WORD SEGMENTATION BY LETTER SUCCESSOR VARIETIES

MARGARET A. HAIFER and STEPHEN F. WEISS
Department of Computer Science, University of North Carolina, Chapel Hill, North Carolina 27514, U.S.A.

Summary—This paper describes a method for automatically segmenting words into their stems and affixes. The process uses certain statistical properties of a corpus (successor and predecessor letter variety counts) to indicate where words should be divided. Consequently, this process is less reliant on human intervention than are other methods for automated stemming.

The segmentation system is used to construct stem dictionaries for document classification. Information retrieval experiments are then performed using documents and queries so classified. Results show not only that this method is capable of high quality word segmentation, but also that its use in information retrieval produces results that are at least as good as those obtained using the more traditional stemming processes.

1. MOTIVATION AND OBJECTIVES

It has been well established that in document classification for information retrieval, the use of a dictionary of word stems plus some sort of stemming process is far superior to the use of a dictionary of each word type (a word form dictionary). The stem dictionary is smaller, requires less frequent updating and provides better retrieval results than does the word form dictionary [5].

The construction of a stem dictionary and removal of affixes from words has traditionally relied heavily on manual processing. In early experiments both processes were done completely by hand. More recent automatic stemmers typically use a dictionary of word stems and a set of affix removal rules. The rules find and remove all valid affixes and undo minor spelling changes (doubled consonant, dropped final E, Y changed to I, etc.). A word is thus reduced to its stem and is then looked up in the dictionary [2, 3, 5]. While such processes are capable of high quality automatic stemming, they still depend heavily on human effort in the construction of the stem dictionary and the affix rules. In addition, all decisions concerning the stemming process must be made in advance.

It is the goal of this study to develop a word segmentation process that requires minimal human intervention and a priori decisions. The process should allow the text itself to determine the proper word segmentation. Such a system would not only be easier to use but would also be more adaptable to new or modified corpora.

Ideally we would like this system to produce perfect word segmentation, indistinguishable from the best manual results. However, this is neither feasible nor necessary. Since we are using stemming as part of an information retrieval system, our practical goal is to produce a stemming process whose use gives retrieval results competitive with manual stemming.

The background and basic approach used in the stemming process are presented on Section 2. The stemming experiments are discussed in Section 3, and Section 4 presents our results. The stemming experiments are evaluated both in terms of overall stemming quality and information retrieval effectiveness.

2. BASIC APPROACH TO WORD SEGMENTATION

The approach used in this study is based on Z. S. Harris' process for breaking phonetic text into its constituent morphemes [4]. But here we will be segmenting lexical text into stems and
affixes. The process is based on the notion of letter successor variety. We will first define the concept and then show how it is used in word segmentation.

Let \( a \) be a word of length \( n \); \( a_i \) is a length \( i \) prefix of \( a \). We will refer to \( a \) as the test word. Let \( D \) be a corpus of words. \( D_0 \) is defined as the subset of \( D \) containing those words whose first \( i \) letters match \( a_i \) exactly. The successor variety of \( a_i \), denoted \( S_{a_i} \), is then defined as the number of distinct letters that occupy the \( i \) + 1st position in words of \( D_0 \). A test word of length \( n \) has \( n \) successor varieties \( S_{a_0}, \ldots, S_{a_n} \) corresponding to the \( n \) initial substrings of the word.

The following example illustrates the concept. Assume the test word is ABE and the corpus is:

```
ABIDE
ABLE
ABODE
AND
ART
AT
BAT.
```

For each prefix of ABE we can define the set \( D_0 \) and from that, the successor variety. The prefix \( A \) is matched by six words. Of these, there are four distinct second letters. The prefix \( AB \) matches three words and there are three different third letters. The complete word \( ABE \) matches no words and hence has a successor variety of zero. These variety counts are summarized in Table 1.

Motivating the use of successor varieties in word segmentation is the fact that within a word, the \( i \)th letter is dependent to some degree on the \( i - 1 \) letters that precede it. Within a natural word unit (e.g. a stem or affix) this dependence is quite strong and increases with increased \( i \). But if the \( i \)th letter begins a new word unit, the dependence is greatly reduced. Consider, for example, the word ANTIMATTER. The initial letter is clearly unrestricted since it has no predecessors. The second letter is somewhat restricted since it must be compatible with the A. Actually this restriction is vacuous since words beginning with A can have any second letter. The T is more restricted; there are 16 letters that can follow AN (ACDEGHKNOTUVXY). And the I is constrained even further. Only right letters can succeed ANT (AEHILORW). The I completes a word unit that may be followed by any letter. Hence the M is not at all dependent on its four predecessors. The successor varieties of the first four prefixes of ANTIMATTER are summarized in Table 2. While not all words behave so dramatically, the basic principle is sound.

