ISSN:0975-9646

Strength and Accuracy Analysis of Affix Removal Stemming Algorithms

Sandeep R. Sirsat
Department of Computer Science
Shri Shivaji Science & Arts College, Chikhli
Distt: Buldana (MS) 443201

Dr. Vinay Chavan
Department of Computer Science
Porwal College, Kamptee, Nagpur (M. S)

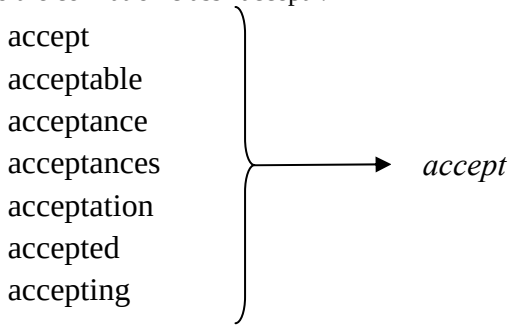
Dr. Hemant S. Mahalle
Department of Computer Science
Phulsing Naik Mahavidyalaya, Pusad
Dist: Yavatmal (M.S.) 445215

Abstract: Many researchers have suggested the techniques for evaluating the strength and similarity of stemming algorithms in different ways. This helps to compute the size of the text corpus reduced by the stemming process, but could not be useful to compute the accuracy of the stemmer. In this paper, we have used one of them to evaluate the strength and suggested one criterion for measuring the strength(WSF) and two criteria for measuring the accuracy(CSF and AWCF) of stemming algorithm. The study evaluates the strength and accuracy of four different affix removal stemming algorithms and found that all the algorithms referred in this paper are strong and heavier, but are less accurate. However, the accuracy of correctly stemmed words and conflating variant words of same group to correct root word is good, but not satisfactory, in Paice/Husk stemmer than the other stemmers.

Keywords – Information Retrieval, inflectional, derivational suffixes, linguistic applications.

I. INTRODUCTION

Stemming is a pre-processing technique that is widely applicable in many text mining applications, such as, Information Retrieval (IR), Topic detection, database search system, and linguistic applications. It is a process of converting the variant words of conflation group to correct root word. For example, the following variant words belong to the conflation class "accept".



A good stemmer will convert all such variant words to the correct root word. Mostly, the stemming algorithms are rule based and use the logical approach for removal or sometimes replacement of inflectional and derivational suffixes. It stems the input word, so that all the variant forms of a word are conflated to the root word. Many rule based algorithms transform these variant forms of word to a stem rather than root word. It helps to reduce the size and complexity of the data in the document collections and in turn it is useful to improve the performance of IR. However, many text mining applications, such as, topic detection, clustering, and document indexing, require much accuracy in index terms rather than index compression.

Thus, it is necessary to evaluate the performance of stemming algorithms on strength and accuracy. The performance of stemming algorithm can be evaluated by using a) Direct assessment method or b) Information Retrieval [9]. There have been many studies of conflation for information retrieval systems as summarized in [6]. Most of these studies have focused on the effect of stemming on retrieval performance measured with recall and precision. A few studies have also looked at stemming as a method for index compression [9]. In this study, we mainly focus on the measuring the correctness of stemming algorithm. That is, how many semantically related words are transformed to a conflation class?

II. AFFIX REMOVAL STEMMING ALGORITHM

The principal of affix removal stemming algorithm is to remove the endings of the word keeping first n letters i.e. to truncate a word up to n^{th} character and remove the rest. Many affix removal stemming algorithms were developed by the researchers. Notable among them are Lovins [1], Porter [2, 4], Paice/Husk[3]. All are aggressive and heavy stemmers; however, they can produce a rather larger number of over steaming errors comparative to the number of under stemming errors.

- **Lovins Stemmer**

The first stemming algorithm was developed by J.B. Lovins that has been influenced by the technical vocabulary [1]. It contains 294 longest endings, 29 conditions and 35 transformation rules. Each ending is associated with one of the conditions. In the first step the longest ending is found which satisfies its associated condition, and is removed. In the second step the 35 rules are applied to transform the ending. The second step is done whether or not an ending is removed in the first step. It is bigger in size than the Porter

algorithm. Although it is faster and aggressive, it works destructively for the short words or words with short stem.

