Stemming Algorithms - A Case Study for Detailed Evaluation

David A. Hull

Rank Xerox Research Centre
6 chemin de Maupertuis
38240 Meylan, France
hull@xerox.fr

June 7, 1995

Abstract

The majority of information retrieval experiments are evaluated by measures such as average precision and average recall. Fundamental decisions about the superiority of one retrieval technique over another are made solely on the basis of these measures. We claim that average performance figures need to be validated with a careful statistical analysis and that there is a great deal of additional information that can be uncovered by looking closely at the results of individual queries. This paper is a case study of stemming algorithms which describes a number of novel approaches to evaluation and demonstrates their value.

1 Introduction

The most common experimental methodology in the field of information retrieval (IR) can be summarized as follows. First, propose a new retrieval strategy which is designed to improve the performance of the system. Second, find an experimental text collection with queries and known relevant documents and run a retrieval experiment using the new strategy and a baseline obtained from a standard approach. Third, compute traditional evaluation measures such as precision and recall. If the new strategy scores enough higher than the baseline, then claim that there is evidence of its superiority. A large fraction of the articles published in information retrieval follow this methodology.

There is good reason why this approach to IR has been so enduring and popular. It provides an objective means of evaluation and can be accomplished with a minimum of experimental work when good test collections are available. However, the ease with which such experimental results can be obtained has lulled many researchers into a sense of complacency. Since all the experimental data necessary to produce evaluation results have been compiled exhaustively in advance, these experiments can be successfully run without the researcher reading a single word of text in either a query or a document! One sometimes wonders whether new retrieval methods have been accepted or rejected based solely on an observed difference in the average value of precision or recall. While this is certainly unlikely, the vast majority of papers in information retrieval following this methodology end with a presentation of precision and recall scores and provide little detailed analysis of the results. It is remarkable that people can spend so much time and energy on developing and implementing new retrieval strategies and so little time on the analysis of results, which is crucial to validate these strategies.

In this paper, we will present a detailed analysis of stemming algorithms. The goal will not be to present any dramatic new approaches to the problem, rather it will be to demonstrate a comprehensive and rigorous strategy for analyzing information retrieval experiments. While the results and conclusions may be of interest to many readers, they are encouraged to concentrate on the details of the methodology of evaluation. We hope to demonstrate that a lot of valuable evidence can be obtained in this manner which might otherwise never be found.

2 Background - The Stemming Problem

In information retrieval, the relationship between a query and a document is determined primarily by the number and frequency of terms which they have in common. Unfortunately, words have many morphological variants which will not be recognized by term-matching algorithms without some form of natural language processing. In most cases, these variants have similar semantic interpretations and can be treated as equivalent for information retrieval (as opposed to linguistic) applications. Therefore, a number of stemming or conflation algorithms have been developed for IR in order to reduce morphological variants to their root form.

The problem of conflation has been approached with a wide variety of different methods, as detailed in Lennon, Pierce, Tarry & Willett (1981), including suffix removal, strict truncation of character strings, word segmentation, letter bigrams, and linguistic morphology. Two of the most popular algorithms in information retrieval, the Lovins (1968) stemmer and the Porter (1980) stemmer, are based on suffix removal. Lovins finds the longest match from a large list of endings while Porter uses an iterative algorithm with a smaller number of suffixes and a few context-sensitive recoding rules.

Krovetz (1993) accurately describes the problems associated with these methods. Most stemmers operate without a lexicon and thus ignore word meaning, which leads to a number of stemming errors. Words with different meanings are conflated to the same stem and words with similar meanings are not conflated at all. For example, the Porter stemmer conflates *general, generous, generation*, and *generic* to the same root, while related pairs like *recognize* and *recognition* are not conflated.

In addition, stems produced by suffix removal algorithms are often not words, which makes it difficult to use them for any purpose other than information retrieval. Interactive techniques which require user input, such as term selection for query expansion, will suffer greatly if the user must work with stems instead of real words. It also becomes difficult to perform dictionary look-up without real words. While each conflation class could be represented by a word rather than a stem, there would still not be a one-to-one correspondance between word stems and word definitions. Dictionary look-up is an important feature in many IR applications. For example, in multi-lingual information retrieval, the query may be subject to automatic translation, in which case the system must be able to find each query term in a transfer dictionary. These problems can easily be overcome by an approach to stemming which uses morphological analysis.

The linguistics groups at Xerox[1] have developed a number of linguistic tools for English which can be used in information retrieval. In particular, they have produced an English lexical database which provides a morphological analysis for any word in the lexicon and identifies its base form. There is good reason to expect that this technology would be ideally suited for use as a stemming algorithm. However, this assumption needs to be tested by conducting experiments using IR test collections.

Morphology is a branch of linguistics that studies and describes how words are formed in language and includes inflection, derivation, and compounding. Inflection characterizes the changes in word form that accompany case, gender, number, tense, person, mood, or voice. Derivational analysis reduces surface forms to the base form from which they were derived, and includes changes in the part of speech.

Xerox linguists have developed a lexical database for English (as well as other languages) which can analyze and generate inflectional and derivational morphology. The inflectional database reduces each surface word to the form which can be found in the dictionary, as follows (Xer 1994):

- nouns $\Longrightarrow$ singular (ex. children $\Longrightarrow$ child)
- verbs $\Longrightarrow$ infinitive (ex. understood $\Longrightarrow$ understand)
- adjectives $\Longrightarrow$ positive form (ex. best $\Longrightarrow$ good)
- pronoun $\Longrightarrow$ nominative (ex. whom $\Longrightarrow$ who)

The derivational database reduces surface forms to stems which are related to the original in both form and semantics. For example, government stems to govern while department is not reduced to depart since the two forms have different meanings. All stems are valid English terms, and irregular forms are handled correctly. The derivational process uses both suffix and prefix removal, unlike most conventional stemming algorithms which rely solely on suffix removal. A sample of the suffixes and prefixes which can be removed is given below (Xer 1994):

[1]Natural Language Theory and Technology (NLTT) at the Xerox Palo Alto Research Center and Multi-Lingual Theory and Technology (MLTT) at the Rank Xerox Research Center in Grenoble, France

- suffixes: ly, ness, ion, ize, ant, ent, ic, al, ic, ical, able, ance, ary, ate, ce, y, dom, ee, eer, ence, ency, ery, ess, ful, hood, ible, icity, ify, ing, ish, ism, ist, istic, ity, ive, less, let, like, ment, ory, ous, ty, ship, some, ure
- prefixes: anti, bi, co, contra, counter, de, di, dis, en, extra, in, inter, intra, micro, mid, mini, multi, non, over, para, poly, post, pre, pro, re, semi, sub, super, supra, sur, trans, tri, ultra, un

The databases are constructed using finite state transducers, which promote very efficient storage and access. This technology also allows the conflation process to act in reverse, generating all conceivable surface forms from a single base form. The database starts with a lexicon of about 77 thousand base forms from which it can generate roughly half a million surface forms (Xer 1994).

3 Description of previous work

There have been a large number of studies which have examined the impact of stemming algorithms on information retrieval performance. Frakes & Baeza-Yates (1992) provides a nice summary, reporting that the combined results of previous studies make it unclear whether *any* stemming is helpful. In those cases where stemming is beneficial, it tends to have only a small impact on performance, and the choice of stemmer among the most common variants is not important. However, there is no evidence that a reasonable stemmer will hurt retrieval performance.

