective. A breakdown of the different reasons for errors is given in Table 8. These errors were identified by grouping the words in the vocabulary file according to initial trigrams (the first three letters of each word); words that are morphological variants will almost always start with the same trigram. Each group was put on a separate line, and lines with only one word were deleted (these were words that had no morphological variants). The remaining lines were edited by hand so that they contained only a root form and its variants (this only involved examining a few hundred lines per file). The resulting file was then processed to identify the words that were not found in the dictionary, and the reasons for conflation failure were identified. The errors due to the algorithm were the result of only processing the most frequent 15 suffixes, and because of a number of special cases that were missed. The words missing from the dictionary were usually technical words (e.g., *superconduct, exosphere, simultaneity*), or proper nouns (e.g., *Algerian, Algeria*; these were the sole cause of this error for the TIME collection). Finally, we found there were words that were not being conflated due to spelling errors and typos. Because the queries and documents are fairly long, the failure to conflate spelling errors with the root is unlikely to have an impact on performance.

[7] Our retrieval system breaks up hyphenated forms into separate words.

The analysis of coverage failures only gives us an approximation of the failures. It is based on the words used in the queries, and the words that would be conflated with them by the Porter stemmer. We wanted to ensure that our coverage was at least as good as the Porter algorithm, but we also wanted to surpass it. The Porter algorithm does not conflate *Europe* and *European*, and these failures were only identified because both word forms occurred in the queries. To capture these cases, we once again made use of trigrams, but this time the trigrams were based on the definitions in the Longman dictionary. Each definition was processed to identify any words that have an initial trigram in common with the word being defined (e.g. the definition for *cylindrical* contains the word *cylinder*). The definitions were then manually reviewed to eliminate false positives, and the resulting data is used to give a direct transformation from the variant to the root form. This is also the way we handle irregular variants (e.g., *formula* and *formulae*; irregular variants are explicitly mentioned in one of the fields in the dictionary). We are still in the process of analyzing the dictionary, and at the moment the conflations are based on a manual examination of those query words that occur in the data. Most of the false positives have already been removed, and the initial breakdown is given in Table 9.

The data from the trigram processing is also the basis for two more modifications to the stemmer. First, the dictionary contains a number of entries in which the word being defined is directly related to the root. For example, the definition for *cylindrical* is: 'of, related to, or having the form of a cylinder'. The existence of the definition would normally prevent *cylindrical* and *cylinder* from being conflated (based on the assumption that the variant has a different meaning than the root). However, if all of the senses of the variant contain a reference to the root, then we will assume they can be conflated without harm. Second, the association between a variant and its root is based on specific senses, and this allows us to use them as part of a word-sense algorithm. We will discuss this further in the following section.

Our assumption was that if a variant occurs in the dictionary, it has a different meaning from the root and should not be conflated with it. We found that almost two-thirds of the derivational variants occur in the dictionary, and this made a significant difference between the performance of the Porter stemmer and the derivational stemmer. We analyzed these variants, and found that about 40% of them can be conflated, about 20% are unrelated, and about 40% are context-sensitive (i.e., the conflation depends on the meaning of the query word). These judgments are based on the presence of the root form in the definition of a variant, or on the degree of overlap between the words in their definitions.

The current performance of the algorithm is given in Tables 4 and 5 (under the column labeled 'deriv'). It includes conflations for proper nouns (Algerian/Algeria), and some conflations based on the variants in the dictionary. With the exception of the NPL collection and the phrasal queries for the WEST collection, the performance exceeds that of the Porter stemmer.

6 The Role of Morphology in Word-Sense Disambiguation

One of the primary motivations for developing a new stemmer is that it plays an important role in resolving lexical ambiguity. The Porter stemmer made such resolution impossible because it involved a shift of representation; we were no longer dealing with words, but with stems. Beyond just a relationship between words, morphological variants are really relationships between word *senses*. There are a number of situations in which the stemmer can exploit these relationships to resolve a word's meaning. For example:

1. Irregular morphology - *antennae* is only a plural of the type of antenna that is associated with an insect, not with a television antenna. Similarly, *media* is the plural of *medium* used in the sense of entertainment, not in the sense of a spiritualist.

2. Part-of-Speech - *intimation* is derived from *intimate* (v), and *intimately* is derived from *intimate* (adj). Because suffixes only attach to roots with particular parts-of-speech, this information can be used to discriminate one homograph from another.

3. Run-Ons - These are typically suffixes that are associated with the end of a dictionary entry, and indicate that the morphological variant may be formed for that word. They have a predictable relationship to the root form, and are primarily a way for the lexicographer to include additional entries without taking up much space (which is a major concern with printed dictionaries). Comparing run-on entries with main entries can make us focus on the data associated with the main entry that would allow us to discriminate it from the run-on (e.g., *boxer* as a type of dog vs. a human).