- **Porter Stemmer**

Among all the affix removal stemming algorithms, Porter's algorithm is most popular for stemming English that has repeatedly shown to be empirically very effective [2], and is called as Porter1 algorithm. The original algorithm consists of 5 phases of word reduction. Each phase has a set of rules written beneath each other, among which only one is obeyed. The rules for removing a suffix will be given in the form

$$(condition)\ S1 \rightarrow S2$$

The condition loosely checks the number of syllables to see whether a word is long enough to replace the suffix or not. For example,

$$(m>1)\ EMENT \rightarrow$$

would map *replacement* to *replac*, but not *cement* to *c* [2]. And the condition part may also contain expressions with *and, or, and not,* so that

*(*d and not (*L or *S or *Z))*

tests for a stem ending with a double consonant other than L, S or Z [2].

Dr. Porter himself has suggested several improvements to the original algorithm. It is called "Porter2" algorithm [4]. The suggested changes are

i. Terminating 'y' changed to 'i' seldom occurrence.
ii. Suffix 'us' does not lose its 's'.
iii. Removal of additional suffixes, including suffix 'ly'.
iv. Add step 0 to handle apostrophe (July 2005).
v. A small list of exceptional forms is included (Nov 2006).

Although, these changes do not make the algorithm very extensive, however, failed to improve the performance and minimize the errors to great extend. As shown in table 2, the accuracy of correctly stemmed words increased only from 31.9% to 34.76%

- **Paice/Husk stemmer**

The Paice/Husk stemmer, developed by Chris D. Paice, was first published in 1990 [3]. This stemmer is a conflation based iterative stemmer. It has specified 120 rules given in a rule file, each of which may specify the removal or replacement of an ending. The Paice/Husk stemmer is more aggressive, but has a tendency to over stem. Although it can be easily implemented, but not quite effective, as more numbers of words conflates to incorrect words. A Perl implementation of Paice/Husk stemmer is revised in 2001 by Mary D. Taffet [5] which has been used to obtain the result for comparison.

- **Dawson Stemmer**

The Dawson stemmer, developed by John Dawson, was first presented in 1974. It has much more comprehensive list of about 1200 suffixes. Like Lovins, it is also a single-pass context-sensitive suffix removal stemmer. The main aim of the stemmer was to refine the rule sets and techniques originally proposed in Lovins stemmer and to correct any basic errors that exist. It has two phases. In the first step, all plurals and combinations of the simple suffixes are included, This increases the size of the ending list to approximately five hundred. In the second phase, the Dawson has employ the completion principle in which any suffix contained within the ending list is completed by including all variants, flexions and combinations in the ending list. This increased the ending list once more to approximately one thousand two hundred terms. However, no such record of this list is available.

The Dawson stemmer applies the technique of partial matching which attempts to match stems that are equal within certain limits. This process is not seen as part of the stemming algorithm and therefore must be implemented within the information retrieval system. Dawson warns that without this additional processing many errors would be produced by this stemmer.

Although, Dawson stemmer is fast in execution, but unsuitable from implementation standard due to very complex structure.

III. Errors Occur in Stemming Process:

As evaluated by Chris D. Paice, possibly there are two kinds of errors occur in stemming process [8, 9].

- **Under-stemming** - occurs when two different words of same conflation class should be stemmed to the same root, but aren't.
- **Over-stemming** – occurs when two words of different class should not be stemmed to the same root, but are.

The Chris D. Paice has described method for counting the errors occurred in stemming process and proved that light-stemming reduces the over-stemming errors but increases the under-stemming errors. On the other hand, heavy stemmers reduce the under-stemming errors while increasing the over-stemming errors [8, 9].

IV. Stemmer Strength

The degree to which a stemmer changes words that it stems is called stemmer strength [7]. A 'weak' or 'light' stemmer is one that handles few suffixes and merges only highly related words, for example, singular and plural forms, inflective variant of verbs ("connect", "connects", "connecting", "connected"). A 'strong' or 'Heavy' , on the other hand, handles more suffixes and merges wide variety of forms (…, "connection", "connections", "connectivity", "connectedly", "connectively", etc).

Frakes and Fox have suggested several criteria for measuring the stemmer strength and similarity [7]. The following are the criteria for measuring the stemmer strength [7].

- **The mean number of words per conflation Class** –

This is the average size of the words of conflation group that are transformed to the same stem (regardless of all weather they are all correct) for a corpus. For example if the words "connect", "connected" and "connecting" are stemmed to "connect", then this conflation class size is three. Stronger stemmers will tend to have more words per conflation class.

