ISSN 2348 – 7968

An Approach for Effective Text Pre-Processing Using Improved Porters Stemming Algorithm

(1)Mrs. R. Jayanthi (2) Ms. C. Jeevitha

(1)Mrs .R. Jayanthi.,Assistant Professor,Pg & Research Department Of Computer Science, Quaid-e-millath govt. College for women.
(2)Ms. C.Jeevitha, MPHIL Research Scholar, Pg & Research Department Of Computer Science, Quaid-e-Millath Govt. College for Women.

Abstract

In real world an increased amount of user need to access relevant information across multiple documents. Pre-processing is the initial process in Text Mining. Stemming is a heuristic process of removing suffixes and sometimes prefixes from words to arrive at the base word. Stemming algorithms are used to transform the words in texts into their grammatical root form. There are several techniques available in stemming. The most widely used and accepted stemming algorithm by the user is porter stemming algorithm. There are many approaches were developed to improve its structure. This paper reveals the affix stripping process and the inaccuracies encountered during prefix stripping in stemming process that proposes the corresponding solutions and also deals with the enhancement of text pre-processing using improved porters algorithm by taking prefixes into an account.

Keywords: Conflation, Information retrieval, Pre-processing, Stemming, Porter stemming.

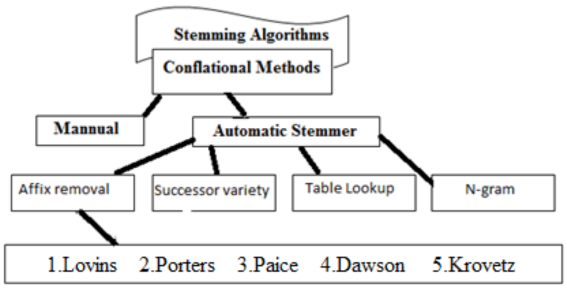
1. Introduction

Stemming is an important step in many of the Information Retrieval (IR) and Natural Language Processing (NLP) tasks. Stemming is usually done by removing any attached suffixes and prefixes (affixes) from actual word to get the root term [1]. It is an effective pre-processing step in Text Mining applications and basic requirement for many areas such as computational linguistics and information retrieval work for improving their recall performance. The stemming filter applies the stemming function to the terms it indexes, and to words in user queries. So in theory all variations of a root word (“render”, “rendered”, “renders”, “rendering”, etc.) are reduced to a single term “render” in the index, saving space. There are two issues in stemming that is over stemming and under stemming. When a term is over stemmed too much of the stem is removed. Over stemming may cause unrelated terms to be conflated. Under stemming is removal of too little of a term and will make the related terms from being conflated [2]. The effectiveness of searching in terms of recall, would be expected to increase if it were possible to *conflate* (i.e., to bring together) the variants of a given word so that they could all be retrieved in response to a query that specified just a single root variant [3]. It is in this light that, a number of so-called stemming Algorithms, or stemmers, have been developed, which attempt to reduce a word to its stem or root form. In a way, the key terms of a query or document are represented by stems rather than by the original words. It implies that different variants of a term can be conflated to a single representative form; it also reduces the size of the dictionary, that is, the number of distinct terms needed for representing a set of documents. A smaller dictionary size results in a saving of storage space and processing time. It used to improve retrieval effectiveness and to reduce the size of indexing files. There is a criterion for judging stemmers that the factors are Correctness, Retrieval effectiveness, Compression performance. Correctness has a concept called over stemming and under stemming. Retrieval effectiveness measured with recall and precision, and on their speed, size and so on.

This paper concentrates only on the improvement of the effectiveness of porters algorithm. A fully automated alternative to truncation is provided by a *stemming algorithm[4]*. This reduces all words with the same root to a single form, the *stem*, by stripping the root of its derivational and inflectional affixes; in most cases, only suffixes that have been added to the right-hand end of the root are removed and this approach to conflation forms the basis of this paper. The removals of prefixes (i.e., strings that have been added at the left-hand end of a root) have been much less studied in the case of English-language retrieval.

2. Stemming Algorithms

There are many stemming algorithms are used in practice for searching and indexing as a theme for Information retrieval system and some other applications.