The trigram groupings also provide data about the associations between morphological variants and particular senses. For example, *save* with reference to some resource (time, money) can be nominalized as *saver*, but *save* with reference to rescue or preservation from danger can be nominalized to *saviour*. These nominalizations are mentioned in separate senses, and the discrimination can be built into a lookup table (i.e., if we see *saver* in the document, the retrieval system can record which sense of *save* it is related to).

As we mentioned in the introduction, we view morphology as an inference process because it involves normalizing concepts (deciding which words refer to the same concept/sense despite a variation in form). We can also get related senses that differ in part of speech, but without having an explicit affix; this is referred to as zero-affix morphology or functional shift. The Longman dictionary explicitly indicates some of these relationships by homographs that have more than one part of speech. It usually provides an indication of the relationship by a leading parenthesized expression. For example, the word *bay* is defined as N,ADJ, and the definition reads '(a horse whose color is) reddish-brown'. However, out of the 41122 homographs defined, there are only 695 that have more than one part of speech. Another way in which LDOCE provides these links is by an explicit sense reference for a word outside the controlled vocabulary; the definition of *anchor* (v) reads: 'to lower an **anchor**^{1} (1) to keep (a ship) from moving'. This indicates a reference to sense 1 of the first homograph.

Zero-affix morphology is also present implicitly, and we conducted an experiment to try to identify instances of it using a probabilistic tagger [Church 88]. The hypothesis is that if the word that is being defined (the definiendum) occurs within the text of its own definition, but occurs with a different part of speech, then it will be an instance of zero-affix morphology. The question is: How do we tell whether or not we have an instance of zero-affix morphology when there is no explicit indication of a suffix? Part of the answer is to rely on subjective judgment, but we can also support these judgments by making an analogy with derivational morphology. For example, the word *wad* is defined as 'to make a **wad** of'. That is, the noun bears the semantic relation of *formation* to the verb that defines it. This is similar to the effect that the morpheme *-ize* has on the noun *union* in order to make the verb *unionize* (cf. [Marchand 63]).

The result of the experiment is that the dictionary contains at least 2636 senses in which the headword was mentioned, but with a different part-of-speech, and all of these senses were related. The instances in which the senses were *not* related were entirely due to errors caused by the tagger. The main causes of error were idiomatic senses, unexpected punctuation, and word types that are infrequent (and thus do not provide enough data about how often they occur with particular parts-of-speech).

We also conducted an experiment to identify related senses through word overlap. As an example, consider the definitions for the words *appreciate* and *appreciation*:

- **appreciate**
  1. to be thankful or grateful for
  2. to understand and enjoy the good qualities of
  3. to understand fully
  4. to understand the high worth of
  5. (of property, possessions, etc.) to increase in value
- **appreciation**
  1. judgment, as of the quality, worth, or facts of something
  2. a written account of the worth of something
  3. understanding of the qualities or worth of something
  4. grateful feelings
  5. rise in value, esp. of land or possessions

The word overlap approach pairs up sense 1 with sense 4 (grateful), sense 2 with sense 3 (understand; qualities), sense 3 with sense 3 (understand), sense 4 with sense 3 (understand; worth), and sense 5 with sense 5 (value; possessions). The matcher we are using ignores closed class words, and makes use of a simple morphological analyzer (for inflectional morphology). It ignores words found in example sentences (preliminary experiments indicated that this did not help and sometimes made matches worse), and it also ignores typographical codes and usage labels (formal/informal, poetic, literary, etc.). It also does not try to make matches between word senses that are idiomatic (these are identified by font codes).

This experiment is similar to one conducted by Lesk (cf. [Lesk 86]), but our experiment is focused on identifying links between morphological variants, and Lesk's experiments were designed to resolve lexical ambiguity directly; that approach only had a success rate of approximately 40%. We found that the success rate between morphological variants was 98% if there were two or more words in common, and 65% if there was only one word in common. Many of the false positives were caused by very general words (e.g., *thing*, *use*, and *make*). Once these words are eliminated the success rate is over 80%.

7 Future Work

This paper has focused on inflectional and derivational variants. Morphology is also important in dealing with phenomena such as acronyms, abbreviations, hyphenation, numbers and proper nouns. These are usually not discussed in the linguistics literature, but are important in a system that operates on real-world text. Especially important are the problems of open and closed compounds ('on-line'/'on line'/'online'). Retrieval systems will often split up hyphenated words and index them separately. This can cause failures due to matching on only one part of the compound, and if the closed form is used in either the query or document it will fail to match the open form.

We are in the process of determining the effect of retaining hyphenated forms, as well as splitting them up, but including them within the context of a proximity operator (i.e., ensuring that they are considered a unit with respect to retrieval). We are also trying to identify closed/open compounds by concatenating every adjacent word in the collection and comparing that against the term dictionary. This has resulted in a list of approximately 600 compounds for each collection, and they will be reviewed by hand to eliminate false positives. All open compounds in the queries will be modified to have proximity operators, and the closed compounds will be added to the query.

With respect to stemming, more work is needed on morphological relationships that are context-sensitive. We will determine the effect of adding morphological variants to the query based on the meaning of the query words.