<table>
<thead>
<tr>
<th>PREFIX ( a_i )</th>
<th>( D_0 )</th>
<th>Successor variety ( S_{a_i} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ABIDE</td>
<td>4 (E, N, R, T)</td>
</tr>
<tr>
<td></td>
<td>ABLE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ABODE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AND</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ART</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AT</td>
<td></td>
</tr>
<tr>
<td>AB</td>
<td>ABIDE</td>
<td>3 (I, L, O)</td>
</tr>
<tr>
<td></td>
<td>ABLE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ABODE</td>
<td></td>
</tr>
<tr>
<td>ABE</td>
<td>empty</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 2. Successor varieties for ANTI...

Within word units the successor variety is low and tends to decrease from left to right, while at boundaries the successor variety rises. By calculating the set of successor varieties for a test word and noting the peaks, we can detect these word units and thereby segment the word.

One further piece of information that is useful in this process is to note when a prefix \( a_i \) is found to match a complete word in the corpus. For example, if the test word is HOPEFUL, the prefix \( A \) will match the corpus entry HOPE.

One problem inherent in this process is that near the end of long words, \( S_{a_n} \) becomes very small. Even for a large corpus there are not many words that match a test word for more than a few letters. When this happens the successor variety is too small to provide much information, and hence boundaries near the end of words go undetected. To remedy this Harris suggests repeating the process but with the reverse of the test word and the reverse of the corpus. For each terminal substring of the test word, a predecessor variety count is determined. The set of predecessor counts can be used in conjunction with the successor counts for improved segmentation. As before, we record those terminal substrings of the test word that match complete corpus entries. For example, the five letter suffix of REUNITEN will match the corpus word UNITE. A complete, although somewhat contrived, example of the successor and predecessor counts is shown in Table 3. The asterisk indicates that a test word segment matches a complete word.

A number of techniques are used to make the process of determining the successor and predecessor counts computationally tractable. First, the corpus and reverse corpus are stored as sorted files. The processing of a test word thus needs to access only a single contiguous subset of each file. Second, all counts are developed in parallel so that only one pass through each file is necessary. And third, access to the file is by a table lookup hash on the first two letters of the test word. This keeps searching overhead to an absolute minimum. The resulting process is sufficiently fast and efficient for large scale application.

3. EXPERIMENTAL DESIGN

The process described above provides the raw data for the segmentation experiments. For each test word there is a vector of successor and predecessor variety counts as well as an indication of which substrings of the test word match a complete corpus entry. For storage economy, this indication is implemented by setting the appropriate variety count negative. For example, the four letter prefix of READABLE might have a successor count of \(-15\) indicating that READ can be followed by 15 different letters and that READ appears in the corpus.

After this raw data are obtained, the actual segmentation can be performed. For this study four basic segmentation strategies are used. Each is described in detail below. Segmentation experiments are performed using each strategy by itself as well as in combination with others.

\( (n) \) Cutoff

By far the easiest way to segment a test word is first to pick some cutoff \( K \) and then break the word wherever its successor (or predecessor or both) variety reaches or exceeds \( K \). This was the
affixes. The process is based on the notion of letter successor variety. We will first define the concept and then show how it is used in word segmentation.

Let \( a \) be a word of length \( n \); \( a_0 \) is a length \( i \) prefix of \( a \). We will refer to \( a \) as the test word. Let \( D \) be a corpus of words. \( D_0 \) is defined as the subset of \( D \) containing those words whose first \( i \) letters match \( a_0 \) exactly. The successor variety of \( a_0 \), denoted \( S_{a_0} \), is then defined as the number of distinct letters that occupy the \( i+1 \) position in words of \( D_0 \). A test word of length \( n \) has \( n \) successor varieties \( S_{a_1}, \ldots, S_{a_n} \) corresponding to the \( n \) initial substrings of the word.

The following example illustrates the concept. Assume the test word is ABE and the corpus is:

- ABIDE
- ABLE
- ABODE
- AND
- ART
- AT
- BAT.

For each prefix of ABE we can define the set \( D_{a_0} \) and from that, the successor variety. The prefix \( A \) is matched by six words. Of these, there are four distinct second letters. The prefix AB matches three words and there are three different third letters. The complete word ABE matches no words and hence has a successor variety of zero. These variety counts are summarized in Table 1.

Motivating the use of successor varieties in word segmentation is the fact that within a word, the \( i \)th letter is dependent to some degree on the \( i-1 \) letters that precede it. Within a natural word unit (e.g., a stem or affix) this dependence is quite strong and increases with increased \( i \). But if the \( i \)th letter begins a new word unit, the dependence is greatly reduced. Consider, for example, the word ANTIMATTER. The initial letter is clearly unrestricted since it has no predecessors. The second letter is somewhat restricted since it must be compatible with the A. Actually this restriction is vacuous since words beginning with A can have any second letter. The T is more restricted; there are 16 letters that can follow AN (ACDEGHIKLOSTUVXY). And the I is constrained even further. Only right letters can succeed AN (AEHLORW). The I completes a word unit that may be followed by any letter. Hence the M is not at all dependent on its four predecessors. The successor varieties of the first four prefixes of ANTIMATTER are summarized in Table 2. While not all words behave so dramatically, the basic principle is sound.

Table 2. Successor varieties for ANTI...