Types of stemming algorithm

2.1 Manual Stemmer

Manual training of the algorithm is overly time intensive and the ratio between the efforts and the increase in accuracy is marginal at best. It is quite tough to attain at real time system for attaining accuracy. These types of stemmer are consumes more time to implement the idea for efficient and effective retrieval system.

2.2 Affix Stemmers

In linguistic, the term affix refers to either a prefix or suffix. In addition to dealing with suffixes, several approaches also attempt to remove common prefixes. For example, given the word indefinitely, identify that the leading “in” is a prefix that can be removed . Many of the same approaches mentioned earlier apply, but go by the name affix stripping. There are four automatic approaches. Affix removal algorithms removes affixes or prefixes from terms leaving a stem. Successor variety stemmers use the frequencies of letter sequences in the text as the basis for stemming .N-gram method conflates the terms based on the number of digrams or n-grams they share .Correctness, retrieval effectiveness and compression performance judges the stemmers.

Three modes in stemming algorithm:

1. Truncating Method: This method concentrates on removing the prefix and suffix of a given word.

These truncating methods contain the suffix removal algorithms like Lovins, Porters, paice/Husk and Dawson stemmer. Lovins stemmer is a context sensitive and single pass stemmer, which removes endings based on the longest-match principle. This algorithm consists of two steps: elimination of the suffixes and management of the remaining stem [5]. Porters algorithm is one of the most commonly used algorithm which attain an efficient result for English language[6]. Paice/Husk stemmer is a conflation based iterative stemmer. When operating with its standard rule-set, it is a rather 'strong' or 'heavy' stemmer. It removes the endings from a word in an indefinite number of steps. The Stemmer uses a separate rule file, which is first read into an array or list [7]. Dawson proposes an adaptation of the Lovins stemmer based on two modifications. Firstly, the recoding phase is omitted and only the partial-matching routine is used to conflate words. Secondly, to fill the gap left by the recoding phase, he greatly increases the list of common suffixes up to 1200 to handle a lot more situations that were absent in the Lovins stemmer and were many times solved in the recording phase [8].

2. Statistical Method: This type of stemmer is based on statistical analysis and techniques to remove the affixes but after implementing some statistical procedure.

The n-gram method since trigram or n-grams could be used. In this method association measures are calculated between the pairs of terms based on shared unique diagrams[9]. HMM Stemmer are finite-state automata where transition between states are ruled by probability functions [10].

3. Mixed Algorithms: It concentrates on both inflectional and derivational morphology analysis in context and content sensitive corpus.

Advantages of stemming algorithm

- It allows the user to find documents without worrying about word forms.
- It reduces the size of the index, since it reduces the number of separate terms indexed by “collapsing” multiple word forms into a single base word
- .It’s faster than using variations.

Disadvantages of stemming algorithm

- The stemming algorithm can sometimes incorrectly conflate words or change the meaning of a word by removing suffixes.
- The stemmed forms are often not proper words, so the terms in the field are not useful for things like creating a spelling dictionary.
- The index time stemming is that information about the full term will be lost or additional storage will be required to storage both the stemmed or unstemmed word.
- The longest-match principle method is requires generating all possible combinations of affixes and the amount of storage space the endings require.

3. Existing Porters Stemming Algorithm

This Stemmer is a linear step Stemmer. Specifically it has five steps applying rules within each step. Within each step, if a suffix rule matched to a word, then the conditions attached to that rule are tested on what would be the resulting stem, if that suffix was removed, in the way defined by the rule[10]. For example such a condition may be, the number of vowel characters, which are followed be a consonant character in the stem (Measure), must be greater than one for the rule to be applied[11]. Once a Rule passes its conditions and is accepted the rule fires and the suffix is removed and control moves to the next step. If the rule is not accepted then the next rule in the step is tested, until either a rule from that step fires and control passes to the next step or there are no more rules in that step whence control moves to the next step. This process continues for all five steps, the resultant stem being returned by the Stemmer after control has been passed from step five[13,14].