In contrast, a recent study by Krovetz (1993) reports an increase of 15-35% in retrieval performance when stemming is used on some collections (CACM and NPL). Krovetz mentions that these collections have both queries and documents which are extremely short, so that the results make sense given that the likelihood of an exact match of surface form should be proportional to the size of the document. For collections with longer documents (TIME and WEST), stemming algorithms are accompanied by a much more modest improvement in retrieval performance.

Krovetz (1993) also develops stemmers based on inflectional and derivational morphology. He finds that these approaches provide about the same improvements as were obtained by the Porter algorithm. However, he notes that the derivational stemmer performed slightly better than the inflectional stemmer over all four collection that he examined and that the majority of its benefit came at low recall recall levels (i.e. when relatively few documents are examined). We should mention that Krovetz's inflectional and derivational stemmers are not equivalent to the Xerox linguistic tools. In particular, he experiments with a smaller collection of suffixes for derivational analysis and it does not appear that he examines prefixes at all (perhaps wisely, as we shall discover). Therefore, his results are not directly comparable to the ones presented in this paper.

4 The Experimental Collection

Our retrieval experiments will use the document collection constructed for the TREC/TIPSTER project. The TREC collection consists of over a million documents (3.3 gigabytes of text) obtained from a variety of sources, including newspapers, computer journals, and government reports. As part of the exhaustive evaluation studies conducted during the TIPSTER project, analysts have constructed 200 queries and evaluated thousands of documents for relevance with respect to these queries. The collection is so large that we have decided to select a subset of the queries and/or documents to use in our experiments.

We use the properties of the stemming problem to help us determine how to apply the TREC collection in our experiments. Previous research has suggested that impact of stemming algorithms on retrieval performance is small, and thus difficult to detect. Therefore, it is important to use as many queries as possible in the analysis, as this will make it easier to detect significant differences between methods using statistical testing. Also, we can have more faith in our conclusions by testing the stemming algorithms on as large a vocabulary as possible. Therefore, we use all 200 TREC queries in our experiments. Given the vast range of potential user requests, even the complete query set seems like a quite inadequate sample.

Since adding more queries definitely improves the quality of the analysis, we might expect a similar benefit from using as many documents as possible. However, when a sufficient sample of documents is already available, the marginal value of adding more documents is limited. In addition, the reality of storage restrictions forces us to be a bit more selective. Our approach to stemming requires that the entire document collection be reindexed for each new stemming algorithm. Storing a half dozen different index files for the full TREC collection is not possible given our current storage allowance. While IR systems can be designed so that conflation classes are generated at retrieval time, such an approach would have required substantial re-engineering of our system.

A breakdown of the relevance judgements by subcollection reveals that the Wall Street Journal has nearly twice as many relevance judgements as its nearest competitor and has at least one relevant document for every one of the 200 queries. Of the roughly 330,000 relevance judgements provided with the first two disks of the TIPSTER collection, about 140,000 of them refer to the Wall Street Journal. Therefore, we have selected the Wall Street Journal, which consists of 550MB of text and about 180,000 articles for our stemming experiments.

The TREC topic descriptions, from which queries are constructed, are long and detailed, providing a very explicit definition of what it means for a document to be relevant. In fact, with an average of 130 terms per topic, they are almost comparable in length to the documents in the Wall Street Journal subcollection (median length = 200 terms). It is common practice to use the complete topic description as a full-text query. The TREC experiments have frequently been criticized for the length of the topics, since this is in such marked contrast to user behavior when querying the typical commercial retrieval system, where queries as short as one or two terms are much more common. While the precision and detail of TREC topics is extremely valuable for making accurate relevance judgements and encouraging researchers to develop sophisticated retrieval strategies, the experimental results may not reflect the tools and techniques that will be most valuable for operational systems.

In order to address this issue, we have constructed shorter versions of the topics which attempt to summarize the key components of the query in a few short phrases (average length = 7 words). This follows the lead of Voorhees (1994) and Jing & Croft (1994) who use the description statement as a summary for the full query. In contrast to their approach, we construct the new queries by hand, as it was felt that some of the description statements did not use important key words. There is certainly an element of subjectivity in this approach, but the queries were constructed without regard to the specific properties of stemming algorithms.

5 The retrieval system and stemming experiments

We used the SMART[2] text retrieval system developed at Cornell University (Buckley 1985) for the information retrieval experiments. We found its high indexing speed (500MB/hr on a Sparc 20) to be extremely valuable for the repeated indexing of the collection that is necessary in stemming experiments. Very frequent terms are removed using a slightly modified version of the SMART stop list, and queries and documents are treated as vectors of term frequencies and analyzed using the vector space model.

Term weighting is used to improve performance. Term frequencies in both queries and documents are dampened by applying the square-root transformation. Document vectors are normalized to have unit length and query term weights are multiplied by the traditional IDF (inverse document frequency = $\log N/n_i$, N = # docs in collection, n_i = # docs containing term i) measure to increase the importance of rare terms. No proper name or phrase recognition is used although these might well be beneficial. Also, no attempt is made to segment long documents, which fortunately are rare in the Wall Street Journal sub-collection.

We will examine the performance of five different stemming algorithms and compare them to a baseline which consists of no stemming at all. Two stemmers are included in the SMART collection; they are a simple algorithm which removes *s*'s from the end of the word and an extensively modified version of the Lovins (1968) algorithm. In addition, we will study the

[2]Available for research purposes via anonymous ftp to ftp.cs.cornell.edu in directory /pub/smart.

Porter (1980) stemmer, and versions of the Xerox English inflectional and derivational analyzers (Xer 1994) slightly modified for the conflation problem.

6 Initial Experiments

The SMART system is used to index the queries and documents separately for each stemming algorithm. Documents are ranked according to their similarity to each query, and the results are passed on to the evaluation algorithms. The final output from SMART is presented in Table 1. What can we conclude from these results? It appears that all stemmers are 4-6% better than no stemming at all according to the 11-pt average precision-recall score, but that there is very little difference between stemmers in terms of retrieval performance. This comes as no real surprise and agrees with the conclusions obtained from many previous studies. It is difficult to draw any additional conclusions using the information that is presented in this table. Unfortunately, the vast majority of experiments presented in the IR literature stop at this point and do not provide a more detailed analysis.

It is worthwhile to make some comments on the evaluation summary produced by the SMART system. This type of table is often taken as a standard for evaluation in that we have seen it reproduced almost verbatim in a fair number of research papers. It takes a considerable amount of cognitive effort to assimilate the hundreds of numbers represented in this table, many of which are redundant. An equivalent amount of information could be conveyed much more concisely. Recall-precision curves are probably best represented graphically with only the averages given in tabular form. The 3-point average is a relic from traditional approaches to evaluation and contains no additional useful information that cannot be obtained from the 11-point average, which is probably more reliable. It might be valuable to retain it in order to compare results to studies conducted in the 70's and 80's when this measure was in favor, but most of them used the 25%, 50%, and 75% points, so it is not even clear if the 3-point average is valuable for that reason.

The document levels (5, 10, 15, 30) are designed for much smaller test collections and should not be used for the TREC collection. Probably something like (10, 50, 100, 500) would be more appropriate. In particular, it is clear that truncated precision provides no additional useful information at the given document levels. Average recall is a meaningless statistic at these small document levels, since it will have no power to differentiate between methods when the queries have hundreds of relevant documents. It is clear from the results presented at the TREC-2 conference (Harman 1993) that researchers using the TREC corpus recognize these problems.