- **Index compression factor(ICF)** –

The index compression factor is defined as

$$ICF = \frac{n - s}{s}$$

Where n is the number of words in the corpus and s is the number of stems. In other words, the index compression factor is the fractional reduction in index size achieved through stemming. For example, a corpus with 50,000 words (n) and 40,000 stems (s), would have an index compression factor of 20%. Stronger stemmers will tend to have larger index compression factors [7].

- **The number of words and stems that differ** –

Stemmers often leave words unchanged. For example, a stemmer might not alter a root word, such as, "engineer". However stronger stemmers more often change the root words than weaker stemmers, for example, "wander" to "wand", "authority" to "author", etc [7].

- **The mean number of characters removed in forming stems** –

Stronger stemmers remove more characters from words to form stems. For example, a stemmer that stems the corpus {accept, accepted, accepting, accepts} to the stem 'accept' would remove an average of (0+2+3+1)/4 = 1.5 characters. A weakness of this metric is that it does not measure transformations of stem endings. Hence Frake and Fox have developed the following measures [7].

- **The median and mean modified Hamming distance between words and their stems –**

The Hamming distance between two strings of equal length is defined as the number of characters in the two strings that are different at the same position. For strings of unequal length, the difference in length is added to the Hamming distance to give a modified Hamming distance function d. This measure takes into account transformations of stem endings. For example, a stemming algorithm might reduce the corpus {try, tried, trying} to the stem tri. The mean modified Hamming distance between the original words and the stem is (1+2+4)/3 = 2.33 characters and the median is 2.

The above measuring criteria are used to compute the stemmer strength. However, Frakes and Fox have also suggested the similarity metrics for measuring the similarity of stemmer [7]. This similarity metrics is based on the inverse of the modified Hamming distance between stems produced by a pair of stemmers [7] which is given below.

The stemmer similarity metric M for a pair of stemming algorithms A1 and A2, given a wordlist W, is the inverse of the mean modified Hamming distance (d) for all words in the wordlist, M(A1,A2,W) = n/Σ d(xi,yi), for i ranging from 1 to n, where n is the size of W and for all words wi in W, xi is the result of the application of A1 to wi and yi is the is the result of the application of A2 to wi. The inverse of the mean is used so that more similar algorithms will have highervalues of M.

For example, suppose W = {ablution, connected, fairies} and that stemming algorithm A1 produces the stems {ablu, connect, fairy} from W, and stemming algorithm A2 produces {ablute, connected, fair} from W. Then M(A1,A2,W) is computed by dividing the wordlist size (3) by the sum of the modified Hamming distances between the stems produced by each stemmer for each word (for example, the modified Hamming distance between ablu and ablut is 1). The result is 3/(2+2+1) = 0.6 as the measure of the similarity of algorithms A1 and A2.

The criteria suggested by Frakes and Fox are quite effective for measuring the strength of the stemmers and similarity between the stemmer. The study could not focus on evaluating the accuracy of the stemmer, i.e how accurately the stemmer could convert the variant words of conflation class to correct root word. In this study, we have suggested two criteria for measuring the accuracy of the stemmer and one criterion for measuring the strength in the next section.

V. EVALUATING PERFORMANCE OF STEMMER USING DIRECT ASSESSMENT METHOD:

The following measuring criteria are used to evaluate the strength and accuracy of the stemmer.

1) Index Compression Factor(ICF):

The index compression factor represents percent by which a collection of distinct words is reduced by stemming. Higher the number of words stemmed, greater the strength of the stemmer. This is calculated as:

$$\text{ICF} = \frac{(\text{N}-\text{S})}{S} \times 100 \quad (1)$$

Where, N – Number of distinct words before stemming.
S – Number of distinct stems after stemming.

2) Word Stemmed Factor (WSF):

It is the percentage of words that have been stemmed by the stemming process out of the total words in a sample. Larger the number of words stemmed, greater the strength of the stemmer. Minimum threshold for this factor should be 50%.

$$WSF = \frac{WS}{TW} \times 100 \quad (2)$$

Where, WS – Number of words stemmed.
TW – Number of words in a sample.