<table>
<thead>
<tr>
<th>PREFIX ( a )</th>
<th>Successor variety ( S_{a_0} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ABE, ABE, ABE</td>
</tr>
<tr>
<td></td>
<td>4 (E, N, R, T)</td>
</tr>
<tr>
<td>AB</td>
<td>ABE, ABE, ABE</td>
</tr>
<tr>
<td></td>
<td>3 (L, O)</td>
</tr>
<tr>
<td>ABE</td>
<td>ABE, ABE, ABE</td>
</tr>
<tr>
<td></td>
<td>empty</td>
</tr>
</tbody>
</table>

Within word units the successor variety is low and tends to decrease from left to right, while at boundaries the successor variety rises. By calculating the set of successor varieties for a test word and rotating the peaks, we can detect these word units and thereby segment the word.

One further piece of information that is useful in this process is to note when a prefix \( a_0 \) is found to match a complete word in the corpus. For example, if the test word is HOPEFUL, the prefix A will match the corpus entry HOPE.

One problem inherent in this process is that near the end of long words, \( S_{a_n} \) becomes very small. Even for a large corpus there are not many words that match a test word for more than a few letters. When this happens the successor variety is too small to provide much information, and hence boundaries near the end of words go undetected. To remedy this Harris suggests repeating the process but with the reverse of the test word and the reverse of the corpus. For each terminal substring of the test word, a predecessor variety count is determined. The set of predecessor counts can be used in conjunction with the successor counts for improved segmentation. As before, we record those terminal substrings of the test word that match complete corpus entries. For example, the five letter suffix of REUNITE will match the corpus word UNITE. A complete, although somewhat contrived, example of the successor and predecessor counts is shown in Table 3. The asterisk indicates that a test word segment matches a complete word.

A number of techniques are used to make the process of determining the successor and predecessor counts computationally tractable. First, the corpus and reverse corpus are stored as sorted files. The processing of a test word thus needs to access only a single contiguous subset of each file. Second, all counts are developed in parallel so that only one pass through each file is necessary. And third, access to the file is by a table lookup hash on the first two letters of the test word. This keeps searching overhead to an absolute minimum. The resulting process is sufficiently fast and efficient for large scale application.

3. EXPERIMENTAL DESIGN

The process described above provides the raw data for the segmentation experiments. For each test word there is a vector of successor and predecessor variety counts as well as an indication of which substrings of the test word match a complete corpus entry. For storage economy, this indication is implemented by setting the appropriate variety count negative. For example, the four letter prefix of READABLE might have a successor count of -15 indicating that READ can be followed by 15 different letters and that READ appears in the corpus.

After this raw data are obtained, the actual segmentation can be performed. For this study four basic segmentation strategies are used. Each is described in detail below. Segmentation experiments are performed using each strategy by itself as well as in combination with others.

(a) Cutoff

By far the easiest way to segment a test word is first to pick some cutoff \( K \) and then break the word wherever its successor (or predecessor or both) variety reaches or exceeds \( K \). This was the
match only 19 corpus entries. Of these, ten have A as their i + 1st letter; each of B through J is the i + 1st letter of only one word. While both prefixes have successor counts of ten, the situations are clearly different. For W1 the results seem valid. But for W2, the very low occurrence of matches with B through J might indicate something unusual such as a foreign word or an abbreviation. These exceptional cases can lead to unusually high successor counts and hence to segmentation errors. To avoid this problem we must use a measure that is sensitive both to the variety and distribution of successor letters.

The information theoretic measure of entropy meets this need. For a system consisting of n events, each with probability \( p_i \), the entropy, \( H \), of the system is defined as:

\[
H = -\sum_{i=1}^{n} p_i \log_2 p_i.
\]

\( H \) is greatest when the events are equiprobable. As the distribution of probabilities becomes more and more skewed, the entropy decreases.

Using entropy in the segmentation process allows the importance of each successor letter to be weighted by its probability of occurrence. Rare successor letters will have a smaller effect on the segmentation decisions than will highly probable successors. In this study entropy is implemented in the following way. Let \(|D_{ai}|\) be the number of words in the corpus that match an i letter prefix of the test word \( \alpha \). Let \(|\mathcal{D}_{ai}|\) be the subset of \( D_{ai} \) in which the i + 1st letter is the jth letter of the alphabet. Then the probability that the successor letter to \( \alpha \) is the jth letter of the alphabet is \(|\mathcal{D}_{ai}| / |D_{ai}|\). In the example above, W1 has 10 possible successor letters each with probability 0.1. W2 also has 10 successors; A has probability 0.53 while the rest have probability 0.05. We can now calculate the entropy of the successor system for a test word prefix \( \alpha \), as:

\[
H_{\alpha} = -\sum_{j=1}^{n} \left| \mathcal{D}_{aj} \right| |D_{ai}| \log_2 \left( \left| \mathcal{D}_{aj} \right| / |D_{ai}| \right).
\]

Using this formula, the i letter prefix of W1 has \( H = 3.3 \), while for W2, \( H = 2.5 \). The distinction between W1 and W2 is now clear.

For every test word we will have associated a vector of successor entropies. In an analogous way we can also determine a vector of predecessor entropies. A cut is made in the test word where one or both of the entropies reaches some cutoff. As with frequency counts, the entropy of segments that match complete corpus words is set negative.