While talking about cognitive load, we should also mention that the presentation of results to four significant digits makes the table a lot more difficult to read. This may seem like a

```
                      Relevant ranked evaluation

Run title:       nostem  remove_s  Lovins   Porter  XLT_infl XLT_deriv
Num_queries:        200       200     200      200       200       200
Total number of documents over all queries
    Retrieved:    20000    20000    20000    20000    20000    20000
    Relevant:     20982    20982    20982    20982    20982    20982
    Rel_ret:       6510     6689     6829     6856     6802     6801
    Trunc_ret:    19794    19805    19817    19798    19788    19837
Recall - Precision Averages:
    at 0.00       0.8174   0.8256   0.8286   0.8119   0.8178   0.8169
    at 0.10       0.6298   0.6409   0.6460   0.6542   0.6548   0.6471
    at 0.20       0.5318   0.5515   0.5545   0.5644   0.5610   0.5599
    at 0.30       0.4567   0.4719   0.4819   0.4808   0.4841   0.4767
    at 0.40       0.3842   0.4040   0.4139   0.4100   0.4104   0.4061
    at 0.50       0.3211   0.3415   0.3524   0.3528   0.3469   0.3495
    at 0.60       0.2533   0.2719   0.2864   0.2809   0.2835   0.2864
    at 0.70       0.1987   0.2114   0.2227   0.2179   0.2211   0.2211
    at 0.80       0.1415   0.1517   0.1603   0.1576   0.1567   0.1584
    at 0.90       0.0841   0.0927   0.0985   0.0989   0.0977   0.0974
    at 1.00       0.0210   0.0220   0.0229   0.0237   0.0237   0.0228
Average precision for all points
   11-pt Avg:     0.3490   0.3623   0.3698   0.3685   0.3689   0.3675
    % Change:                 3.8      6.0      5.6      5.7      5.3
Average precision for 3 intermediate points (0.20, 0.50, 0.80)
    3-pt Avg:     0.3314   0.3482   0.3557   0.3583   0.3549   0.3559
    % Change:                 5.1      7.3      8.1      7.1      7.4

Recall:
    Exact:        0.4252   0.4293   0.4361   0.4401   0.4382   0.4328
    at  5 docs:   0.0634   0.0676   0.0663   0.0678   0.0674   0.0682
    at 10 docs:   0.1150   0.1132   0.1121   0.1114   0.1131   0.1093
    at 15 docs:   0.1430   0.1453   0.1495   0.1477   0.1502   0.1493
    at 30 docs:   0.2233   0.2279   0.2256   0.2300   0.2295   0.2273
Precision:
    Exact:        0.3255   0.3344   0.3414   0.3428   0.3401   0.3401
    At  5 docs:   0.6010   0.6040   0.5950   0.6060   0.6070   0.5980
    At 10 docs:   0.5575   0.5615   0.5545   0.5600   0.5550   0.5495
    At 15 docs:   0.5163   0.5260   0.5250   0.5313   0.5287   0.5267
    At 30 docs:   0.4612   0.4697   0.4710   0.4775   0.4753   0.4747
Truncated Precision:
    Exact:        0.3275   0.3357   0.3431   0.3446   0.3416   0.3414
    At  5 docs:   0.6010   0.6040   0.5950   0.6060   0.6070   0.5980
    At 10 docs:   0.5575   0.5615   0.5545   0.5600   0.5550   0.5495
    At 15 docs:   0.5163   0.5260   0.5251   0.5316   0.5287   0.5267
    At 30 docs:   0.4619   0.4700   0.4717   0.4782   0.4758   0.4752
```

Table 1: Output from SMART evaluation procedure.

Query	Measure	nostem	remove s	Lovins	Porter	Inflect	Deriv
full	11-pt avg	0.349	0.362	0.370	0.369	0.369	0.368
	% change	-	3.8	6.0	5.6	5.7	5.3
short	11-pt avg	0.181	0.200	0.211	0.209	0.203	0.210
	% change	-	10.6	16.8	15.5	12.4	16.2
full	P at 10 docs	0.558	0.562	0.555	0.560	0.555	0.550
short	P at 10 docs	0.316	0.339	0.346	0.358	0.346	0.349

Table 2: 11-pt average precision-recall and average precision at 10 documents retrieved for the stemming algorithms

trivial point, but it can only encourage people to make decisions based on differences in the 3rd or 4th decimal place. In our experience using statistical testing in information retrieval evaluation studies, it is very rare to find a difference of less than 1% (on an absolute scale) in precision or recall which is statistically significant. This may change if studies start to use very large numbers of queries, but for most purposes three decimal places is certainly sufficient.

Applying these suggestions, we prefer to replace the SMART evaluation table with Table 2. The revised table shows the evaluation results for both the full and the short queries. Sharply cutting query size heavily reduces retrieval performance. This is certainly something to keep in mind when reading about the quality of modern experimental retrieval systems, as measured by the TIPSTER evaluations. Our system may get 5-6 relevant documents out of the first 10 with extensive full-text queries but this number drops to an average of 3-4 of 10 for shorter queries, and the average performance figures also conceal a very large variation among individual queries. For example, half the full queries have an absolute difference greater than 0.10 between the best and worst algorithms, and 35 have differences greater than 0.20. For the short queries, the numbers are 109 and 49 respectively.

However, in terms of average precision-recall, the two query types exhibit very similar performance. The percent difference figures from the table are a little misleading, since the short queries have a baseline that is so much lower, therefore we will tend to use the absolute difference, as in the previous paragraph. The inflectional stemmer looks slightly less effective for short queries, but it is hard to be certain. The average precision at 10 docs for short queries also has a similar pattern, although the Porter stemmer seems to perform a bit better than the other alternatives.

The average precision at 10 docs for the full queries follows a different pattern. When the query is well defined and the user is only looking at a few documents, stemming provides absolutely no advantage. This helps to explain a bit of the inconsistency in the literature. Lennon et al. (1981) evaluates at 5 and 20 documents retrieved and Harman (1991) evaluates at 10 and 30 documents retrieved and find no difference between stemmers. Harman's average precision-recall scores look very similar to our own. Krovetz (1993) reports much larger improvements for stemming in collections where both queries and documents are short, but the other collections show a similar 4-5% improvement in performance.

It seems natural to consider stemmers as recall enhancing devices, as they are creating more matches for each query term. Our results reflect this hypothesis, as stemming is no help at low recall and provides some benefits when performance is averaged over a number of recall levels. This suggests that there may be greater benefits to stemming when evaluating at higher recall levels. Stemming is slightly more helpful for short queries, as demonstrated by the fact that no stemming leads to a drop in precision at 10 docs for short queries but not for long ones. As Krovetz demonstrates, stemming becomes much more valuable when both queries and documents are short, as this is when it is most difficult to find term matches between query and document.

7 Revised Evaluation Measures

We already have a pretty good idea of when stemming is useful, but it would be nice to have a more complete picture of average performance. In evaluation, it is important to find an acceptable compromise between the amount of evaluation data presented and the ability of the researcher to assimilate it. Tague-Sutcliffe & Blustein (1995) demonstrates that most of the evaluation measures currently in favor are highly correlated, so presenting all of them is a waste of effort. We choose to use three different measures, average precision at 11 recall points (APR11), average precision at 5-15 documents examined (AP[5-15]), and average recall at 50, 60, . . . 150 documents examined (AR[50-150]). The first measurement is the current evaluation standard and captures a wide range of different performance characteristics. The second measurement is designed to estimate performance for shallow searches and the third is chosen to capture a more in-depth inquiry by the user.