3.1 Algorithm for porters

Step 1: Gets rid of plurals and –ed or –ing suffixes
Step 2: Turns terminal ‘y’ to ‘i’ when there is another vowel in the stem.
Step 3: double suffixes to single ones: -ization,-ational,etc.
Step 4: Deals with suffixes –full,-ness etc.
step5: Takes off –ant,-ence,etc and Removes a final –e.

The general rules are of the form:
(condition) S1 -> S2 Where S1 and S2 are suffixes.

4. Proposed methodology for effective stemming

This proposed methodology has works consciously about the removal of pre-processing steps that is tokenization, removal of digits, punctuations and stop word removal before entering into a stemming process. Further it concentrates on stripping process from both right and the left end of a single word that is both the prefix and suffix of a given word encountered into this stemming process. Finally root word is obtained as end of the stemming process and it also segregating the root word is a legitimate real word or not by ensuring with POS tagger which taken noun, verb and adjective as an account for classification of real word terms. There a possible way for identifying the unreal word by using this POS tagger is explained in Figure 2. In order to profiling the most relevant word as a suggestion for unreal word can be achieved by calculating Euclidean Distance Measures.

4.1 Pre-processing steps

Pre-Processing is one of the important process which removes noisy data to make it as a structural information. Some of the pre-processing steps are involved in this paper like stop-word removal, Punctuation and digits removal, since the domain has taken from English data set naturally these text contains sufficient number of unmeaning list. So before performing a stemmer it should get involved into cleaning process for attaining good result.

4.1.1 Tokenization

Tokenization is the process of breaking the sentences as well as the text file into word delimited by white space or tab or new line etc. Outcome of this tokenization phase is a set of word delimited by new line.

4.1.2 Punctuation Removal

In general a document may contain lot of punctuations in the text which never heals any meaning. So better it can remove for attaining best result.

4.1.3 Stop-Word and single letter word Removal

The word which fails to reveal the meaning when it is alone such a word are termed as stop-words. There are many followings comes under stop-word category such as Prepositions, conjunctions, articles, auxiliary verbs, relative pronouns and so on. These types words have extremely high term frequency in a corpus are known as Stop-words. As a step of stemming the stop-words need to be removed before further processing. So the Single-Letter-Words are removed in this phase.

4.1.4 Stemming has achieved by affix removal

After processing all the above steps, words (tokens) are ready for stemming. The stemming system was developed using lexical look up based approach using three different lexicons suffix, prefix and root. The system mainly works in two steps and it is shown in figure 3.Firstly the input word is queried in the root lexicon; if it is found, then it is considered as a root word e.g. “disagreement”. Secondly if it fails then the system queried into prefix and suffix lexicon for affixes, if it is present then its rule number is retrieved and do affix striping according to rule which were written in the lexicon against retrieving rule number e.g. in the word ” **disagreement**”, one prefix “dis” and one suffix “ent” is present. If the root word is found after stripping suffix/ prefix or both then the system will store the root word. Table 1 shows the affix removal process for improved porters algorithm.