Using the four basic segmentation strategies and various combinations, 15 experiments were performed. Not all possible combinations of fundamental strategies are included. Some experiments were eliminated when preliminary results showed them to be unfruitful.

(c) Determination of stems

Once a word has been segmented, it remains to determine which segment is to be considered the stem. In most cases the first segment is taken as the stem. If, however, this initial segment occurs in many different words, it is probably a prefix. The second segment should then be taken as the stem. Experiments showed that initial segments that occur in more than twelve different words are generally prefixes. For some words, such as APEMAN, the first and second segments match corpus words and hence both appear to be stems. In these cases the word is assumed to be compound and both segments are considered stems. Multiple prefixes are very
rare in English and therefore segments beyond the second are not considered as candidates for word stem.

Since stems will be used only for lexical normalization, our definition of stem may be somewhat relaxed from the traditional definition. We define a stem to be the segment of a word that specifies the true stem and distinguishes it from all other stems. For example, COMPUT is an acceptable stem for COMPUTER, COMPUTATION, COMPUTING, etc. All words beginning COMPUT belong to the COMPUTE family so no ambiguity can result. COMP, however, is not an acceptable stem since it cannot distinguish COMPUTER from COMPANY and COMPARE.

(i) Corpora

Three sets of test data are used in these experiments. The first is a subset of the Brown Corpus[7]. Various sized pieces of this large document collection were used to determine the effect of corpus size on segmentation quality. The Brown Corpus was also used to determine workable cutoff values for use in later experiments. To permit information retrieval experiments, two sets of documents and queries were used. One is a collection of 75 documents and 5 queries from the Carolina Population Center (CPC) Library. The other is a set of 82 documents from the 1963 annual meeting of the American Documentation Institute (ADI), and 35 queries prepared by the SMART project at Cornell University. Both document-query collections have predefined relevancy decisions for use in evaluating retrieval effectiveness.

Words of length one or two were removed from all the tests. Such words virtually never require segmentation and only serve to cloud the segmentation issue. For example, inclusion of AN and BE in the corpus can cause BEAN to be split into BE-AN.

As is standard for information retrieval collections, the CPC and ADI collections do not contain function words. For the ADI collection, function word removal was accomplished using a function word list provided by SMART. In the CPC collection, the process was approximated by deleting words of length four or less. Stripped of function words, the CPC collection contains approximately 6200 word types while the ADI contains 7100.

4. RESULTS AND ANALYSIS

(a) Preliminary experiments

Before beginning the actual segmentation experiments some preliminary work was done using the Brown Corpus. The purpose was first, to determine the sensitivity of the segmentation process to collection size, and second, to develop a set of servicable cutoff values for use in later experiments.

To determine the effect of corpus size on successor and predecessor variety counts, various size pieces of the corpus were analyzed. Successor counts stabilized when the corpus reached about 2000 word types. Predecessors did not stabilize until about 5000 entries. Common suffixes such as ED, ING and S can be associated with a large variety of stems. Hence predecessor counts for suffixes are typically higher than successor counts for stems or prefixes. It therefore requires a larger corpus to obtain the full variety of predecessors. Both the ADI and CPC collections exceed the 5000 word minimum and hence it is felt that results obtained with these collections should not be significantly biased by collection size.

These preliminary experiments also showed that a successor cutoff of 5, a predecessor cutoff of 17 and a sum cutoff of 23 produced serviceable results for collections of 5000 words or more. Since letter dependency is virtually nil across word boundaries, it was decided to treat those portions of test words that match a complete corpus entry as having unlimited successor or predecessor variety. Hence a negative variety count will always satisfy cutoff or peak criteria.

(b) Segmentation experiments

This section describes briefly the 15 segmentation experiments performed on the CPC collection. The results of each experiment are compared with the ideal using the measures of precision and recall. Precision indicates the fraction of word cuts that are correct; recall indicates how complete a job is being done.

\[
\text{PRECISION} = \frac{\text{Number of correct cuts made}}{\text{Total number of cuts made}}
\]

\[
\text{RECALL} = \frac{\text{Number of correct cuts made}}{\text{Total number of true boundaries}}
\]

Experimental results are summarized in Table 4.

(i) Successor count reaches cutoff. Preliminary experiments showed that the use of successor frequency with a cutoff was completely unsatisfactory. Words tended to have high successor counts near their beginning but this trailed off almost monotonically as is seen in the example below.

<table>
<thead>
<tr>
<th>Successors</th>
<th>10 18 6 7 2 3 1 1 1 1 1 2</th>
</tr>
</thead>
</table>

A low cutoff will produce disastrously low precision while a high cutoff will produce similarly low recall. Further experimentation with this method was therefore not performed.

(ii) Successor and predecessor counts reach cutoffs. When segmentation required that both the successor and predecessor counts reach cutoffs, more reasonable results were obtained. The precision is quite good (0.894), while the recall is fair (0.511). This process eliminates one of the problems encountered in experiment 1 by preventing high successor counts near the beginning of words from producing cuts. On the other hand, many "almost good enough" values of the successor and predecessor counts are ignored and hence the low recall.