One might wonder why we choose to average at 5-15 and 50-150 documents examined rather than using 10 and 100 documents respectively, since the latter measures are much more intuitive. This is for two reasons. First, averaging over a range of values captures more information than using a single document level. For example, AP[5-15] differentiates between document rankings of (1,2,3) and (8,9,10) while average precision at 10 docs gives them an identical score. Second, when evaluating non-parametric statistical tests over these evaluation measures, it is important to have as few ties as possible to satisfy the underlying distributional assumptions of the tests. Averaging helps to break ties. Statistical testing will be described in the next section. Table 3 presents the revised evaluation scores. Since these results nearly duplicate the ones presented in the previous section, there is little need for additional commentary. AR[50-150] behaves very similarly to APR11.

Unfortunately, it sometimes happens that average measurements do not adequately describe

Query	Measure	nostem	remove s	Lovins	Porter	Inflect	Deriv	sig. test
full	APR11	0.348	0.361	0.369	0.367	0.368	0.366	RLPID $\succ$ N
full	AP[5-15]	0.556	0.562	0.557	0.562	0.562	0.555	-
full	AR[50-150]	0.414	0.423	0.427	0.429	0.428	0.426	LPID $\succ$ N
short	APR11	0.179	0.198	0.209	0.206	0.201	0.208	RLPID $\succ$ N
short	AP[5-15]	0.313	0.339	0.343	0.356	0.345	0.354	RLPID $\succ$ N
short	AR[50-150]	0.263	0.282	0.288	0.290	0.285	0.288	RLPID $\succ$ N

Table 3: Revised evaluation scores

overall performance. For example, if a few queries have a huge variablility in performance between methods, while the rest have much smaller differences, then the average scores will be dominated by the high variance queries and completely insensitive to trends in the remainder of the data. For this reason, we present an alternative evaluation measure that is based on the ranking of scores between methods for each query.

Query	nostem	remove s	Lovins	Porter	Inflect	Deriv
Q178	0.228	0.170	0.486	0.486	0.191	0.169
ranks	4	2	5.5	5.5	3	1
Q187	0.127	0.129	0.117	0.124	0.132	0.188
ranks	3	4	1	2	5	6

Table 4: Example of within-query ranking

Table 4 presents the APR11 scores for two of the short queries. Clearly, query Q178 has more variability than query Q187. We can normalize for unequal variance by replacing the score for each method by its ranking with respect to the scores of other methods for the same query. Most people would naturally wonder if this is desirable. Surely Q178 is demonstrating a much more important difference than Q187 which is completely lost when the scores are ranked. The answer to this is maybe, but maybe not. In this example, Q178 has only 3 relevant documents while Q187 has 429! Most of the variability in Q178 is accounted for by the change in ranking of a single relevant document. In this example, it could well be the case that the differences described by Q187 are more reliable. Since in our experiment, the number of relevant documents per query ranges from 2 to 591, there is good reason to believe that it might be worthwhile to examine an evaluation measure that normalizes for unequal variance[3]. Using the ranking strategy described above, the evaluation results can be summarized by taking the average rank over all queries, as presented in Table 5.

How does one compare average ranks and average precision-recall scores? It is hard to say, since the measures are on different scales. The disadvantage of ranks is that any sense of absolute performance is lost. The ranks also depend on which methods are being compared. Remove one

[3]This problem is one of the unfortunate side-effects of using of subset of the TREC collection. If we used the full collection, we would have less variation in the number of relevant documents per query.

Query	Measure	none	rem s	Lov	Port	Infl	Der	sig. test
full	APR11	2.70	3.12	3.83	3.76	3.79	3.81	LPID $\succ$ R $\succ$ N
full	AP[5-15]	3.34	3.48	3.66	3.57	3.58	3.37	-
full	AR[50-150]	2.83	3.29	3.68	3.63	3.79	3.79	LID $\succ$ R $\succ$ N, P $\succ$ N
short	APR11	2.62	3.25	3.70	3.79	3.68	3.96	LPID $\succ$ R $\succ$ N
short	AP[5-15]	3.05	3.45	3.55	3.69	3.52	3.76	RLPID $\succ$ N
short	AR[50-150]	2.63	3.20	3.60	3.91	3.70	3.97	LPID $\succ$ R $\succ$ N, D $\succ$ L

Table 5: Average rank measurements

stemming algorithm from the analysis and the ranks will change, perhaps significantly. However, there are a number of advantages as well. For example, it is now simple to compare between the different evaluation scores and also between short and long queries, since the average ranks of all evaluation measures are on the same scale. Note that average performance is 3.5 for a ranking based on six methods.

While the overall pattern is about the same, there are several noticeable differences between the average ranks for the full and short queries.

(1) AP[5-15] nostem - No stemming is proportionally less effective for short queries. We can thus conclude that stemming always improves performance for short queries.

(2) AP[5-15] Deriv - The derivational stemmer works better over the short queries, where it can help to find a match, than on the long queries, where it adds noise to the results.

(3) AR[50-150] Porter - The Porter stemmer appears to work better with the short queries than the long queries at high recall.

Since rank based analysis is only a relative measure, it should not be used as the only means of evaluation, but it provides a nice supplement to the data provided by average precision and recall.

8 Statistical Testing

In some experiments, it is obvious from the evaluation scores that one method is much more effective than another. However, differences in performance are often small and it is hard to know whether the observed differences actually represent a consistent effect or are an artifact of the sample of queries used in the experiment. Statistical hypothesis testing can provide valuable evidence about whether the experimental results have a more general significance. Statistical testing is particularly important for the stemming data, where the observed differences are only a few percentage points of precision or recall. Previous articles on stemming algorithms have found similar results and researchers have disagreed about whether they represent a real improvement.

Hypothesis tests make the preliminary assumption that all retrieval strategies are equally effective. The test determines the probability that the observed results could occur by chance (or p-value) given the initial hypothesis. If the p-value is very small, then evidence suggests that the observed effect reflects an underlying difference in performance. Statistical testing is important because the queries used for testing represent only a very small sample from the set of all possible queries. If the number of queries is small relative to the observed difference in evaluation scores, then the experimental results are not generally applicable. It is important to recognize that all conclusions are based on the assumption that the known queries are a representative sample from the set of potential queries. If the test queries are excessively biased towards particular subtopics in the collection, then the experimental results will not be generally applicable regardless of the verdict of the statistical analysis.

Our approach to statistical testing is based on the Analysis of Variance (ANOVA), which is useful for detecting differences between three or more experimental methods. The two-way ANOVA assumes that the evaluation scores can be modelled by an additive combination of a query effect and an effect due to the choice of experimental method. When these effects are factored out, it is assumed that the remaining component of the score is random error, and that all such values are drawn from an identical Normal distribution, independent of query or method. It is unlikely that these assumptions will be strictly accurate, but it is hoped that they represent a reasonable approximation. Testing these assumptions will be part of the work of the statistical analysis.

The same approach can also be applied to the ranking data described in the previous section, and this alternative is known in the statistical literature as the Friedman Test. In order to run the tests, the evaluation scores must be computed separately for each query and method, generating a table which is used as input to the statistical algorithms. Hull (1993) gives a mathematical description of the statistical tests, which shows how they can be run. Alternatively, these statistical methods can be found in most elementary statistics texts on experimental design, such as Neter, Wasserman & Kutner (1985) for the ANOVA and Conover (1980) for the Friedman test.