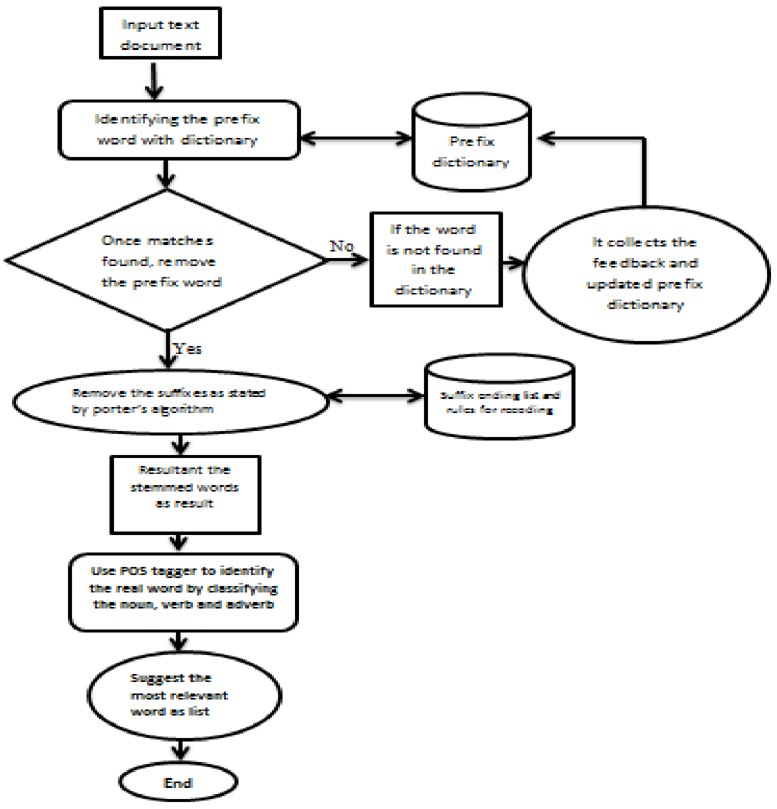

Input text document
Identifying the prefix word with dictionary
Prefix dictionary
Once matches found, remove the prefix word
No
If the word is not found in the dictionary
It collects the feedback and updated prefix dictionary
Yes
Remove the suffixes as stated by porter's algorithm
Suffix ending list and rules for recoding
Resultant the stemmed words as result
Use POS tagger to identify the real word by classifying the noun, verb and adverb
Suggest the most relevant word as list
End

Figure 2: Architectural view of proposed work

4.2 Proposed Algorithm:

Step 1: Text document is given as input to the query string.

Step 2: Removal of punctuation, digits and stop word like Preposition, conjunction, article, ect.,

Step 3: Identify and removing the prefixes by comparing with predefined prefix dictionary lookup.

Step 4: Once matches found, the prefix word is fired.

Step 5: Apply the suffix removing rules as stated by porters suffix list.

Step 6: Remaining root word is consider as a stemmed words.

Step 7: Apply POS tagger for identifying the legitimate meaningful word.

Step 8: Display the most relevant words as profiling suggestion list for unreal words.

Step 9: It also get the feedback from the user for additional prefixes updated with the dictionary and go to step 2 until the prefix word to be stemmed.

Table 1: Affix removal process

word	Length of original word	Prefix	suffix	Result	Space occupied for result in memory
unauthorized	12	Un	iz,ed	author	6
disqualified	12	Dis	Ed	qualify	7
uncomfortable	13	Un	able	comfort	7
Unhelpful	9	Un	Ful	help	4
Dangerous	9	---	Ous	danger	6
Antibody	8	Anti	---	body	4
uninteresting	13	Un	Ing	interest	8

4.3 Implementing prefixes

Here prefix are implemented in this algorithm which contain prefix set dictionary for word look-up. There are many prefixes are available in English language.

Some of the Prefixes
Anti,bi,co,contra,counter,de,di,dis,en,extra,in,inter,intra,micro,mid,poly,post,pre,re,semi,super,supra,sur,trans,mini, multi,non,over,para,tri,ultra,un,kilo.

Table 1: Example for prefix word

Dis	Im	Un	Pre	Mis
Disagree	Impolite	Unable	prearrange	Misbehave
Disarm	Impossible	unbroken	Pre-test	Mislead
Disconnect	impure	unknown	preschool	Misplace
Disloyal	Imperfect	uncertain	prepaid	Misread

In this prefixes certain words change the meaning from positive to negative.

4.3.1 Totally positive to negative (Negative prefixes)

Class1: a, anti, dis, in, un, non, il, ir,..

For prefix **‘a’**, there should be followed by consonant as a beginning word.

Political	Apolitical

Sexual	Asexual
Theist	Atheist

For prefix **'Dis'**, there should be followed by consonant or vowel as a beginning word.

Agree	Disagree
Comfort	Discomfort
Orient	disorient

For prefix **'il'**, there should be followed by consonant or vowel as a beginning word.

Legal	Illegal
Legible	Illegible
Literate	Illiterate

4.3.2 Partially converting meaning dimensionality:

Class 2: Stating the time Eg: pre, post, bi.

Class 3: Some word just remove the prefix and reveals meaningful word.