(iii) Successor plus predecessor reach cutoff. Some of the restrictiveness of experiment 2 can be eased by requiring only that the sum of the two counts reach the cutoff. In this way an

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 successor ≥ cutoff</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>2 successor ≥ cutoff and predecessor ≥ cutoff</td>
<td>0.594</td>
<td>0.311</td>
</tr>
<tr>
<td>3 successor + predecessor ≥ cutoff</td>
<td>0.848</td>
<td>0.555</td>
</tr>
<tr>
<td>4 successor neg.</td>
<td>0.904</td>
<td>0.318</td>
</tr>
<tr>
<td>5 predecessor neg.</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6 successor neg. or predecessor ≥ cutoff</td>
<td>0.778</td>
<td>0.713</td>
</tr>
<tr>
<td>7 successor or peak/plateau</td>
<td>0.846</td>
<td>0.714</td>
</tr>
<tr>
<td>8 successor and predecessor at peak/plateau</td>
<td>0.787</td>
<td>0.599</td>
</tr>
<tr>
<td>9 successor + predecessor at peak/plateau</td>
<td>0.941</td>
<td>0.818</td>
</tr>
<tr>
<td>10 successor neg. or predecessor at peak/plateau</td>
<td>0.848</td>
<td>0.917</td>
</tr>
<tr>
<td>11 hybrid of exp. 2 and 5</td>
<td>0.910</td>
<td>0.610</td>
</tr>
<tr>
<td>12 successor neg. or predecessor neg. at peak/plateau</td>
<td>0.720</td>
<td>0.758</td>
</tr>
<tr>
<td>13 sum of entropies ≥ cutoff</td>
<td>0.889</td>
<td>0.596</td>
</tr>
<tr>
<td>14 entropy version of exp. 11</td>
<td>0.874</td>
<td>0.526</td>
</tr>
<tr>
<td>15 relaxation of exp. 14</td>
<td>0.818</td>
<td>0.700</td>
</tr>
</tbody>
</table>
exceptionally high value of one can compensate for a lower value of the other. Using this procedure the number of correct cuts is increased about 10 per cent giving a recall of 0.565. The number of incorrect cuts however, is nearly doubled lowering the precision to 0.848.

(iv) Successor frequency negative. Breaking words after initial substrings that match a complete word yields very accurate results (precision = 0.904). Less than 10 per cent of the cuts are incorrect. These errors are due mostly to words which contain a shorter word other than their stem. For example, ELECTRONIC is broken after ELECT. But this method is very conservative and cannot handle words with minor spelling changes. COMPUTATION and COMPUTING, for example, are not segmented at all. It is therefore not surprising that this method has the lowest recall (0.318) of any experiment.

(v) Predecessor frequency negative. The reverse of experiment 4 is to segment where a predecessor count is negative. This, however, yields no useful results. Cuts are made only where a suffix such as ABLE matches a corpus word. And such occurrences are simply too rare to be of any value.

(vi) Successor frequency negative or predecessor frequency reaches a cutoff. This experiment combines the cutoff and whole word segmentation techniques. Specifically, a cut is made if either the predecessor frequency exceeds a cutoff or the successor frequency is negative. The other two possibilities, that of successor frequency reaching a cutoff and the predecessor frequency being negative, are excluded for reasons discussed in experiments 1 and 5. The results show a moderate value for both precision (0.778) and recall (0.711). The increased recall is due to the fact that the two segmentation criteria are each finding their own set of boundaries. Unfortunately they are also making distinct errors and hence the reduced precision.

(vii) Successor frequency at peak or plateau. The third basic segmentation technique breaks words where frequency counts reach a peak or plateau. In this experiment, only the successor frequency is used. Results show a poor precision (0.486) with a moderately good recall (0.734). Less than half of the cuts made by this process are correct. One reason for this is that peaks often occur near the beginning of words, particularly after the first vowel. Peaks occur, for example after BE, CA, CO, DE and DI. Words such as COUNCIL and DIFFUSION are broken into CO-UNCIL and DI-FUSION. The high recall, on the other hand, indicates that a majority of true segment boundaries are characterized by a successor peak or plateau.

(viii) Successor and predecessor at peak or plateau. In order to counteract some of the tendency of experiment 7 to overcut near the beginning of words, this experiment makes cuts only where both successor and predecessor counts are at a peak or plateau. Individually predecessor and successor counts indicate many false cuts. But it is fairly unlikely that the same false cut will be found by both counts. Thus the precision for this method is improved to 0.787. The recall, however, is reduced to 0.569. This represents an overall improvement over experiment 7 since while the number of correct cuts is reduced by about 25 per cent, the number of errors is reduced by more than 500 per cent.

(ix) Sum of frequency counts at peak or plateau. As in experiment 3, this experiment uses the sum of the predecessor and successor frequency counts. Breaks are made where the sum reaches a peak or plateau. And like experiment 3, recall improves (0.828) at the expense of precision (0.441). The reason for this is that experiment 9 produces about three times as many cuts as experiment 8, and the vast majority (about 80 per cent) of these new cuts are incorrect. The increased number of cuts is due largely to the fact that the sum of the counts has more peaks and plateaus than does either count by itself. In the example below the successor and predecessor vectors each have a single peak (circled) and no plateaus. The sum, however, has a peak and two plateau points.

