Application of Natural Language Processing Tools in Stemming

B. P. Pande
Dept. of IT
SSJ Campus, Almora
Kumaun University, India

Prof. H. S. Dhami
HOD, Mathematics
SSJ Campus, Almora
Kumaun University, India

ABSTRACT
In the present work an innovative attempt is being made to develop a novel conflation method that exploits the phonetic quality of words and uses some standard NLP tools like LD (Levenshtein Distance) and LCS (Longest Common Subsequence) for Stemming process.

General Terms
Information Retrieval (IR), Stemming.

Keywords
Phonetic based stem generation system.

1. INTRODUCTION
In linguistic morphology, stemming is the process for reducing inflected (or sometimes derived) words to their stem, base or root form—generally a written word form. One technique for improving IR performance is to provide searchers with ways of finding morphological variants of search terms.

Several stemming algorithms have been suggested in the literature, which can be classified as affix removal methods, statistical methods and lexicon based (or mixed) methods. Lovin’s algorithm [8], Porter’s algorithm [14], Paise/Husk stemmer [13] and Harman’s ‘S’ stemmer [6] are some of the affix removal techniques. These algorithms apply a set of rules, typically known as transformation rules applied to each word. These algorithms are language specific and depend on priory knowledge of language morphology. Statistical algorithms try to cope with this problem by finding distributions of root elements in a corpus but they require rich computing resources as heavy computations are necessary for such approaches. Some highlighted statistical stemming algorithms available in the literature are: Successor Variety [5], Corpus based Stemming [19], N-gram Stemming [9], HMM based stemmer [11], YASS [10]. The third type of stemming utilizes the inclusion of dictionary lookups. The strength of such approaches is in their ability to produce morphologically correct stems but the major and obvious flaw in dictionary-based algorithms is their inability to cope with words, which are not in the lexicon. It is also true that a lexicon must be manually created in advance, which requires significant efforts. Robert Krovetz’s stemmer [7] is one of the example of such approach. Some interesting variations to stemming can be seen in recent years. These are Joshua S. English [4], Eiman Tamah, Al-Shammari [17], J. Snajder et al. [15], Lourdes Araujo et al. [1]. The current method takes into account how the words are being pronounced, thereby utilizing the phonetic quality of words and apply a simple set of rules to achieve the desired stem.

2. PROBLEM STATEMENT
A comprehensive study of the literature has revealed that no stemming algorithm has utilized the science of phonology. By deductive reasoning, we feel that words that share the same morphological invariant or root have similar pronunciation up to some initial point. This intuition motivated us to study and use phonetic algorithms, targeted towards the development of a conflation system.

3. PROPOSED METHOD
For a given word, our idea is to collect words that share same phonetic code first, and then apply some sort of filtering to acquire the most appropriate stem among them. We hypothesize a phonetic based stem generation system.

3.1 Phonetic Algorithm
Phonetic algorithms encode words based on their pronunciation. In his article, Brijesh Shankar Singh [16] has advised that Soundex algorithm [12] (a name matching algorithm based on 6 phonetic sound classifications) could be used for removing affixes. This comment has inspired us to use Metaphone algorithm [2], the advancement of Soundex as a tool to develop a stemming system.

Naushad UzZaman and Mumit Khan [18] have presented an extension of T9 system (Text on 9 keys, a predictive text technology used in cell phones) called T12. In this text input system, the author used the phonetic encoding: the metaphone code or key and collect words that has same metaphone key hits. The initial tasks of our system is to generate the metaphone code first, and then apply some sort of filtering to acquire the most appropriate stem among them. We hypothesize a phonetic based stem generation system.

3.2 Similarity Measures
For filtration, the input word needs to be matched against all words in hand. We shall use two NLP tools for this purpose. Levenshtein distance (LD) or Edit distance (ED), available online at http://www.merriampark.com/ld.htm is a measure of the similarity between two strings. The distance is the number of deletions, insertions, or substitutions required to transform the source string (S) into the target string (T). The Levenshtein distance algorithm has been used in Spell checking, Speech recognition, DNA analysis, and Plagiarism detection.
The longest common subsequence (LCS) [3] is the longest subsequence common to all sequences in two strings. It is a classic computer science problem, the basis of diff (a file comparison program that outputs the differences between two files), and has applications in bioinformatics.