3) Correctly Stemmed words Factor(CSWF):

It is the percentage of words that have been stemmed correctly out of the number of words stemmed. Higher the percentage of this factor, higher will be the accuracy of the stemmer. Minimum threshold for this factor should be 50%.

$$CSWF = \frac{CSW}{WS} \times 100 \quad (3)$$

Where, CSW – Number of correctly stemmed words.
WS – Total number of words stemmed.

4) Average Words Conflation Factor (AWCF):

This indicates the average number of variant words of different conflation group that are stemmed correctly to the root words. To calculate AWCF, we first compute the number of distinct words after conflation as:

$$\text{NWC} = \text{S} - \text{CW}$$

Where, CW – Number of correct words not stemmed.
Thus, Word Conflation Factor is obtained as:

$$AWCF = \frac{\text{CSW} - \text{NWC}}{CSW} \times 100 \quad (4)$$

Higher the percentage of AWCF, higher will be the accuracy of the stemmer.

For example, if the corpus have variant words of conflation group, such as, "accepts", accepted", "accepting", "acceptance", "acceptable", "acceptances", acceptation", and the stemming algorithm transformed 5 words among

them correctly to the root word, either changing others to improper stem, or living unchanged. Then the word conflation factor for this class is 5/7*100 ie 71%. Thus the average words conflation factor (AWCF) gives the average percentage of variant words of different conflation groups that are transformed to the correct root word.

Analysis of stemmers (Complete Corpus)	**Lovins Stemmer**	**Porter1 Stemmer**	**Porter2 Stemmer**	**Paice-Husk Stemmer**
No. of Unique words before stemming	29417	29417	29417	29417
No. of Unique Stem after stemming	15213	16944	16944	15050
Index Compression Factor	48.29	42.4	42.4	48.83

Table 1 - Computing Index Compression Factor for complete corpus.

Analysis of stemmers (A-alphabet words)	**Lovins Stemmer**	**Porter1 Stemmer**	**Porter2 Stemmer**	**Paice/Husk Stemmer**
Total Words (TW)	1858	1858	1858	1858
Number of Distinct words before stemming (N)	1571	1571	1571	1571
Number of Distinct words after stemming (S)	683	756	727	558
Index Compression Factor (ICF)	**56.52**	**51.88**	**53.72**	**64.63**
Number of words Stemmed(WS)	1363	1248	1237	1319
Words Stemmed Factor (WSF)	**73.35**	**67.17**	**66.58**	**70.99**
Correctly Stemmed words (CSW)	379	399	430	379
Incorrectly stemmed words (ISW)	984	849	807	940
Correctly Stemmed words Factor (CSF)	**27.80**	**31.97**	**34.76**	**28.73**
Correct Words not stemmed (CW)	210	323	334	252
No. of Distinct words after conflation (NWC)	473	433	393	306
Average Words conflation Factor	**-24.8**	**-8.52**	**8.6**	**19.26**

Table 2 – Result obtained by different stemmers applied to 'a'-alphabet words

.

V. RESULT ANALYSIS

To evaluate the performance of the stemmer described in this paper, we have applied these algorithms to the sample vocabulary downloaded from the web site [10]. It contains 29714 distinct words, arranged into "conflation groups". Some of them are incorrect words. For example, there are **287 incorrect words** in the sample of 1858 words which begin with alphabet 'a'. **Table 1** shows the index compressions factor applied to all vocabulary of 29714 words. To measure the strength and accuracy of stemmer, we considered a sample of 1858 words containing 'a' alphabet words and analyze the result using the measuring criteria specified in section V. The result of the most noticeable aggressive stemmers referred in this paper is shown in **Table 2**.

From T**able 1,** it is observed that the index compression factor (ICF) obtain by Porter1 and Porter2 stemmer is comparatively less [42.4%] than ICF of Paice/husk stemmer [48.83%] and Lovins stemmer [48.29%]. Further the word stemmed factor (WSF) obtained by all the algorithms are above 65% which is above the threshold value. This shows that the strength of all the stemmers is strong and all are aggressive in nature. But Lovins stemmer is more aggressive than Paice/Husk which in turn more aggressive than Porter stemmer.

However, there is comparatively large difference between Lovins stemmer and the other stemmers on AWCF, but not differs much on CSWF as shown in table 2. Paice/Husk stemmer obtains 28.73% CSWF and 19.26% AWCF, Porter2 stemmer Obtains 34.76% CSWF and 8.6% AWCF, whereas Lovins stemmer obtains 27.80% CSWF and -24.80% AWCF. Thus the accuracy of correctly stemmed words and conflating variant words of same group to correct stem is good, but not satisfactory, in paice/Husk stemmer than the earlier stemmers.

