UACEE International Journal of Advances in Computer Science and its Applications – IJCSIA
Volume 3 : Issue 2 [ISSN 2250 – 3765]
Publication Date : 05 June 2013

An effective stemmer in Devanagari script

Monika Dogra
M.TECH: Department of CSE
Lovely Professional University
Jalandhar, India
monikadogra16@gmail.com

Abhishek Tyagi
Assistant Professor: Department of CSE
Lovely Professional University
Jalandhar, India
ABHISHEK.16857@lpu.co.in

Upendra Mishra
Assistant Professor: Department of CSE
IMS Engineering College, Ghaziabad
UP, India
upendra.mishra13@gmail.com

***Abstract*— In today's word of internet web search engines are developing the techniques to make the surfing faster. Stemming is a technique used by web search engines for prefix and suffix removal from the derived word. Stemming provides the way to store similar documents together. This research work aims at the development of Hindi stemmer based on Devanagari script for stripping both prefixes as well as suffixes from derived word to provide better stemming than previous stemmers. Proposed stemmer uses the hybrid approach which is the combination of lookup algorithm, suffix stripping algorithm and prefix removal algorithm.**

***Keywords*— natural language processing, stemming, over-stemming, under-stemming, inflected word, Information Retrieval –IR, conflation.**

I. Introduction

With the increase in networks, technology, information there is requirement of fast and accurate information retrieval. With the increase in information day by day search engines need more efficient techniques for retrieving the data faster and accurate. Stemming is a process used in web search engines for information retrieval. Stemmer reduces inflected words to their root forms, called stem. Using stemming different variants of a tem are conflated to single representative form e. g. stemmer, stemming stemmers all are conflated to single root i.e. *stem*. Stemming extends a search to cover grammatical variations on a word. For example, a search for *wise* would also find *wiser* and *wisest*. A search for *compute* would also find *computing*, *computed*, and *computes*. One more advantage of stemming is that it reduces the index file. In index file only the root word is stored. When a particular word is searched in search engine, web search engine simply returns all the inflected forms related to that word as result of query.

Stemmers are available for many languages including English, French, Arabian languages, Hindi and many more. There are stemmers developed for Hindi but still we lack an efficient stemmer for Hindi. This research paper describes a stemmer for Hindi language based on 'Devanagari' script. This stemmer strips both prefix as well as suffix simultaneously for accurate stemming.

The first section of the research paper includes the introduction part. Second section is about review of literature, in next section available stemming algorithms are discussed; after that stemming problems and proposed stemmer are explained along with flow chart and working images.

II. Review of literature

Natural language processing (NLP) is a field of Computer Science and linguistics concerned with the interactions between computers and human (natural) languages; NLP began as a branch of artificial intelligence. In theory, natural language processing is a very attractive method of Human computer Stemming was introduced in the year 1968 for the first time. Julie Beth Lovins was the first to write a stemmer. A later stemmer was written by Martin Porter and was published in the July 1980 issue of the journal Program. This stemmer was very widely used and became the de-facto standard algorithm used for English stemming. Stemming process is also called conflation [1]. Many algorithms have been developed for stemming in Hindi all has their own advantages and disadvantages. A 'light weight stemmer for Hindi'[2] developed by Ananthakrishnan Ramanathan and Durgesh D Rao is a simple and inexpensive stemmer for

Hindi. This stemmer uses suffix stripping algorithm based on set of rules. It reduces the words to their stem by stripping off the suffixes by using a list of provided suffixes. But this stemmer faces problems of over-stemming and under-stemming. Also the table size for storing words and corresponding suffixes was too large. So it was difficult to manage and it also requires large space for storage and thus slow response time.

A stemmer for Punjabi language 'Design and Development of a Stemmer for Punjabi' [3] by Prince Rana has presented a good combination of stemming techniques. This stemmer uses the hybrid approach by combining two algorithms. This stemmer has combined suffix stripping technique with brute force technique. This stemmer uses a brute force approach. Brute force is a systematic approach to search for all the possible solution in the database. This approach uses a lookup table; this table contains root words and corresponding inflected words. Stemming is done by finding a word in the table if the match is found then the root word is generated. Brute force algorithm requires large number of words in their database. If the word is not found in the database then we are using suffix stripping to handle these words.[5] Because of the use of brute force this stemmer minimizes the problem of over-stemming and under-stemming.