<table>
<thead>
<tr>
<th>Successor</th>
<th>1 3 4 5</th>
<th>Predecessor</th>
<th>7 6 4 3 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM</td>
<td>8 9 8 5</td>
<td>3 3 3 3 3 3</td>
<td></td>
</tr>
</tbody>
</table>

Occasionally a peak in one of the count vectors will be smoothed out in the sum. But the net effect is a large increase in the number of peaks and plateaus.

A few words are segmented successfully only with this technique. For example, EXPLOR-E where it can be properly equated to EXPLORATION, EXPLORING. Many words, however, are broken improperly. For example, CRUCIAL becomes CRU-CIAL which can then be confused with CRUDE and CRUELTY.

(x) Successor frequency negative or predecessor frequency at peak or plateau. This experiment combines the full word and peak-plateau techniques. Breaks are made where the successor frequency is negative or where the predecessor frequency is at a peak or plateau. The full word criterion finds stems that have not been changed by their affix (e.g. FIND-E). The successor count is used to locate common suffixes, particularly those which have modified the spelling of their stem (COMPUTE + ER = COMPUTER). This results in the highest recall of all experiments (0.937) but poor precision (0.484). Only 7 per cent of the true cuts are missed. Of these, many are double suffixes such as MENTS and ORS which do not have sufficiently high predecessor counts. Some others are words ending in IN or IE. These suffixes are invariably preceded by S or T so predecessor frequency cannot detect the suffix. As in previous experiments, the low precision is due to the increased number of cuts made.

(xi) Hybrid of experiments 2 and 6. This method combines the predecessor and successor in a less restrictive way than previous experiments. A break is made wherever one is strongly indicated by one of the counts and confirmed by a moderate value for the other count. Specifically, a cut is made when the successor is negative (strong indication) and the predecessor count is at least 3 (moderate confirmation), or where the predecessor count is at least 17 (strong) and the successor is at least 2 (moderate). Previous experiments have shown that some sort of combination of the successor and predecessor is necessary to achieve acceptable results. Unfortunately the union of the measures produces too many incorrect cuts, while the intersection of the methods is too restrictive. Experiment 11 is a middle ground and achieves quite good results: precision is 0.910, recall is 0.610. While the recall is not as good as in some other experiments, it is the highest among the high precision experiments.

(xii) Successor entropy negative or predecessor entropy reaches cutoff. The remaining experiments use the entropy measure. In this experiment words are broken where an initial segment matches a whole word or where the predecessor entropy reaches 3. The experiment is similar to experiment 6 and achieves similar results. Precision is 0.720, recall is 0.728.

Preliminary experiments showed that most common suffixes have predecessor entropies greater than three. For example, ES = 3.679, ING = 3.697 and MENT = 3.122. Some less common suffixes have smaller values, but lowering the cutoff would admit too many errors.

(xiii) Sum of entropies greater than cutoff. The experiment is the entropy analog to experiments 3 and 9. The absolute values of the successor and predecessor entropies are added. If either is negative, the sum is set negative. A cut is made where the sum is negative or reaches the cutoff of 4. The results are mediocre: precision is 0.609 and recall is 0.596. Both measures are lower than the results of the previous entropy experiment.

(xiv) Entropy version of experiment 11. This experiment follows the same philosophy as experiment 11. A word is broken where its successor entropy is negative (strong indication) and its predecessor entropy reaches 0.8 (moderate confirmation), or where the predecessor entropy is
negative or reaches 3.0 (strong) and the successor entropy reaches 1.0 (moderate). The results while good (precision = 0.874, recall = 0.526) are not as good as experiment 11 and are quite similar to experiment 2.

(xv) Relaxation of experiment 14. This is an attempt to improve recall by relaxing the rules of experiment 14. Some of the words that were uncut in experiment 14 were found to have some successor entropy values of zero. This indicates that a word segment must be followed by a particular letter. For example, the segment CONSEQ must be followed by the string UEN. Hence the zero entropies in the example below.

\[
\text{CONSEQUENCES} \\
\text{Succ:} 2 \cdot 1 \cdot 2 \cdot 3 \cdot 2 \cdot 7 \cdot 2 \cdot 5 \cdot 1 \cdot 8 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 9 \cdot 0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \\
\text{Pred:} 0 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 0 \cdot 9 \cdot 1 \cdot 8 \cdot 1 \cdot 2 \cdot 1 \cdot 8 \cdot 3 \cdot 7 \cdot 3 \cdot 3
\]

The slight rise in the successor entropy after UEN indicates a desired break but is not large enough to trigger the cutoffs used in experiment 14.