We will also be examining the effect of variant spellings. The data from the trigrams provided a number of instances of spelling variants (e.g., *judgement* vs. *judgment*), and there are also many variants that are explicitly mentioned in the dictionary. We found several instances in which spelling variation can have an impact on performance. For example, the queries in the NPL collection use the British spelling of some words (*transistorised*), and the Porter stemmer would not conflate this with *transistor*. Surprisingly, although the queries use the British spelling, the American spelling (*transistorized*) is used in the documents themselves.

We performed an experiment involving manual filtering of word-senses. This filtering improved the performance on one collection (WEST), but decreased the performance on another (CACM). We are in the process of determining why there was a decrease, but it provides an interesting supplement to Harman's work, in which user filtering was only simulated ([Harman 88]).

Finally, we will use the data on zero-affix morphology to conduct an experiment involving word-sense disambiguation with part-of-speech. The words in the queries and documents with be tagged using a stochastic tagger ([Church 88]). Any query words that have a tag that is different from the document word will be treated as a different term; the words that are instances of zero-affix morphology will be treated as matching despite the different tag.

8 Conclusion

Comparisons of stemmed and unstemmed collections have usually indicated relatively little improvements in performance, and a query-level analysis has indicated that stemming harms almost as many queries as it helps [Harman 91]. We have shown that stemming *does* result in significant improvements in performance, and this improvement is most significant when the documents are fairly short. We have also given a breakdown of the relative effectiveness of inflectional vs. derivational morphology, and shown that stemming to lexemes provides consistent improvement over a nonconflated collection, and generally exceeds the performance of a standard stemmer. Morphology is not simply a relation between words, but between word *senses*, and is therefore an important component of an algorithm for word-sense disambiguation. We have described some of these relationships and noted how a stemmer can be modified to use them in resolving ambiguity. Finally, we have described two simple techniques for identifying senses that are related, and shown these techniques to be highly effective. These relationships will be used to test the hypothesis that unrelated senses are an effective means for separating relevant from non-relevant documents.

Acknowledgements

This research was supported by the NSF Center for Intelligent Information Retrieval at the University of Massachusetts.

References

[Aichison 88] Aichison J, "Reproductive Furniture and Extinguished Professors", in *Language Topics. An International Collection of Papers by Colleagues, Students, and Admirers of Professor Michael Halliday*, R. Steele and T. Threadgold (eds), Amsterdam, Philadelphia, Vol. II, pp. 3-14, 1988.

[Badecker and Caramazza 89] Badecker W and Caramazza A, "A Lexical Distinction Between Inflection and Derivation", *Linguistic Inquiry*, Vol. 20(1), pp. 108-116, Winter, 1989.

[Choueka 92] Choueka Y, "Responsa: An Operational Full-Text Retrieval System with Linguistic Components for Large Corpora", in *Computational Lexicology and Lexicography: a Volume in Honor of B. Quemada*, A. Zampoli (ed), Giardini Press, Pisa, 1992.

[Church 88] Church K., "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text", in *Proceedings of the 2nd Conference on Applied Natural Language Processing*, pp. 136-143, 1988.

[Harman 88] Harman D, "Toward Interactive Query Expansion", *Proceedings of the Eleventh International Conference on Research and Development in Information Retrieval*, pp. 321-331, 1988.

[Harman 91] Harman D, "How Effective is Suffixing?", *Journal of the American Society for Information Science*, Vol. 42(1), pp. 7-15, 1991.

[Krovetz 91] Krovetz R, "Lexical Acquisition and Information Retrieval", in *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, Uri Zernik (ed), Lawrence Erlbaum Pub., pp. 45-64, 1991.

[Krovetz 92a] Krovetz R and Croft W B, "Lexical Ambiguity and Information Retrieval", *ACM Transactions on Information Systems*, Vol. 10(2), pp. 115-141, 1992.

[Krovetz 92b] Krovetz R, "Sense-Linking in a Machine Readable Dictionary", *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. 330-332, 1992.

[Lesk 86] Lesk M., "Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to tell a Pine Cone from an Ice Cream Cone", *Proceedings of SIGDOC*, pp. 24–26, 1986.

[Lovins 68] Lovins J, "Development of a Stemming Algorithm", *Mechanical Translation and Computational Linguistics*, Vol. 11, pp. 22-31, March 1968.

[Marchand 63] Marchand H, "On a Question of Contrary Analysis with Derivational Connected but Morphologically Uncharacterized Words", *English Studies*, 44, pp. 176-187, 1963

[Popovic and Willet 92] Popovic M and Willet P, "The Effectiveness of Stemming for Natural Language Access to Slovene Textual Data", *Journal of the American Society for Information Science*, Vol. 43, No. 5, pp. 384-390, 1992.

[Porter 80] Porter M, "An Algorithm for Suffix Stripping", *Program*, Vol. 14(3), pp. 130-137, 1980.

[Stevens 70] Stevens R, *Automatic Indexing: A State-of-the-Art Report*, NDI Monograph 91, 1970.