The ANOVA and the Friedman Test operate in two stages. First, a test statistic is computed which indicates whether there is any evidence of a difference between the methods. If the results indicate that there is a difference (in our experiments, we define a significant difference as one which produces a p-value less than .05), then a separate multiple comparisons test is applied to determine which contrasts are significant. In this paper, we examine all pairwise differences in average scores. The results are presented in Table 3 and Table 5 above in a compact form. A description of how to interpret those results is given below.

The expression $x \succ y$ indicates that method x is significantly better than method y according to the appropriate statistical test (ANOVA for the average scores, Friedman for the average ranks). The abbreviations are: N = nostem, R = remove s, L = Lovins, P = Porter, I = inflectional, D = derivational. For example, for AR[50-150] average ranks using short queries, the entry is:

$$\text{LPID} \succ \text{R} \succ \text{N, D} \succ \text{L}$$

which indicates that Lovins, Porter, Inflectional, and Derivational stemmers are all better than remove s, which is in turn significantly better than no stemming at all. Furthermore, the Derivational stemmer is also significantly better than the Lovins stemmer.

The test results based on ANOVA are quite consistent. All of the others stemmers are significantly better than no stemming, except for AP[5-15] with the full queries where there is no significant difference between methods. The magnitude of a significant difference is roughly 0.01, except in the case of AP[5-15] for short queries, where it is 0.023. Therefore, with 200 queries, we can detect very small absolute differences in performance.

The Friedman test finds more significant differences than the ANOVA. In particular, it consistently finds that remove s is less effective than the other stemmers except for AP[5-15]. The magnitude of a significant difference in ranking is roughly 0.35. There are some puzzling inconsistencies in the statistical results. For example, the derivational stemmer is significantly better than the Lovins stemmer using AR[50-150] short query rankings, but the actual recall scores of the two methods are equal to three decimal places. We will come back to this result at the end of the next section.

It is worth noting that the Friedman Test appears to be consistently more powerful than the ANOVA, in the sense that it finds more significant differences between methods. This is a bit surprising, since the Friedman Test is a nonparametric test (not model based), and tests from this family are usually less powerful than the ANOVA, when its assumptions are reasonable. This suggests that it would be a good idea to check those assumptions.

The ANOVA model assumes that the error variance is constant across queries. Given that each query has a widely variable number of relevant documents, there is good reason to expect that this condition may be violated. As we saw in the previous section, the evaluation score of queries with very few relevant documents may change dramatically with the change in rank of a single relevant document. We can examine the data more closely to see how badly the assumption of equal variance is violated.

Let us presume that the variance is constant and equal to the error variance obtained from the ANOVA model, which we will denote as σ^2. It is well known in statistics (Box, Hunter & Hunter 1978) that:

$$\frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1}$$

where s^2 is the variance of a sample of size $n-1$ drawn from a normal population with variance σ^2. We can compute s^2 for each query and compare the distribution of the test statistic above to the appropriate chi-square reference distribution. In particular, we determine the expected value of maximum of a sample of 200 observations drawn from the chi-square distribution. If our data is actually drawn from this distribution, we would expect an average of one observation to equal or exceed this value. In practice, we find that between 11-18 observations exceed the expected maximum for each of our experiments.

The results above indicate that without question, the assumption of equal variance is violated. We have suggested that queries with few relevant documents will tend to have higher variability than those with many relevant documents. We examined this hypothesis and found that while a few of the queries with highest variability had fewer than 10 relevant documents, there was no consistent pattern. We also tested whether variability in performance was significantly correlated with average performance and also obtained a negative result. From this analysis, we believe that the results from the Friedman Test should be more reliable for our experiments. It also provides evidence that examining average rank data as well as average score data is valuable in determining which experimental methods are most effective.

The reader should keep in mind that the statistical tests described here have been chosen for this particular experiment. In general, the optimal test may depend on the number of experimental methods being compared. For instance, both Friedman and ANOVA have different forms when there are only two methods (Hull 1993), and Conover (1980) suggests that there is a more powerful alternative to the Friedman Test for cases when three or four methods are being compared.

9 Identifying Important Queries

In the previous section, we obtained strong evidence that variance is not constant over queries. Since queries with high variance are much more important in determining the average performance of an experimental method than their low variance counterparts, it is important to understand what kinds of document ranking patterns cause this behavior. We have found three major patterns which can lead to high variance:

(1) A lot of relevant documents are ranked higher in some methods than in others.

(2) The query has very few relevant documents and thus scores are sensitive to changes in the ranking of one or two relevant documents.

(3) The query is hard and relevant documents tend to have low ranks in general. However, some methods rank a few relevant documents relatively well, which can make a large difference for many evaluation measures.

Clearly, factor (1) describes the type of behavior we are looking for and these queries deserve to be considered very important. Some people might argue that queries of type (2) and (3) are also important, because they demonstrate real changes in the ranks of relevant documents, however we believe that this is a dangerous conclusion. If only a few relevant documents are being ranked higher, it is hard to know if this is due to the fact that they are relevant or some reason unrelated to relevance. We would be particularly suspicious of type (3) queries, for one would hope that a method which is valuable would improve the ranking of a reasonably large sample of relevant documents. This does not mean that an effect which improves the rank of a single relevant document is not real and important in some cases, merely that this is a less reliable indicator of value than a method which improves the rank of many relevant documents.

Therefore, looking only at the queries which satisfy factor (1) would be a valuable exercise. One could remove the effect of factor (2) by deleting all queries with few relevant documents. While this approach might be interesting, it does not address factor (3), and some queries with few relevant documents do demonstrate consistent behavior as defined by factor (1). Fortunately, we can separate queries of type (1) from the others using the same statistical methods applied in the previous section.

The traditional approach to statistical testing determines that one experimental method is better than another when it performs consistently better over a large number of queries. Suppose one examines only a single query and asks if one experimental method is significantly better than another just with respect to that query. This question can be addressed using a statistical test on the ranks of relevant documents, and the answer is that a method is significantly better if it equals or improves the rank of the majority of the relevant documents relative to the non-relevant documents.

However, the problem is not as simple for a single query. First, one would like the test to ignore cases where a method simply rearranges the ranks of the relevant documents, therefore relevant documents should each be ranked independently with respect to only the non-relevant documents. Second, it is difficult to determine the best way to measure the importance of a change in ranking. For example, a change in rank from 15 to 1 is probably more important than a change from 1000 to 500 although the absolute rank difference is much smaller. Therefore, using the ranks directly in a statistical test seems like a bad idea. One might consider using the actual document similarity scores, but these would then have to be normalized between different methods, and this is a very difficult problem.

Fortunately, there is a simple solution which avoids having to select an adhoc normalizing algorithm. The Friedman Test looks only at relative rankings and thus is insensitive to the type of problem described in the previous paragraph. The Friedman Test is applied as follows. Create a table whose rows are the relevant documents and whose columns are the experimental method. Each entry in the table is the rank of the given relevant document for that particular experimental method with respect to all the non-relevant documents. The Friedman Test replaces the similarity rank of the document with its performance rank relative to the other experimental methods. This method can be used to judge statistical significance on a query by

		full query rank < 200		short query rank < 200	
		sig	not	sig	not
all	sig	125	13	139	34
docs	not	2	60	3	24

Table 6: Per-query significance tables

query basis.

