Lightweight Stemming Approach for Punjabi Language Text: An NLIDB Subsystem Alternative

Harjit Singh[1], Ashish Oberoi[2]
[1]APS Neighbourhood Campus, Punjabi University Patiala, Punjab, India
[2]School of Engineering, RIMT University, Mandi Gobindgarh, Punjab, India
(e-mail: hjit@live.com)

Abstract— The stemmer is a basic component of any natural language processing application. The purpose is to remove any suffixes attached to the basic word forms so that those words could be processed more efficiently for further use. There are various stemmers developed for different languages. There also exist some stemmers already developed for Punjabi language text. Those are efficient for use in natural language processing but are heavyweight as developed to stem large amount of text for translation and transliteration purposes. A natural language interface to databases (NLIDB) processes a natural language query which is single or two line question. Most of the words in the query are useless so they are removed and need not be stemmed. The remaining words will mostly be the nouns which belong to the entities about which the data is queried. The heavyweight stemmer takes a number of steps to stem a single word which may reduce the overall processing speed of NLIDB system. A lightweight stemmer efficient enough to stem only required words is sufficient to get the required speed of processing from NLIDB system. For this purpose, a simple and efficient approach is presented in this paper. The approach can be extended further to improve the efficiency for stemming large amount of text.

Keywords— *Natural Language Processing, Punjabi Stemmer, Lightweight Stemmer, Lexicon based stemming*

I. INTRODUCTION

A word can be affixed in number of ways for use in a sentence making it inflected word. The affix is attached to make the sentence grammatically correct. A set of rules of grammar are followed when writing formatted sentences in a language. Every language uses a specific grammar and hence follows those rules. Natural Language Processing (NLP) is a research area under Artificial Intelligence (AI) that works for processing of human languages in order to make machines or computers understand them. Basically, it is to make human-computer interaction in natural languages.

The words of the text are processed exclusively one by one. The suffixes or prefixes attached to the words make the process ambiguous and more cumbersome. So these prefixes and suffixes are removed to make stem word by the procedure called stemming. Stemming uses approaches like Brute Force approach using data tables, Rule based approach using set of rules, Statistical approach etc. There are various stemmers developed for different languages. There also exist some stemmers already developed for Punjabi language text. Those are efficient for use in natural language processing but are heavyweight as developed to stem large amount of text for translation and transliteration purposes [1].

Dinesh Kumar et al. (2010) [2] developed a Punjabi language stemmer using Brute Force Approach. A database of words is used containing root and inflected words. It searches word in database with Brute Force technique. The matched word if found within database provides corresponding root word. If no match occurs, it creates the root word by removing the suffix from input word. The suffix removal process uses a list of suffixes of Punjabi language. It is process in two steps to make the stem word. The step one try to find word in the database table, if failed step two removes part of the word making stem word.

Dinesh Kumar et al. (2011) [3] developed a Punjabi language stemmer using same Brute Force Approach. A database of words is used containing root and inflected words. It searches word in database with Brute Force technique. The matched word if found within database provides corresponding root word. If no match occurs, it creates the root word by removing the suffix from input word. The suffix removal process uses a list of suffixes of Punjabi language. It is process in two steps to make the stem word. The step one try to find word in the database table, if failed step two removes part of the word making stem word. By storing more number of words in the database, they showed the performance improved over the earlier one in this stemmer.

Vishal Gupta et al. (2011) [4] developed a Punjabi stemmer to stem words being absent in dictionaries. The dictionary words are grammatical words but the words used in a language that are not available in dictionaries may be proper names or may be invalid words. The stemmer tries to remove the suffixed part of words including attaching alternate suffix to make stem words. Proper name is the name of some place or some person or some concepts etc. A collection of those names including nouns is prepared by author for validity of output by checking the word in the list. It uses some rules to remove an affix and make a stem of word. It also attaches some suffix to the word (if required) to stem correctly.

Chandni Dhawan et al. (2013) [5] developed a stemmer using hybrid technique. In the technique suffix stripping, suffix substitution and Brute Force is used. Two database tables. Table one having root words, table two having removable suffixes and attachable suffixes (if any). It searches word in database with Brute Force technique. The matched word if found within database confirms root word. If no match occurs, it creates the root word by removing the suffix from input word. The suffix removal process uses a list of suffixes of Punjabi language. If required, alternate suffix is appended to the word to stem correctly. The output is matched in table of root words for surety.

Vishal Gupta (2014) [6] developed a Punjabi stemmer which can stem nouns or pronouns or verbs or adverbs or adjectives or proper names. Lists are prepared for verb suffixes and adverb suffixes and adjective suffixes including pronoun suffixes. It creates the root word by removing the suffix from input word. If required, alternate suffix is appended to the word to stem correctly. Proper names may be names of places, persons or concepts etc. Punjabi words collection is created by author for validity checking of output by matching in the collection failing which displays an error message.

Garima Joshi et al. (2014) [7] developed a stemmer using hybrid approach including Table lookup with Rule based approach. Table lookup is same Brute Force. A collection of root words, a collection of inflected words and their root words, a collection of synonyms, a collection of rules for suffix setting is used. It searches word in database with Table Lookup technique. The matched word if found within the collection of root words confirms the root word. If no match occurs, it creates the root word by removing the suffix from input word using collection of rules for suffix setting. If required, alternate suffix is appended to the word to stem correctly. The output is matched in collection of root words for surety.