'Maulik an Effective Stemmer for Hindi Language' is the latest stemmer developed for Hindi language. This stemmer is totally based on Devanagari script. This stemmer uses the hybrid approach. It has combined brute force technique with suffix stripping algorithm. Combination of these two approaches makes stemming error free by reducing problems like under-stemming and over-stemming. Brute force technique makes use of database. When search for a word is requested stemmer searches for the word in lookup table. If requested word is found in database then result is returned to the user. If word is not present in database then stemmer uses suffix removal technique for reducing words to their stem. Suffix stripping is based on a set of rules. The use of lookup table requires lots of space but it reduces the errors due to over-stemming and under-stemming. Accuracy of brute force algorithm depends on the accuracy of lookup table. This stemmer is totally based on Devanagari script. It gives better performance by reducing problems like under-stemming and over-stemming. These hybrid approaches gives the best results.

III. Stemming algorithms

There are various algorithms available for the development of stemmers they have been discussed in this section.

i. *Lookup table:* lookup algorithm basically uses database table. Database already contains the roots. Lookup approach is easy, but it requires a large database and hence large amount of manual work is required. It's another disadvantage is that it can only stem those words which are contained in database.

ii. *Suffix stripping algorithm*: This is rule based algorithm for suffixes stripping. This algorithm makes use of predefined rules. E.g. rules in Hindi may be like:-

- if the word ends in **rk**, replace with **'uk'**
- if the word ends in **f;ka** , replace with **' h '**
- if the word ends in **' rs '**, replace with '**uk**'.

iii. *Lemmatization*: A more complex approach to stem a word is lemmatization. In this process we first determine the part of speech of a word, and then apply different normalization rules for each part of speech.

iv. *Hybrid algorithms:* Hybrid approach makes use of combination of two or more approaches. Simple example of hybrid algorithm may be suffix algorithm combined with lookup approach in this hybrid approach first of all; stemmer will look for the root word in the lookup table, if the word is not found in that table only then suffix removal approach is used. Hybrid approach increases the efficiency of the stemmer.

v. *Affix stemmers:* Affix stemmers removes either prefixes or suffixes from the inflected word. The term affix in grammar means prefix or suffix. Different rules are applied to remove prefixes and suffixes according to language in which stemming is performed.

vi. *Matching algorithms*: Matching algorithm is one another algorithm which uses a stem database. All the stems which cannot be divided further are contained in this database. for example for Hindi such words may be : केला , काला , भाई , बादल , दिशा. Choice of the algorithms is made according to language and requirement of stemmer.

IV. Stemming problems

There are two major problems which reduces the quality of stemmer. These two problems Over-stemming and under-stemming are the two performance measurements according to which correctness of a stemmer is measured The proposed stemmer will reduce these problems of over-stemming and under-stemming by using hybrid technique.

i. *Over-stemming*: If two words distinct (having different meanings) to each other are converted to same stem or root, this is called over-stemming problem. E.g. suppose कहावत and कहानी both are reduced to the word कहा. Meaning of कहावत is 'saying or 'quotes', on the other hand meaning of कहानी is 'story'. Even these two words are totally different in meaning still they are reduced to same stem कहा which means 'to say'. This problem is called over-stemming problem.

ii. *Under-stemming*: When two words belonging to same conceptual group are stemmed to different roots this problem is called the problem of under-stemming. For example both the word **better** and **best** must be reduced to the **good** which is the root of both these words, but if they are not all stemmed to good it indicates an error, this is called an under-stemming problem.

v. Proposed stemmer

This research proposal aims at the development of Hindi stemmer based on Devanagari script. The stemmer will work simultaneously for stripping both prefix as well as suffix from derived word to provide better stemming than previous stemmers. This stemmer uses the hybrid approach. Hybrid approach is the one which combines two or more stemming approaches. Three algorithms lookup approach, prefix removal and suffix stripping algorithm are hybridized to achieve accurate stemming in this stemmer. Lookup algorithm basically makes the use of database, known as lookup table. In database root words along with all their inflected forms (derived words) are stored. Database contains a list of thousands of words. When the user enters an inflected word for stemming, the stemmer searches for the presence of that inflected word in the database. If the inflected word is found in the lookup table then its corresponding root word is returned to the user. Lookup table provides users with error free stemming. The correctness of lookup algorithm depends on correctness of words stored into the lookup table. Lookup table would require manual entry of words. Since the words are already entered into the lookup table, errors like over stemming and under stemming are reduced. Second algorithm which stemmer is using is Suffix stripping algorithm, which would work for the removal of suffixes. Suffix stripping algorithm works on set of predefined rules. For the removal of prefixes, third algorithm i.e. prefix removal algorithm is applied. Prefix removal would also be based on the predefined rules. Use of prefix and suffix removal algorithms stems most of the words and gives a better way of stemming which was lacking in previously available stemmers. Moreover, lookup table enhances the quality of stemming by reducing the errors. Hence the proposed stemmer would prove boon for information retrieval in Hindi. Working of stemmer is explained in flow chart given below.