Further, the following things have been observed:

The word stemmed factor (WSF) obtained by Lovins stemmer is comparatively high (73.35%), but CSF (27.80%) and AWCF (-24.8%) is comparatively low. This is because, besides the words having inflectional and derivational suffixes, it also transforms the root words to incorrect stem. This results in occurrence of both over-stemming and under-stemming errors. However, occurrence of over-stemming errors is more than under-stemming errors. The occurrence of under-stemming errors is high in Lovins stemmer as compare to other stemmers, as NWC is high. This occurs because, the variant words of same conflation class are transformed to different stems, such as,

agree → agre
agreed → agreed
agreement → agre
agreeing → agree

The Porter stemmers mainly focused on truncating the variant words of conflation group to same stem, as it could not consider the morphological/Linguistic rules in forming the condition.

The AWCF obtained by Porter1 stemmer is also negative as compare to porter2 stemmer. This is because more over-stemming errors occurs, as "wander" get changed to "wand", "ached" get changed to "ach", etc.

The performance of Paice/Husk stemmer is slightly better than other stemmers on ICF, and WSF, AWCF. The AWCF is more in Paice/Husk (19.26%) than the Porter2 (8.6%), whereas CSF is low (28.73%) than the porter2 (34.76%). The reason behind this is that, Paice/Husk stemmer is more aggressive than porter2 as WSF obtained is higher i.e. (70.99%) than the WSF obtained by porter2 (66.58%) and the number of distinct words after conflation (NWC) is great in Paice/Husk stemmer than the Porter2 stemmer. Thus the number of occurrence of over-stemming and under-stemming errors is less as compare to Porter2 and other stemmers.

VI. CONCLUSION

Thus, it has been conclude that all the stemming algorithms discussed in this paper are comparatively strong and aggressive, but are less accurate. All tends to produce both over-stemming and under stemming errors. However, the occurrence of under-stemming errors in Paice/Husk stemmer is comparatively low. The ACWF obtained by Lovins and Porter1 stemmer shows negative percentage. This is because the number of words that stems to incorrect words is more than the correctly stemmed words. Thus in both the cases over-stemming and under-stemming errors occurred more than the others. Further the AWCF of Paice/husk stemmer is comparatively positive; still it has the problem of occurrence of over-stemming errors, as the ICF and WSF is comparatively high. The CSF and AWCF is obtained by Porter2 stemmer is quite good, but it produces the over-stemming errors as compare to under-stemming errors.

REFERENCES

1] Lovins J. B. 1968: “Development of a stemming algorithm”. Mechanical Translation and computational Linguistics 11(1-2), 22-31, 1968.

2] Porter M. F. 1980: “An algorithm for Suffix Stripping”, Program\14\ no. 3, pp 130-137, July 1980.

3] Paice C. , Husk, G. “Another Stemmer”, ACM SIGIR Forum, 24(3), 56-61, 1990.

4] Porter M. F. revised in Nov. 2006, available at http://snowball.artarus.org/algorithms/english/ stemmer.html

5] Mary D. Taffet (mdtaffet@syr.edu), Syracuse University- Perl Implementation of Paice/Husk stemmer Revisions: 08/23/2001 – available at http://www.comp.lancs.ac.uk/computing/research/stemming/Links/implementations.htm

6] Frakes, W. "Stemming Algorithms." In Frakes, W. and R. Baeza-Yates, (ed.) Information Retrieval: Data Structures and Algorithms. Englewood Cliffs, NJ: Prentice-Hall, 1992.

7] Frakes W. B., Fox C. J. “Strength and similarity of affix removal stemming algorithms”. ACM SIGIR Forum, Volume 37, No. 1. 2003, 26-30.

8] Paice Chris D. 'Method for Evaluation of Stemming Algorithms Based on Error Counting' JASIS 47(8), August 1996, 632-649.

9] Paice Chris D. “An evaluation method for stemming algorithms”. Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval. 1994, 42-50.

10] Sample English vocabulary available at http://snowball.tartarus.org/algorithms/english /voc.txt

11] Dawson J.L., 1974: "Suffix removal for word conflation," *Bulletin of the Association for Literary & Linguistic Computing*, **2** (3), 33-46.