We do not intend to examine the results of per-query significance tests individually, as this would require a lot of human effort. Rather, we intend to use this approach as a filter to identify those queries which clearly demonstrate a significant difference, as they should be the most valuable for making general conclusions about the performance of different experimental methods. Therefore, the strategy is to compute average performance over only those queries which demonstrate individually that they describe a significant difference between methods. This will filter out all queries of type (2) and (3) above, since they will not produce consistent differences in relevant document rank. However, it will also eliminate queries where there is truly no difference between methods, so the reader is cautioned that the performance differences found by this approach do not represent unbiased estimates of the actual difference between methods, since variation will probably be magnified by this approach.

There are some issues to consider before applying this method. First, poorly ranked documents will tend to have much more variability in their ranks than highly ranked documents. This is because the density of non-relevant documents will be much higher in the low similarity score range. Second, rank changes among poorly ranked documents are just as important as rank changes among highly ranked documents for the test statistic, which is probably not a desirable property. We have some concerns that random variation in the ranks of low scoring documents may obscure significant improvements in the ranks of high scoring documents. For this reason, we conduct two different versions of the test, one over all relevant documents, the second over only those relevant documents which have a rank of less than 200 for at least one evaluation method. A comparison of the approaches is presented in Table 6.

All queries which have a p-value $< .05$ according to a Friedman Test based on the ranks of their relevant documents are considered significant. The contingency table indicates that very few (2 or 3) queries have significant results for the high scoring documents obscured by lots of lower ranking documents, so this is not an important problem. It is much more often the case (13 or 34 queries) that a query has significant variability when all documents are analyzed but not when the analysis is restricted to high scoring documents. For the most part, this occurs when the sample of high scoring documents is too small to obtain a significant difference. We decided to define the important queries as those which demonstrate significant differences according to both approaches above, which leaves us with 125 full queries and 139 short queries

Average Scores

Query	Measure	nostem	remove s	Lovins	Porter	Inflect	Deriv	sig. test
full	APR11	0.375	0.393	0.402	0.399	0.400	0.397	RLPID $\succ$ N
full	AP[5-15]	0.623	0.640	0.628	0.642	0.640	0.631	-
full	AR[50-150]	0.377	0.391	0.394	0.398	0.397	0.391	RLPID $\succ$ N
short	APR11	0.208	0.233	0.244	0.241	0.238	0.245	RLPID $\succ$ N
short	AP[5-15]	0.374	0.410	0.412	0.429	0.421	0.426	RLPID $\succ$ N
short	AR[50-150]	0.259	0.284	0.287	0.294	0.290	0.288	RLPID $\succ$ N

Average Ranks

Query	Measure	none	rem s	Lov	Port	Infl	Der	sig. test
full	APR11	2.50	3.19	3.82	3.74	3.88	3.88	LPID $\succ$ R $\succ$ N
full	AP[5-15]	3.24	3.52	3.50	3.64	3.58	3.51	-
full	AR[50-150]	2.66	3.32	3.78	3.70	3.84	3.70	RLPID $\succ$ N, LI $\succ$ R
short	APR11	2.51	3.24	3.58	3.90	3.85	3.92	PID $\succ$ R $\succ$ N, L $\succ$ N
short	AP[5-15]	2.94	3.44	3.52	3.70	3.69	3.71	RLPID $\succ$ N
short	AR[50-150]	2.54	3.22	3.40	3.99	3.89	3.97	PID $\succ$ LR $\succ$ N

Table 7: Evaluation scores for the important queries

if the remainder are filtered out. Table 7 presents the average evaluation scores for this subset.

The significant differences increase to roughly 0.015 (0.032 for short AP[5-15]) for average scores and 0.42 for average ranks, due to the fact that fewer queries are being analyzed, however the results remain roughly the same. This reassures us that the potential problems that we discussed at the beginning of the section do not change the results a great deal. We will concentrate on the average rank statistics since they are directly comparable to the rank statistics computed using all queries.

For the full queries, the results are basically identical, although the average rank of no stemming drops even further relative to the other methods. This is probably a result of removing the queries where stemming is not important and indicates that when stemming is important, not stemming is rarely the right decision. For the short queries, there is another interesting pattern. The Lovins stemmer appears to be less effective while the Porter and Inflectional stemmer are proportionally more effective according to average rank. However, this pattern is not duplicated in the average scores, where the Lovins stemmer is fully competitive with the other stemmers. A closer look at the scores reveals that the Lovins stemmer performs slightly worse for a lot of queries and much better for a few queries, which explains the lower score with respect to the rank based measure.

One possible explanation is that Lovins stems more heavily than all the other stemmers. This means that it conflates a lot more terms, which reduces performance slightly on a lot of queries by adding extra noise. However, occasionally an additional important term is recognized, which helps performance a lot. Therefore, while average performance is the same, choosing the

Q#	score	Lovins	Porter	Inflect	Deriv
86	61.37	0.216	0.225	0.230	0.546
39	57.27	0.092	0.343	0.365	0.094
21	53.5	0.596	0.302	0.278	0.484
82	39.23	0.438	0.282	0.151	0.406
143	26.11	0.199	0.101	0.007	0.249
128	24.87	0.246	0.047	0.251	0.255
182	24.42	0.305	0.341	0.262	0.115
98	24.37	0.501	0.507	0.326	0.331
147	20.27	0.032	0.186	0.187	0.034
70	15.9	0.635	0.496	0.495	0.647

Table 8: Revised evaluation scores

Lovins stemmer may degrade performance slightly for a lot of queries in exchange for helping more significantly with a few queries. Note that this hypothesis has not been verified explicitly and applies only to the short queries when evaluated using APR11 or AR[50-150].

10 Detailed Analysis

While we have learned a lot from our extensive analysis of the average performance figures, it has not really helped us find out exactly why one stemming algorithm works better than another. If we wish to improve the current technology in stemming algorithms, it is important to see specific examples where the type of stemming makes a large difference in performance. Therefore, we will take a detailed look at a number of individual queries and figure out which word stems account for differences in performance. For this section of the analysis, we rely only on the short queries, for it will be far easier to identify the important word stems in these examples.

At this point, it is very clear that no stemming and remove s are not really competitive with the other stemming algorithms, so they will not be considered further. Among the others we have found little evidence which suggests how to choose between them, other than a hint that the Lovins stemmer may have slightly different behavior. For the detailed query analysis, we start with only those queries which were judged important based on a per-query Friedman Test, as described in the previous section. We then compute the error variability of the queries and rank them in decreasing order of variability, just as we did when testing the equal-variance assumption for ANOVA, except now we are working only with four stemming algorithms. We compute the chi-square score for each query and find that 11 exceed the expected maximum score of 12.07 for a sample of 139 queries with equal variance. The top 10 of these are presented in Table 8.

The queries shown above can be divided into 6 categories according to their behavior.

(1) Lovins, Deriv $\gg$ Inflect, Porter - 21, 70, 82, 143

(2) Inflect, Porter $\gg$ Lovins, Deriv - 39, 147

(3) Deriv $\gg$ others - 86

(4) Deriv $\ll$ others - 182

(5) Porter $\ll$ others - 128

(6) Lovins, Porter $\gg$ Inflect, Deriv - 98

We examine each of these categories in more detail.