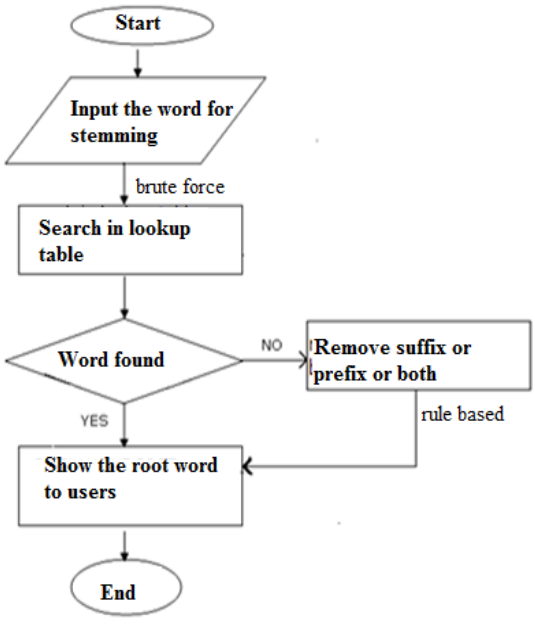

Figure 1 flow chart

In the present stemmer processing of word would be done in three steps. This section explains how user is going to use this stemmer. In very first step user enters the word `vudgs` as shown in figure2. In the next step user press stem button and processing of the word starts. During this processing stage prefix and suffix which are attached to the word are removed. After the processing , output is shown to the user, which is a root word of `vudgs` i. e `dgk` shown in figure 3.

a. *Enter word for stemming*:

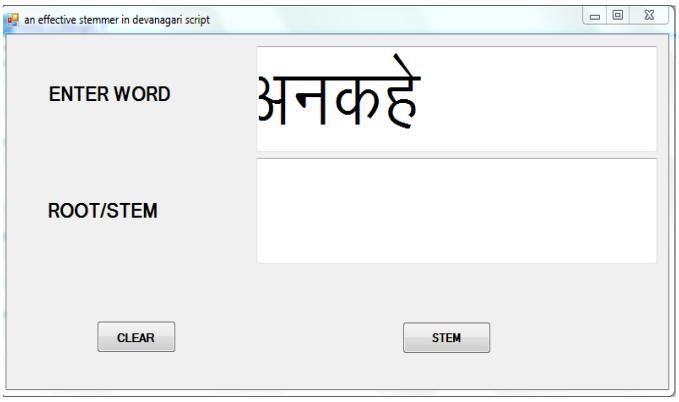

Figure 2 input to stemmer

b. *Processing*: suffixes and prefixes are removed by stemmer in this step based on various predefined rules.

c. *Output:* output is shown to users which consist of a root word.

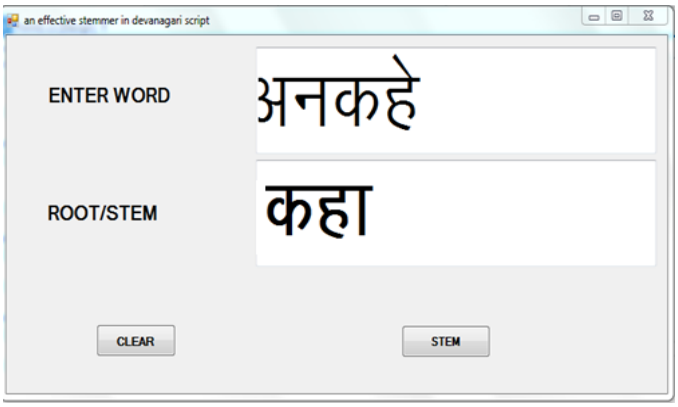

Figure 3 output of stemmer

vi. Results

Performance is directly proportional to efficiency. Accuracy of the stemmer is measured by applying tests. This stemmer works for 60 suffixes and 20 prefixes. Database contains 1500 words. Accuracy is measured by applying given formulae:

$$\text{ACCURACY}(\%) = \frac{\text{Acccurate words after Stemming}}{\text{Number of Inflected words entered}} * 100$$

$$\text{AVERAGE ACCURACY}(\%) = \frac{\text{Sum of All Accuracy}(\%)}{\text{Number Of Groups}}$$