We attempt to solve this problem by softening the cutoffs following word segments with zero successor entropy. A break is made after the 8th letter if the conditions of experiment 14 are met or if the 1-5th successor entropy is zero and the 6th successor or predecessor entropy reaches a small cutoff (0.8). This method will then break CONSEQUENCES after CONSEQUEN allowing it to be associated with CONSEQUENCE and CONSEQUENTLY. As desired, recall is improved over experiment 14 (0.700) but the precision decreases (0.818). Overall this represents an improvement since about two thirds of the new cuts made by experiment 15 are correct.

(xvi) Summary. The results of the segmentation experiments are tabulated in Table 4. Clearly none reaches the ideal, although several experiments, 11 and 15 in particular, are quite good.

(c) Stemming results

For use in information retrieval, a segmentation scheme must not only break words correctly but must also be able to associate words with common stems. To test this, the CPC test text as segmented by each experimental method was fed to the stem determination algorithm described in Section 3.

Each family of words derived from a common stem was determined by hand. This was then compared with the output of the stemming programs. The results are summarized in Table 5. The results show that good quality automated stemming is indeed achievable by the majority count method. For each of the experiments, the quality of the stemming is highly correlated with the segmentation quality. For the best experiments (11 and 15) more than half of the word families have all their members correctly identified. Three quarters of the families have at least a majority of their members identified.

Errors in the stemming process can be divided into two classes. First, a word may simply not be associated with its correct stem. For example, WIVES is not associated with other words from the WIFEs stem. In about 20-30 per cent of the word families a majority of the word members are classified in this way. A far more serious error occurs when a word is associated with a stem to which it does not belong. For example, ELECTRON may be associated with the stem ELECT. This produces false synonymy and can seriously depress information retrieval effectiveness. The worst offenders in this respect are the experiments that have very high recall and poor precision (experiments 7, 9 and 10). This tends to confirm the intuitive notion that when dealing with the precision-recall trade-off, it is better to err in the direction of conservatism (fewer cuts, higher precision) than to make too many cuts.

(d) Method of choice

Performing information retrieval experiments with each of the experimental processes was not feasible. Therefore it was decided to determine the best of the segmentation processes and concentrate retrieval experiments on it. Two experiments, numbers 11 and 15, seem to be the best candidates for this honor. Their differences are first, experiment 11 uses frequency counts directly while 15 requires the added computation involved in determining the probabilities and in calculating the entropies. Second, experiment 11 has higher precision and lower recall than 15. And third, as a result of its larger number of cuts, experiment 15 makes many more incorrect stem assignments (8 vs 1 per cent for experiment 11). After consideration of these factors, method 11 was chosen as the "best". All information retrieval experiments described below use this method.

Table 6 shows a more detailed picture of stemming results for method 11 with both the CPC and ADI collections. The ADI was not used in previous experiments and therefore its results should be free from bias due to prior exposure. The results are quite good; 74 per cent of the words in the CPC collection and 61 per cent of the ADI words are stemmed correctly.

For about a quarter of the words in each collection, the calculated stem is longer than the true stem and contains the true stem as a proper substring. Multiple affixing is the primary cause of these long stems. For example, HOPEFULLY the segmentation process has no problem removing the LY. But the second affix, FULL, is less likely to meet the criteria for segmentation.

Table 6. Results for method 11

<table>
<thead>
<tr>
<th></th>
<th>CPC</th>
<th>ADI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computed stem is correct</td>
<td>74%</td>
<td>61%</td>
</tr>
<tr>
<td>Computed stem is too long</td>
<td>22%</td>
<td>38%</td>
</tr>
<tr>
<td>Computed stem is too short</td>
<td>5%</td>
<td>10%</td>
</tr>
<tr>
<td>Computed stem is wrong</td>
<td>1%</td>
<td>1%</td>
</tr>
</tbody>
</table>
Hence the word will be assigned the stem \textit{HOPEFUL}. Because of its scientific nature, the ADI collection has a higher percentage of multiply affixed words and hence a higher percentage of long stems than does the CPC. Over 60 per cent of the ADI words which are given long stems have more than one affix. About 42 per cent of the long stems given the CPC collection are multiply affixed.

A second cause of long stems is nonstandard spelling changes. It is because of such changes that the following pairs of words are not associated: \textit{COMMUNICATE} and \textit{COMMUNIQUES}, \textit{DESCRIBED} and \textit{DESCRIPTIVELY}, \textit{TRANSMITTED} and \textit{TRANSMISSIONS}. Such words account for another 20 per cent of the ADI's long stems. The remaining long stems share little in common. However, they do tend to have either stems which are not found as complete words or nonstandard affixes.