Class 4: There is a word which gives the result as unreal word, so it changes to their synonyms with limited length as consider for handling the space in storage. Eg: Kilo, contra, counter, de, en, extra, inter, intra, etc.,

Table 2: view of different word dimensionality

Word	After prefix	Suggestion for antonyms
Dislike	Like	Hate, abhor, detest, loathe..
Incorrect	Correct	Wrong, bad, unfair, amiss, astray..
Antioxide	Oxide	For Noun no changes

4.4 Advantages of this prefix work:

1. It increases the searching speed when it is applied to searching as an application.
2. Prefix search can increase the recall.
3. This algorithm produced an effective result for medical term data set.

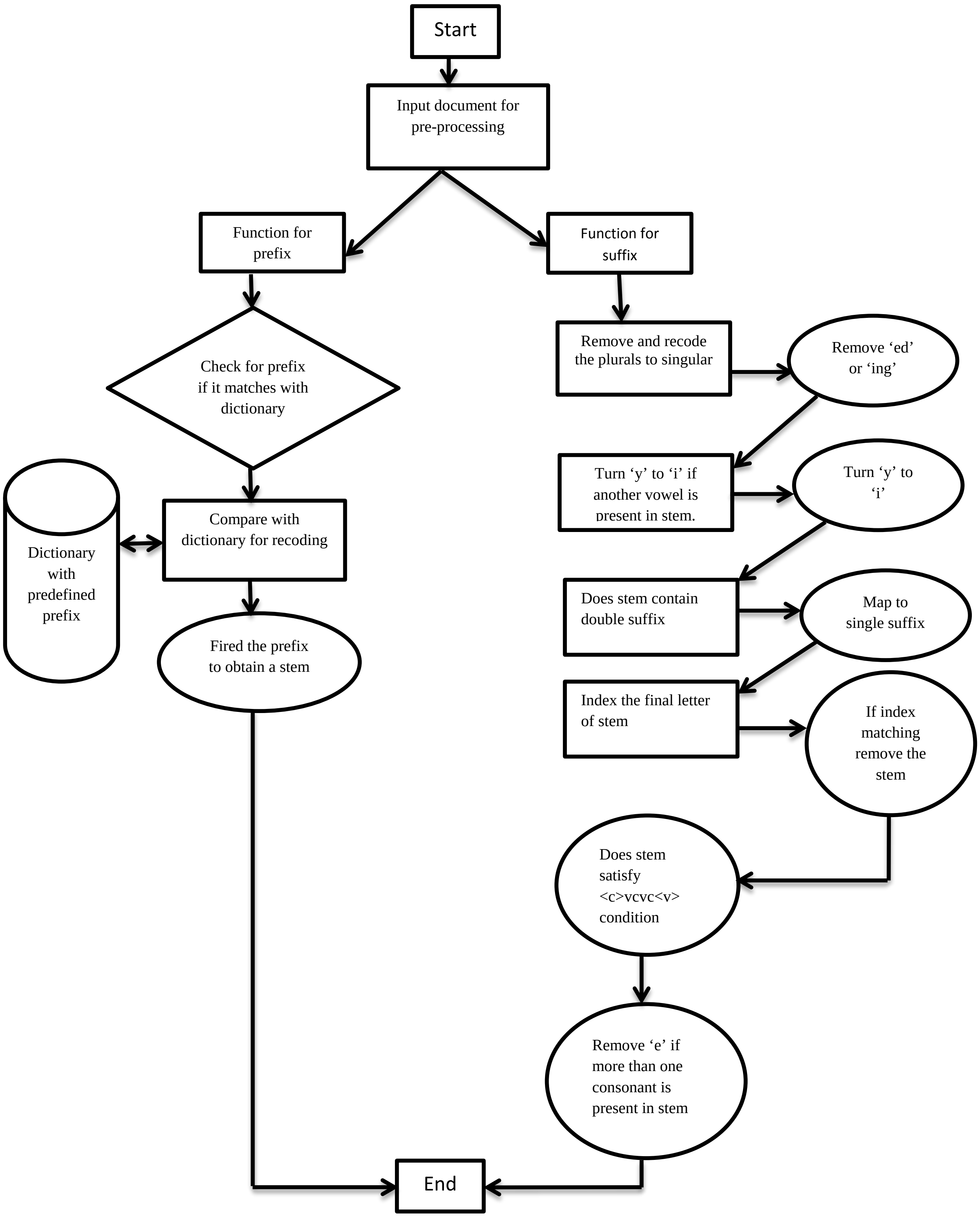

Start
Input document for pre-processing
Function for prefix
Function for suffix
Check for prefix if it matches with dictionary
Dictionary with predefined prefix
Compare with dictionary for recoding
Fired the prefix to obtain a stem
Remove and recode the plurals to singular
Remove 'ed' or 'ing'
Turn 'y' to 'i' if another vowel is present in stem.
Turn 'y' to 'i'
Does stem contain double suffix
Map to single suffix
Index the final letter of stem
If index matching remove the stem
Does stem satisfy <c>vcvc<v> condition
Remove 'e' if more than one consonant is present in stem
End

Figure 3: Affix Stripping process for Improved Porters algorithm

4.5 Conclusion and Future work

The prefix removal approach along with suffix removal in porters stemming algorithm contributes to reduce the space occupied by the word in the memory of the computer especially there are many words that differ beginning but similar roots. The primary goal of this algorithm achieving the beneficial result for information retrieval systems. The ultimate aim of this prefixes has combined with the suffix stemming algorithm for reducing the space dimensionality and speed up indexing and search, especially when combined with Information Retrieval System. Though it reduces the space in memory, it fails to concentrate the lexicon study of the word with changing their original meaning dimensionality. This paper suggests that antonyms profiling words may also help to match the original meaning with minimum length as a concern.