### 3.3 Rule Generation

Having two metrics in hand, we have devised two simple rules to filter out the word list. Suggestion of both the rules has an empirical base. Considering the following words with their EDs and LCSs with the word *Superconductivity*, following table can be generated:

<table>
<thead>
<tr>
<th>Word</th>
<th>ED with word</th>
<th>LCS with word</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supercargo</td>
<td>11</td>
<td>7</td>
</tr>
<tr>
<td>Supercharge</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>Supercoil</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>Superconduct</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>Superconduction</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>Superconductive</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>Superconductivity</td>
<td>0</td>
<td>17</td>
</tr>
<tr>
<td>Superconductor</td>
<td>5</td>
<td>17</td>
</tr>
<tr>
<td>Supercontinent</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>Supercool</td>
<td>10</td>
<td>7</td>
</tr>
</tbody>
</table>

If we draw these points in XY plane, the graphical representation would be:

![Graphical representation](image)

**Figure 1**

The equation of the line thus plotted is

\[
(Y - y_1) = \frac{(y_2 - y_1)}{(x_2 - x_1)}(X - x_1)
\]

\[
(Y - 12) = \frac{(13 - 12)}{(4 - 5)}(X - 5)
\]

\[
(Y - 12) = (-1)(X - 5)
\]

\[X+Y=17\]  \(\ldots\)  \(\text{(3.3.1)}\)

Also, the length or string size of the input word *Superconductivity*,

\[|\text{SUPERCONDUCTIVITY}|=17\]

This makes us to infer that,

\[\text{ED} + \text{LCS} = \text{Input word length}\]  \(\ldots\)  \(\text{(3.3.2)}\)

It is obvious that the points that do not satisfy the equation (3.3.1) shall fall outside the line, so we shall rule out the words for which (3.3.2) does not hold. Again, we can see that still there are some words (supercharge, supercoil and supercool) that need to be excluded. For this, we reason as following:

The pair (ED, LCS) can satisfy the rule (3.3.2) with following three possibilities:

i. \(\text{ED}>\text{LCS}\)

ii. \(\text{ED}=\text{LCS}\)

iii. \(\text{ED}<\text{LCS}\)
The condition (i) implies that the number of operations required to convert string \( A \) to string \( B \) is greater than what is common between them. Consequently there is very low probability that \( A \) is the stem of \( B \).

Condition (ii) shows that there are as many number of operations to convert string \( A \) to string \( B \) as what is common between them. As such a relative higher value of LCS has the same high ED value associated with it. So, we may think of a situation where we have to cover some larger distance from the end point of common sub string to the end of target string to match string \( A \) to string \( B \), no matter how large sub string they have in common.

Condition (iii) implies that the numbers of operations to convert string \( A \) to string \( B \) are less then what they have common. This shows an ideal situation where two strings have relatively higher common sub sequence and there are few operations needed to convert string \( A \) to string \( B \). So, there is high probability that string \( A \) is the stem of \( B \).

On the basis of above justification, the other rule for filtration can be accepted as:

\[ ED < LCS \] 

3.4 Selection of stem

Stem is the valid word which is of shortest length among all of its morphological variants. Thus, we select the word for which ED is maximum and corresponding LCS is minimum from the filtered list of words. If there are more than one word, then we choose the word as stem whose length is minimum.

4. THE ALGORITHM

The process shall be performed in following steps:

1. Accept the input word.
2. Generate its Metaphone Code.
3. Find the list of words that have the same Metaphone code as the input word up to 4 characters.
4. Calculate the Edit Distance (ED) and Longest Common Subsequence (LCS) for each word with respect to the input word.
5. Rule out the words for which following does not hold:
   I. \( ED + LCS = \) Input word length
   II. \( ED < LCS \)
6. Select the word(s) for which ED is maximum and corresponding LCS is minimum.
7. If there are more than one word, then choose the word as stem whose length is minimum.

5. ILLUSTRATIVE EXAMPLES

Example 1:
Suppose we have to find the stem of the word *Superconductivity*.

1. Input word= Superconductivity
2. Metaphone Codes= SPRKNTKFT
3. Words with the code SPRK:
   i. Supercargo (SPRKRK)
   ii. Supercharge (SPRKJ)
   iii. Supercoil (SPRKL)
   iv. Superconduct (SPRKNTKFT)
   v. Superconduct (SPRKNTKXN)
   vi. Superconductive (SPRKNTKFXN)
   vii. Superconductivity (SPRKNKFT)
   viii. Superconductor (SPRKNKFT)
   ix. Supercontinent (SPRKNKNNT)
   x. Supercool (SPRKL)
4. EDs and LCSs for these words are shown in following Table 2.
5. Checking of rule is shown in following Table 2.
Table 2