Puneet Thapar (2014) [8] developed stemmer using Naïve algorithm. A lookup table having inflected words including root words used. It searches word in lookup table with Naive technique. The matched word, if found within lookup table provides corresponding root word. If no match occurs, it creates the root word by removing the suffix from input word. The suffix removal process uses a list of suffixes of Punjabi language.

Rajeev Puri et al. (2015) [9] developed stemmer with the help of Punjabi Wordnet. It is a revised stemmer showing improvement over already developed stemmer by Vishal Gupta et al. (2011). It uses some rules to remove an affix and make a stem of word. It also attaches some suffix to the word (if required) to stem correctly. A collection of root words is prepared by author for validity of output by checking the word in the list, failing which displays an error message.

So as above, this paper provided the information regarding almost all the stemmers developed on Punjabi stemming that motivated to develop some more scheme in this area. It contributed to propose lightweight Punjabi language stemmers.

In this paper, Section I is the Introduction and review of already developed stemmers for Punjabi language, Section II describes the need of lightweight stemming, Section III describes my Lightweight Stemming Approach for Punjabi Language Text, and Section IV concludes the contents of paper.

II. NEED OF LIGHTWEIGHT STEMMING

A natural language interface to databases (NLIDB) processes a natural language query which is single or two line question. Most of the words in the query are useless so they are removed and need not be stemmed. The remaining words will mostly be the nouns which belong to the entities about which the data is queried. The heavyweight stemmer takes a number of steps to stem a single word which may reduce the overall processing speed of NLIDB system.

A lightweight stemmer efficient enough to stem only required words is sufficient to get the required speed of processing from NLIDB system. For this purpose, a simple and efficient approach is presented in this paper. The approach can be extended further to improve the efficiency for stemming large amount of text.

III. LIGHTWEIGHT STEMMING APPROACH

The stemming approach presented here is a lightweight stemming process that is suitable to stem required words from a Punjabi language query after removing useless words. The approach uses a lexicon of only base word forms. There is no need to store inflected forms of each word which may reduce the processing speed of stemming. The word to be stemmed is taken as input word which may already be a stemmed word or may be an inflected form of any base word stored in the lexicon. The stemming process will be successful if the stem form of the word will be available in the lexicon of base word forms. So the correctness of this approach is directly related to the size of the lexicon. If the correct stem word is absent in the lexicon, the stemming approach may end up with incorrect stemming which may lead to inaccuracies in further processing or words.

The input word is searched in the lexicon of base word forms. If the word is found as a whole in the lexicon, then it is already a stem word and there is no need to stem it. If input word is not matched as a whole, then it may be an inflected form of some base word stored in the lexicon of base word forms. It may have attached a suffix composed of one, two or three characters. The character(s) may be vowel(s) but in most cases it is a consonant or a combination of vowel(s) and consonant(s). Whatsoever the case may be, we are not going to match exact suffix pattern in this approach. If the input word is not found as a whole in the lexicon of base word forms, the next step is to remove one character from the end of input word and attach a ਾ (kanna) and then pattern match it with the words available in the lexicon of base word forms. There may not be any match, which will return the empty result set. But there may also occur a match with one or more words stored in the lexicon. In this case the result set of words returned is sorted based on the word length in ascending order. The first word in the sorted result set of words (i.e. the smallest word) in taken as the output stem word. If no word is found then remaining word is directly pattern matched with the words available in the lexicon of base word forms without attaching any ਾ (kanna) and smallest word is taken as output stem word.

Now, reconsidering the situation, if input word is not pattern matched with any word in the lexicon of base word forms, the result set returned will be empty. In this case, two characters are removed from the end of the input word and remaining word by attaching a ਾ (kanna) is again pattern matched with the words available in the lexicon of base word forms. There may not be any match, which will return the empty result set. But there may also occur a match with one or more words stored in the lexicon. In this case the result set of words returned is sorted based on the word length in ascending order. The first word in the sorted result set of words in taken as the output stem word. . If no word is found then remaining word is directly pattern matched with the words available in the lexicon of base word forms without attaching any ਾ (kanna) and smallest word is taken as output stem word.

Again, reconsidering the situation, if input word is not pattern matched with any word in the lexicon of base word forms, the result set returned will be empty. In this case, three characters are removed from the end of the input word and remaining word by attaching a ਾ (kanna) is again pattern matched with the words available in the lexicon of base word forms. There may not be any match, which will return the empty result set. But there may also occur a match with one or more words stored in the lexicon. In this case the result set of words returned is sorted based on the word length in ascending order. The first word in the sorted result set of words in taken as the output stem word. If no word is found then remaining word is directly pattern matched with the words available in the lexicon of base word forms without attaching any ਾ (kanna) and smallest word is taken as output stem word.