(1) Lovins, Deriv $\gg$ Inflect, Porter

Q21: superconductivity vs. superconductor

Unstemmed Word	Lovins	Porter	Inflect	Deriv
superconduct(ed/ing)	superconduc	superconduct	superconduct	conduct
superconduction	superconduc	superconduct	superconduction	conduct
superconductive	superconduc	superconduct	superconductive	conduct
superconductively	superconduc	superconductiveli	superconductively	conduct
superconductivity('s)	superconduc	superconductiviti	superconductivity	conduct
superconductor(s/'s)	superconduc	superconductor	superconductor	conduct

The query talks about research in *superconductivity* while many of the documents refer only to *superconductors*. Only the Derivational and the Lovins stemmers make this match. Note that the Derivational stemmer overstems by removing the prefix and so loses a bit in performance for this reason.

Q70: surrogate motherhood vs. surrogate mother

The query asks about surrogate *motherhood* while many of the documents talk about surrogate *mothers*. Only the Derivational and the Lovins stemmers make this connection.

Q82: genetic engineering vs. genetically engineered product

Unstemmed Word	Lovins	Porter	Inflect	Deriv
genetic('s)	genet	genet	genetic	genetic
genetically	genet	geneticalli	genetically	genetic
geneticist(s/'s)	genet	geneticist	geneticist	genetic
genetics('s)	genet	genet	genetics	genetics

The Derivational and Lovins stemmers relate *genetic* and *genetically*. The Inflectional stemmer lost out even more because *engineering* did not get conflated to *engineer*.

Q143: U.S. government protection of farming

Unstemmed Word	Lovins	Porter	Inflect	Deriv
farm(s/'s/ed)	farm	farm	farm	farm
farming('s)	farm	farm	farming	farm
farmer(s/'s)	farm	farmer	farmer	farm

The Inflectional and Porter stemmers did not recognize that *farmer* and *farming* were related. The Inflectional stemmer did not conflate *farming* and *farm* and thus was completely ineffective.

The behavior of the inflectional stemmer with respect to *farming* and *engineering* deserves a more complete explanation. Both words can be either a noun or a verb. According to inflectional rules, the noun form should not be stemmed while the verb form should. We do not apply a part-of-speech tagger to the text before stemming and are therefore forced to make an arbitrary decision. Our decision not to stem *engineering* and *farming* turns out to be a poor one for IR performance, although it may be the right one from the linguistic perspective in the absence of a part-of-speech tag.

(2) Inflect, Porter $\gg$ Lovins, Deriv

Q39: client-server architectures

The derivational and the Lovins stemmer equate *server* with *serve*. This is a very bad decision since *serve* is a common term used in a number of contexts and *server* has a much more specific meaning, particularly in the domain of computers and technology.

Q147: productivity statistics for the U.S. economy

The Derivational and the Lovins stemmer equate *productivity* and *produce*. One again, this turns out to be a bad decision for the reasons described in the previous example.

Note the contrasts between sections (1) and (2). Conflating *farmer* to *farm* is good but conflating *server* to *serve* is bad. *Superconductivity* should definitely be related to *superconduct* but converting *productivity* to *produce* turns out to be a bad decision. This clearly indicates that it is impossible to produce an ideal stemmer using only suffixing rules. There are only two ways to make a distinction in the examples above. One is to construct new rules or exception lists by hand, the other is to identify which conflation pairs are used in similar contexts by conducting a corpus-driven analysis. One such approach is presented in Croft & Xu (1995).

(3) Deriv $\gg$ others

Q86: bank failures

The Derivational stemmer converts *failure* to *fail*. The other stemmers do not make this connection. This is an example where linguistic knowledge can recognize a special case which is missed by suffixing rules.

(4) Deriv $\ll$ others

Q182: commercial overfishing

The Derivational stemmer converts *overfishing* to *fish*. This is the second example where prefix removal is harmful.

(5) Porter $\ll$ others

Q128: privatization of state assets

The Porter stemmer equates *privatization* with *private*. This is certainly unfortunate for information retrieval. The irony here is that the Derivational stemmer would probably have made the same mistake, but the word *privatization* is not in the lexicon! This points out another potentially serious problem that is shared by both the Inflectional and Derivational stemmer. Since both stemmers analyze on the basis of linguistic rules, a word must be in the lexicon in order to be recognized.

We make the decision not to stem any word which does not appear in the lexicon. While this works well for the most part, since most unrecognized words are proper names, there is a distressingly large list of exceptions. This is a particular problem when using the Wall Street Journal, because a lot of the important financial and economic terminology is not included in our general purpose lexicon. Many of these words are valuable index terms. Furthermore, there is a constant influx of new vocabulary into language, which will be reflected in newspaper reporting. It might be a good idea to construct a guesser which applies a rules-based algorithm when the word does not appear in the lexicon, because at the moment the linguistic stemmers do not even conflate plurals of unidentified nouns. We have implemented one special rule to remove possessives ('s) since this is obviously an important factor for proper names and the rule is extremely unlikely to make any errors. A more comprehensive guesser could easily be constructed from traditional suffix removal algorithms (like Porter or Lovins).

(6) Lovins, Porter $\gg$ Inflect, Deriv

Q98: manufacturers of fiber optics equipment

The inflectional and derivational stemmers do not stem *optics* to *optic*. For the Derivational stemmer, this is a linguistically motivated decision which turns out to be very unfortunate for information retrieval.

The detailed query analysis has been very informative. We have recognized that there are many examples where the suffix-removal rules do not produce ideal behavior. We have also discovered that the Inflectional and Derivational stemmer make a number of decisions for linguistic reasons that are not optimal for information retrieval. Also, the two linguistic stemmers could probably be improved by adding domain-specific lexicons or by building a guesser for unknown terms based on traditional stemming algorithms.

Unfortunately, it is not so easy to get useful information out of a detailed analysis in many information retrieval experiments. For example, routing experiments work with large numbers of variables and the generated queries are constructed using hundreds of relevant documents, so it is very difficult to determine what factors are most significant for a particular query. In particular, it may not be a single factor, but a combination of small improvements from a number of different factors which leads to a real difference in performance. Detailed evaluation is also costly in terms of human resources, and there is no guarantee that solutions to problems found with individual queries will generalize to produce algorithms which improve performance on a larger scale. We address one such example in the next section. Nonetheless, we contend that this approach to evaluation is valuable for studying stemming algorithms, and one can certainly imagine other IR problems where it could be helpful.

11 Modified Derivational Stemmer

The detailed query analysis has revealed that prefix removal is generally a bad idea for a stemming algorithm. The derivational stemmer suffers in performance on a number of occasions for this reason. For example, prefix removal for the terms *overfishing* (Q128) and *superconductor* (Q21) is certainly undesirable. A quick scan over the unexamined queries reveals several more instances where prefix removal causes problems: Q1 - *antitrust, illegal* and Q4 - debt *rescheduling.*

		Prefix Removal		
Query	Measure	Y	N	sig?
full	APR11	0.366	0.368	N
full	AP[5-15]	0.555	0.559	N
full	AR[50-150]	0.426	0.426	N
short	APR11	0.208	0.210	Y
short	AP[5-15]	0.354	0.356	N
short	AR[50-150]	0.288	0.289	N

Table 9: Effect of prefix removal on stemming performance

To verify this hypothesis, we printed out a list of all terms for which prefix removal took place in the Wall Street Journal. It quickly became obvious that in the vast majority of cases, stemming would be undesirable. A lot of prefixes, such as *anti-, un-*, and *il-* reverse the meaning of the root form. Therefore, we created a new version of the Derivational stemmer that only modified the suffix of the word, in order to see whether this would make a significant difference in retrieval performance. Table 9 compares performance to the original version.