<table>
<thead>
<tr>
<th>Word</th>
<th>ED with word Superconductivity</th>
<th>LCS with word Superconductivity</th>
<th>Rule Follow?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(I)</td>
<td>(II)</td>
<td></td>
</tr>
<tr>
<td>Supercargo</td>
<td>11</td>
<td>7</td>
<td>N</td>
</tr>
<tr>
<td>Supercharge</td>
<td>11</td>
<td>6</td>
<td>Y</td>
</tr>
<tr>
<td>Supercoil</td>
<td>9</td>
<td>8</td>
<td>Y</td>
</tr>
<tr>
<td>Superconduct</td>
<td>5</td>
<td>12</td>
<td>Y</td>
</tr>
<tr>
<td>Superconduction</td>
<td>4</td>
<td>13</td>
<td>Y</td>
</tr>
<tr>
<td>Superconductive</td>
<td>3</td>
<td>14</td>
<td>Y</td>
</tr>
<tr>
<td>Superconductivity</td>
<td>0</td>
<td>17</td>
<td>Y</td>
</tr>
<tr>
<td>Superconductor</td>
<td>5</td>
<td>12</td>
<td>Y</td>
</tr>
<tr>
<td>Supercontinent</td>
<td>7</td>
<td>11</td>
<td>N</td>
</tr>
<tr>
<td>Supercool</td>
<td>10</td>
<td>7</td>
<td>Y</td>
</tr>
</tbody>
</table>

Since for the words Supercargo and Supercontinent Rule (I) is violated, so we shall drop these words.
Similarly, for the words Supercharge, Supercoil and Supercool Rule (II) has violated, so we shall drop these words too.

6. Out of the remaining words, word(s) for which ED is maximum and LCS is minimum:
   Supercorduct (5, 12)
   Superconductor (5, 12)

7. Since there are more than one word, and length of Superconduct is less than Superconductor, i.e.

\[ |SUPERCONDUCT| < |SUPERCONDUCTOR| \]

So we select Superconduct as the stem of the input word Superconductivity.

Example 2:
Suppose we have to find the stem of the word Motherhood.

1. Input word= Motherhood
2. Metaphone Code= M0RT
3. Words with the code M0RT:
   i. Mother (M0R)
   ii. Mothered (M0RT)
   iii. Motherhead (M0RT)
   iv. Motherhood (M0RT)
   v. Mothery (M0R)
4. EDs and LCSs for these words are shown in following Table 3.
5. Checking of rule is shown in following Table 3.

Table 3

<table>
<thead>
<tr>
<th>Word</th>
<th>ED with word Motherhood</th>
<th>LCS with word Motherhood</th>
<th>Rule Follow?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(I)</td>
<td>(II)</td>
<td></td>
</tr>
<tr>
<td>Mother</td>
<td>4</td>
<td>6</td>
<td>Y</td>
</tr>
<tr>
<td>Mothered</td>
<td>3</td>
<td>7</td>
<td>Y</td>
</tr>
<tr>
<td>Motherhead</td>
<td>2</td>
<td>8</td>
<td>Y</td>
</tr>
<tr>
<td>Motherhood</td>
<td>0</td>
<td>10</td>
<td>Y</td>
</tr>
<tr>
<td>Mothery</td>
<td>4</td>
<td>6</td>
<td>Y</td>
</tr>
</tbody>
</table>
6. Word(s) for which ED is maximum and LCS is minimum:
   Mother (4, 6)
   Mothery (4, 6)
7. Since there are more than one word, and length of Mother is less than Mothery, i.e.
   \[|\text{MOTHER}| < |\text{MOTHERY}|\]
So we select Mother as the stem of the input word Motherhood.

Example 3:

Suppose we have to find the stem of the word Farming.

1. Input word= Farming
2. Metaphone Code= FRMNK
3. Words with the code FRMN:
   i. Farm (FRM)
   ii. Farming (FRMNK)
   iii. Form (FRM)
   iv. From (FRM)
4. EDs and LCSs for these words are shown in following Table 4.
5. Checking of rule is shown in following Table 4.

<table>
<thead>
<tr>
<th>Word</th>
<th>ED with word Farming</th>
<th>LCS with word Farming</th>
<th>Rule Follow?</th>
<th>(I)</th>
<th>(II)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Farm</td>
<td>3</td>
<td>4</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Farming</td>
<td>0</td>
<td>7</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Form</td>
<td>4</td>
<td>3</td>
<td>Y</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>From</td>
<td>5</td>
<td>3</td>
<td>N</td>
<td>N</td>
<td></td>
</tr>
</tbody>
</table>

So we shall drop the words Form and From.

6. Word(s) for which ED is maximum and LCS is minimum:
   Farm (3, 4)
So we select Farm as the stem of the input word Farming.

6. CONCLUSION
We have proposed a novel phonetic based stem generation system that exploits the phonetic quality of words. The present system, based on phonetic quality can handle misspelled input words and produces morphologically correct word as stem. Actual application of the designed algorithm to the testing data gave nearly 100 percent results in terms of producing correct stems. The used phonetic algorithm can be extended to Double Metaphone or Metaphone 3. Such type of research work paves the way for using more than 4 characters of phonetic spelling which in turn can generate better results.

7. REFERENCES