REFERENCES

[1] A Comparative Study of Stemming Algorithms .Ms. Anjali Ganesh Jivani et al, Int. J. Comp. Tech. Appl., Vol 2 (6), 1930-1938. IJCTA | NOV-DEC 2011

[2] chapter 8: stemming algorithms: W. B. Frakes Software Engineering Guild, Sterling, VA 22170

[3] Frakes William B. “Strength and similarity of affix removal stemming algorithms”. ACM SIGIR Forum, Volume 37, No. 1. 2003, 26-30.

[4] Willett,P.(2006) The Porter Stemming algorithm: then and now. Program: Electronic Library and Information Systems,40(3).pp.219-223. ISSN 0033-0337.

[5] CHAPTER 8: STEMMING ALGORITHMS: W. B. Frakes Software Engineering Guild, Sterling, VA 22170

[6] Frakes William B. “Strength and similarity of affix removal stemming algorithms”. ACM SIGIR Forum, Volume 37, No. 1. 2003, 26-30.

[7] J. B. Lovins, “Development of a stemming algorithm,” Mechanical Translation and Computer Linguistic., vol.11, no.1/2, pp. 22-31, 1968.

[8] Paice Chris D. “Another stemmer”. ACM SIGIR Forum, Volume 24, No. 3. 1990, 56-61.

[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] Porter M.F. “An algorithm for suffix stripping”. Program. 1980; 14, 130-137.

[11] Porter M.F. “Snowball: A language for stemming algorithms”. 2001.

[12] Krovetz Robert. “Viewing morphology as an inference process”. Proceedings of the 16th annual international ACM SIGIR conference on Researchanddevelopmentininformationretrieval.1993,191-2

[13] Porter M.F. “An algorithm for suffix stripping”. Program. 1980; 14, 130-137.

[14] Porter, M.F., (2002) “Developing the English Stemmer”, http://snowball.tartarus.org/.

[15] “Design algorithm for prefix stemming english words language”, Amin mubark Alamin Ibrahim *et al., IJITR, volume no.2.

[16] “Affix removal stemmer for natural language text in nepali”, Abjijit paul,Arndam Dey, Bipul sugan Purkayastha, International Journal of computer application,Volume 9.

[17] R. Krovetz, "Viewing morphology as an inference process", In Proceedings of the 16 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1993, pp. 191-202.

[18]"A survey on pre-processing in text mining" Dr.Anandhakumar & Ms.Padmavathy, International journal of advance research in computer science, Volume11.

[20] http:// www.yourdictionary.com/stem