Query	Measure	none	rem s	Lov	Port	Infl	Der	sig. test
full	APR11	2.68	3.10	3.81	3.75	3.77	3.90	LPID $\succ$ R $\succ$ N
full	AP[5-15]	3.32	3.47	3.63	3.54	3.56	3.48	-
full	AR[50-150]	2.81	3.25	3.69	3.62	3.78	3.87	LPID $\succ$ R $\succ$ N
short	APR11	2.63	3.25	3.68	3.75	3.68	4.01	LPID $\succ$ R $\succ$ N
short	AP[5-15]	3.03	3.47	3.55	3.69	3.52	3.75	RLPID $\succ$ N
short	AR[50-150]	2.63	3.20	3.56	3.90	3.70	4.01	LPID $\succ$ R $\succ$ N, D $\succ$ L

Table 10: Ranks with revised Derivational stemmer

The average scores changes very little, but all the differences are in the right direction, which is an encouraging sign. To further examine this question, we replace the old Derivational stemmer with the new one and recompute the average ranks and the Friedman Test, as shown in Table 10. Both Table 9 and Table 10 are computed using the full query set. Although the revised derivational stemmer has slightly higher ranks, there is no significant change in the results. There are simply not enough examples where prefix removal makes a real difference in our limited query sample to make a final conclusion.

12 Conclusions

After an exhaustive analysis, we have reached the following conclusions concerning stemming algorithms.

(1) Some form of stemming is almost always beneficial. The only case where there is not overwhelming evidence in favor of stemming is when the full TREC queries are used and very few documents are examined. The average absolute improvement due to stemming is small, ranging from 1-3%, but it makes a large difference for many individual queries. There are probably particular queries where stemming hurts performance, but we did not investigate this issue.

(2) Simple plural removal is less effective than more complex algorithms in most cases. However, when only a small number of documents are being examined, plural removal is very competitive with the other algorithms.

(3) There is no difference between the other stemmers in terms of average performance. For short queries, the Lovins stemmer tends to perform slightly worse for many queries but a lot better for a few others. This may be a result of overstemming although we have not produced any concrete evidence for this hypothesis.

(4) The detailed query analysis demonstrates that the same suffix-removal rule can be beneficial in some cases and harmful in others. This suggests that rules-based suffix removal may not be the ideal approach to stemming in the long run.

(5) There are a number of problems with using linguistic approaches to word analysis directly as stemming algorithms. First, these methods are based on a lexicon and cannot correctly stem words which are not contained in the lexicon. Second, many decisions about the root form of a word which are properly motivated from the linguistic perspective are not optimal for information retrieval performance. It is clear that linguistic analysis tools must be tailored for information retrieval applications, and potentially further optimization for the particular domain may be important.

(6) The detailed query analysis suggests that prefix removal may be an undesirable feature in stemming algorithms. However, there aren't enough negative examples in the queries examined to prove this hypothesis.

The primary goal of this paper is to present a case study which demonstrates the potential of detailed analysis and the careful use of statistical testing to improve the quality of information retrieval experiments. We tried the following approaches to evaluation to learn more about the characteristics of stemming algorithms.

(1) Multiple versions of the query are used to demonstrate the impact of query length on retrieval performance.

(2) Multiple evaluation measures are used to correspond to different user needs. In particular, average within-query ranking is introduced as a measure which is insensitive to high-variance queries and allows the researcher to easily compare the results produced by different evaluation measures.

(3) Statistical Testing is applied to all evaluation measures to determine which differences are consistent across queries. The assumptions of these tests are examined, and we demonstrate that the ANOVA is not accurate because variance in performance changes greatly from query to query.

(4) We present a method for identifying important queries using statistical testing on a per-query basis which has several valuable functions. It can be used to filter out queries with few relevant documents and queries which don't find consistent differences between methods. Average performance measured over important queries magnifies the contrasts between methods and the technique emphasizes the most useful queries for further detailed analysis.

(5) We give a detailed analysis of individual queries, which provides hard evidence about when and why particular methods succeed or fail and suggests a number of potential improvements for the linguistic stemming algorithms.

Some or all of these techniques should be valuable for any information retrieval experiment that is evaluated using traditional measures.

Of the conclusions about stemming algorithm given above, the last three could not have been made without the detailed query analysis. None of the remaining conclusions could be demonstrated as clearly without using alternative evaluation measures like average rank and the extensive application of statistical testing. This paper demonstrates that information retrieval experiments can really benefit from an analysis that looks far beyond average precision and recall.

Acknowledgements Special thanks go to Gregory Grefenstette for implementing the stemming algorithms based on English inflectional and derivational morphology. We are grateful to Donna Harman and the analysts from the TIPSTER project for providing such an outstanding resource which has really helped to advance the field of IR. Finally, we would like to thank Chris Buckley for writing and supporting the SMART text retrieval system and making it available for research use. This is a real service to the information retrieval community.

References

Box, G., Hunter, W. & Hunter, J. (1978), *Statistics for Experimenters*, John Wiley and Sons, pp. 118–119.

Buckley, C. (1985), Implementation of the smart information retrieval system, Technical Report 85-686, Cornell University.

Conover, W. (1980), *Practical Nonparametric Statistics*, 2nd edn, John Wiley and Sons.

Croft, W. B. & Xu, J. (1995), Corpus-specific stemming using word form co-occurence, *in* 'Symposium on Document Analysis and Information Retrieval (SDAIR)', University of Nevada, Las Vegas.

Frakes, W. & Baeza-Yates, R., eds (1992), *Information Retrieval: Data Structures and Algorithms*, Prentice-Hall.

Harman, D. (1991), 'How effective is suffixing?', *Journal of the American Society for Information Science* **42**(1), 321–331.

Harman, D., ed. (1993), *Proceedings of the 2nd Text Retrieval Conference (TREC-2).*

Hull, D. (1993), Using statistical testing in the evaluation of retrieval performance, *in* 'Proc. of the 16th ACM/SIGIR Conference', pp. 329–338.

Jing, Y. & Croft, W. B. (1994), An association thesaurus for information retrieval, *in* 'Proc. of Intelligent Multimedia Retrieval Systems and Management Conference (RIAO)', pp. 146–160.

Krovetz, R. (1993), Viewing morphology as an inference process, *in* ‘Proc. of the 16th ACM/SIGIR Conference’, pp. 191–202.

Lennon, M., Pierce, D., Tarry, B. & Willett, P. (1981), ‘An evaluation of some conflation algorithms for information retrieval’, *Journal of Information Science* **3**, 177–183.

Lovins, J. (1968), ‘Development of a stemming algorithm’, *Mechanical Translation and Computational Linguistics* **11**, 22–31.

Neter, J., Wasserman, W. & Kutner, M. (1985), *Applied Linear Statistical Models*, 2nd edn, R.D. Irwin.

Porter, M. (1980), ‘An algorithm for suffix stripping’, *Program* **14**(3), 130–137.

Tague-Sutcliffe, J. & Blustein, J. (1995), A statistical analysis of the TREC-3 data, *in* ‘Proceedings of the 3rd Text Retrieval Conference (TREC-3)’. To appear.

Voorhees, E. M. (1994), Query expansion using lexical-semantic relations, *in* ‘Proc. of the 17th ACM/SIGIR Conference’, pp. 61–69.

Xer (1994), *Xerox Linguistic Database Reference*, english version 1.1.4 edn.