Table 1 accuracy table

s. no	No of groups for testing	No of inflected words entered	Accurate words after stemming	accuracy
1	1st group	300	282	94 %
2	2nd group	200	189	94.5%
3	3rd group	200	187	93.5%
4	4th group	450	415	92.22%
5	5th group	250	234	93.6%
6	6th group	270	264	97.77%

This stemmer is purely based on Devanagari script and it gives the accuracy of 94.26%. This stemmer is very effective in terms of fewer problems like under-stemming and over stemming.

VII. Conclusion and Future work

This stemmer gives accuracy of 94.26% and reduces the stemming problem. Though this stemmer is effective for both prefixes and suffixes but still it has a big drawback. This stemmer is using a lookup table approach which is inefficient approach. Lookup table approach basically makes use of database. Manually words have to be stored in database. Lookup approach reduces the stemming errors but it requires lots of manual work and storage space. This stemmer can be made much more efficient in terms of space requirement by creating dictionary free stemmer.

Acknowledgment

I express my profound sense of gratitude to the support given to me by mentors for their support and encouragement. 1 also acknowledges the encouragement and moral support received from my family and friends.

References

[1] Julie Beth Lovins, "Development of a Stemming Algorithm*"Mechanical Translation and Computational Linguistics, 1963,Vol No.11, Issue No.1, pp 22-31.

[2] Dinesh Kumar and Prince Rana "Design and Development of a Stemmer for Punjabi", International Journal of Computer Applications (0975 – 8887) Volume 11– No.12, December 2010.

[3] Ananthakrishnan Ramanathan and Durgesh D Rao "A Lightweight Stemmer for Hindi" In Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL), on Computatinal Linguistics for South Asian Languages (Budapest, Apr.) Workshop,pp 42-48.

[4] MAULIK: An Effective Stemmer for Hindi Language Upendra Mishra et al. / International Journal on Computer Science and Engineering (IJCSE) Vol. 4 No. 05 May 2012

[5] Mudassar M. Majgaonker and Tanveer J Siddiqui(2010) "Discovering suffixes: A Case Study for Marathi Language" International Journal on Computer Science and Engineering Vol. 02, No. 08, 2010, 2716-2720.

[6] Abduelbaset M. GOoweder, Husien A. Alhammi , Tarik Rashed, and Abdulsalam Musrati " A Hybrid Method for Stemming Arabic Text " Journal of computer Science, URL: http://eref.uqu.edu.sa/files/eref2/folder6/f181.pdf.

[7] M.F Porter (1980)" An algorithm for suffix stripping" Published in Program, Vol No.14, Issue No.3, pp 130-137,URL:http://www.cs.odu.edu/~jbollen/IR04/readings/readings5.pdf.

[8] http://www.ispeakhindi.com/2012/04/04/suresh-295-suffixes/

[9] http://en.wiktionary.org/wiki/Category:Hindi_suffixes

[10] http://blogs.transparent.com/hindi/primary-suffix-in-hindi/

About Author (s):

Ms. Monika Dogra is currently pursuing Masters of Technology from Lovely Professional University(LPU), Phagwara, India. She has done B.Tech in Computer Science from Lovely Professional University. Her Research area of interest includes Natural Language Processing and Machine Learning, handwriting recognition.

Mr. Upendra Mishra is working as Assistant Professor in Computer Science Department of IMS Engineering College, UP, India. He did his Masters of Technology (MTech) in C.S.E. from Lovely Professional University (LPU), Punjab. And Bachelors of Technology (BTech) in C.S.E from College of Engineering & Management, Punjab .His areas of research are Natural Language Processing (NLP) and Machine Learning.

Mr. Abhishek Tyagi is currently working at the rank of Assistant Professor in Computer Science Department of Lovely Professional University, Jalandhar, India. He did his Masters of Technology (MTech) in Computer Science from Lovely Professional University (LPU),