Assigning a word an overly long stem is not as serious an error as some others. Consider the following set of words:

| CONFORMING | FORMED | INFORMAL |
| CONFORMS | FORMING | INFORMATION |
| FORM | FORMS | INFORMATIONAL |
| FORMAL | FORMULA | INFORMATIVE |
| FORMALIZATION | FORMULAS | INFORMED |
| FORMALLY | FORMULATE | MICROFORMS |
| FORMATTED | FORMULATED | REFORMULATING |
| FORMATING | FORMULATES | TRANSFORMATIONAL |
| FORMATION | FORMULATING | TRANSFORMATIONS |
| FORMATS | FORMULATION |
| FORMATTED | FORMULATIONS |

All are derived from \textit{FORM} and contain this stem as a substring. But only 15 of the 31 words are correctly assigned to \textit{FORM}. Ten are assigned long stems; six are assigned to \textit{FORMULA} and two each to \textit{FORMAT} and \textit{TRANSFORM}. The remaining six are given incorrect stems. Such results are typical of many word classes. Notice first that the long stems cause no confusion among words with different true stems. Second, while all of these words derive from the Latin \textit{FORMA}, it may be argued that there are actually several word classes represented. By this argument the long stems do not seem unreasonable. And third, what would 31 word classes have been normalized to only 10 (assuming the incorrect stems are distinct). Although this is not perfect normalization, it is far better than no normalization at all. Nearly half of the long stems words behave like \textit{TRANSFORM}. That is, two or more words with the same true stem are assigned the same long stem.

A far more serious error is the short stem: the computed stem is a proper substring of the true stem. This can cause ambiguity between words of different parentage. The automatic stemming algorithm produces very few short stems for the CPC collection. In the ADI, however, 10 per cent of the words are assigned to short stems. On examination it was found that a major cause of this problem is the presence in the ADI of short words (four letters or less). The CPC collection contains only words of five letters or more. It was originally felt that the exclusion of short words from the CPC would depress results since some stems (like \textit{ACTING}) would be eliminated while longer inflected forms (\textit{ACTING}) would remain. But such words were found to be very rare. In addition the short words cause many incorrect segmentations. For example, the presence of \textit{RED} and \textit{RING} cause \textit{APPEAL}, \textit{CLEAN} and \textit{COMPASSION}.

short, they may also be confused with the stems of \textit{APPEAL}, \textit{CLEAN} and \textit{COMPASSION}. About 45 per cent of the short stem errors in the ADI would not have been made had short words been eliminated.

Some of the remaining short stems resulted from the incorrect segmentation of very short words. Examples include \textit{DIS-C}, \textit{BL-T}, and \textit{UNI-T}. The solution to much of the short stem problem seems quite simple: remove all short words from the corpus. Such words rarely require segmentation and their removal has an overall positive effect on the results.

There are very few cases where the stem is totally wrong. This does happen when the wrong word segment is chosen to be the stem. Generally this is because a prefix occurs too infrequently to permit it to be identified as such. In \textit{TRANSPLANT}, for example, \textit{TRANS} is taken to be the stem. Sometimes a prefix and a stem are taken to be two stems. For example, \textit{NONPROFIT} is diagnosed to be composed with the stems \textit{NON} and \textit{PROFIT}. This is not so serious since the correct stem is included.

A final source of incorrect stems are those words which appear to be made up of a valid stem and a valid suffix, but which in fact, contain neither. Examples include:

- CAREER (\textit{CARE} + \textit{ER})
- CENTER (\textit{CENT} + \textit{ER})
- CORNER (\textit{CORN} + \textit{ER})
- EARLY (\textit{EAR} + \textit{LY})

The only way to handle such words is to include them on a list of special cases. Fortunately there are very few of these words and the errors they cause are all but insignificant. Incorrect stems from all sources occur in only one word in a hundred.

(e) Information retrieval experiments

The ultimate test of the segmentation process is its performance as an information retrieval aid. To test this, the automatic stemmer was used to construct stem dictionaries for the ADI and CPC collections. Words that were assigned the same stem were given the same concept number. The documents and queries in each collection were then classified using these dictionaries. Finally, retrieval was performed using the SMART information retrieval system [6]. For comparison, retrieval was also performed using several other types of dictionaries. For the CPC collection, we used a word form dictionary (each distinct type is assigned a unique concept number), a traditional stem process (a stem dictionary plus a set of affix rules) and a phrase thesaurus. Retrieval was done with the ADI collection using a word form dictionary, a traditional stem process, SMART's manually defined stem classification, and the SMART thesaurus.

Results are evaluated using the measures of precision and recall. Precision is the fraction of retrieved documents that are relevant to the query. Recall is the fraction of the relevant document set that is retrieved.

The results are summarized in Figs. 1 and 2. They show that the automatic stemmer is clearly superior to the word form dictionaries and just as good as the more traditional stem processes. With the ADI collection, the various stemming results are so similar that for readability, only a containing envelope is shown in Fig. 2. At its widest point, the envelope represents only a 6 per cent precision differential, and at that point the automatic stemmer is 3 per cent better than the SMART stemmer and 6 per cent better than the traditional stemming process. The ADI thesaurus is significantly better than the other dictionaries. This is to be expected since a thesaurus provides semantic as well as lexical normalization.
5. CONCLUSION

In this study we have developed a word segmentation process that does not rely heavily on manual processing or a priori decisions. It allows a corpus to determine the segmentation rules and hence is more adaptable to changes in a collection, new collections or even new languages. Results show that accurate word segmentation is achieved by this process. More importantly, the information retrieval results obtained using the segmentation process are virtually identical to those obtained with the more manually oriented forms of stemming. Systems such as this which take some of the burden off the information specialist will become increasingly important as automated information retrieval gains in usage and scope.

REFERENCES