Efficiency of this algorithm is dependent solely on the availability of correct stem words in a Lexicon without any inflected word. The algorithm is implemented in C# using SQL Server as backend database to store Lexicon of stem words. The implementation is tested with 618 Punjabi inflected nouns and test results are shown in Table 1.

Table 1: Efficiency Calculations

No. of characters removed	No. of words correctly stemmed	Percentage
One Character Removal	211	34.1423948
Two Character Removal	270	43.6893204
Three Character Removal	62	10.0323625
Accuracy Percentage =	**543**	**87.8640777**

Algorithm 1: Lightweight Stemming Algorithm

```
Step 1: Let Word Input as $W_i$
            Stem Output as $S_o$
            Lexicon as $L_x$
            Result Set as $R_s$
            Kanna as ਾ
Step 2: Read $W_i$
Step 3: Search $W_i$ as a whole in $L_x$
Step 4: If Found then $S_o=W_i$, Goto Step 21
Step 5: Remove One Character from End of $W_i$
Step 6: Attach ਾ to $W_i$ and Pattern Match in $L_x$
Step 7: If Match Found Then
           Sort $R_s$ in Ascending Order
           $S_o$=1st Word in Sorted $R_s$
           Goto Step 21
Step 8: Pattern Match $W_i$ in $L_x$
Step 9: If Match Found Then
           Sort $R_s$ in Ascending Order
           $S_o$=1st Word in Sorted $R_s$
           Goto Step 21
Step 10: Remove One More Character from End of $W_{i\backslash}$
Step 11: Attach ਾ to $W_i$ and Pattern Match in $L_x$
Step 12: If Match Found Then
           Sort $R_s$ in Ascending Order
           $S_o$=1st Word in Sorted $R_s$
           Goto Step 21
Step 13: Pattern Match $W_i$ in $L_x$
Step 14: If Match Found Then
           Sort $R_s$ in Ascending Order
           $S_o$=1st Word in Sorted $R_s$
           Goto Step 21
Step 15: Remove One More Character from End of $W_i$
Step 16: Attach ਾ to $W_i$ and Pattern Match in $L_x$
Step 17: If Match Found Then
           Sort $R_s$ in Ascending Order
           $S_o$=1st Word in Sorted $R_s$
           Goto Step 21
Step 18: Pattern Match $W_i$ in $L_x$
Step 19: If Match Found Then
           Sort $R_s$ in Ascending Order
           $S_o$=1st Word in Sorted $R_s$
           Goto Step 21
Step 20:Word Not Found in $L_x$
Step 21: Display $S_o$
```

Most of the words are stemmed by removing two characters from the input word and many are stemmed by removing one character, very little number of correct stemming are done by removing three characters from the input word. The algorithm for Lightweight Stemming Approach for Punjabi Language Text is shown as Algorithm 1.

IV. CONCLUSION

The purpose of stemming is to remove any suffixes attached to the basic word forms so that those words could be processed more efficiently for further use. There are various stemmers developed for different languages. There also exist some stemmers already developed for Punjabi language text. Those are efficient for use in natural language processing but are heavyweight as developed to stem large amount of text for translation and transliteration purposes. A natural language interface to databases (NLIDB) processes a natural language query which is single or two line question. Most of the words in the query are useless so they are removed and need not be stemmed. The remaining words will mostly be the nouns which belong to the entities about which the data is queried. The heavyweight stemmer takes a number of steps to stem a single word which may reduce the overall processing speed of NLIDB system. A lightweight stemmer efficient enough to stem only required words is sufficient to get the required speed of processing from NLIDB system. For this purpose, a simple and efficient approach is presented in this paper. The approach can be extended further to improve the efficiency for stemming large amount of text.

REFERENCES

[1] Cristian Moral, Angélica de Antonio, Ricardo Imbert and Jaime, Ramírez, "A survey of stemming algorithms in information retrieval", Information Reseach, Vol. 19, No. 1, March, 2014

[2] Dinesh Kumar, Prince Rana, "Design and Development of a Stemmer for Punjabi", International Journal of Computer Applications (ISSN: 0975 – 8887) Volume 11– No.12, pp. 18-23, December 2010

[3] Dinesh Kumar, Prince Rana, "Stemming of Punjabi Words by using Brute Force Technique", International Journal of Engineering Science and Technology (IJEST) ISSN : 0975-5462, Vol. 3 No. 2, 1351-1358, Feb 2011

[4] Vishal Gupta, Gurpreet Singh Lehal, "Punjabi Language Stemmer for nouns and proper names", Proceedings of the 2nd Workshop on South and Southeast Asian Natural Language Processing (WSSANLP), IJCNLP 2011, Chiang Mai, Thailand, pp. 35–39, November 8, 2011

[5] Chandni Dhawan, Jashanpreet Singh, Kamaldeep Garg, Hybrid Approach for Stemming in Punjabi, International Journal of Computer Science & Communication Networks (ISSN:2249-5789),Vol 3(2), pp. 101-104, 2013

[6] Vishal Gupta, "Automatic Stemming of Words for Punjabi Language", Advances in Signal Processing and Intelligent Recognition Systems-Vol. 264, Springer International Publishing Switzerland, DOI: 10.1007/978-3-319-04960-1_7, pp. 73-84, 2014

[7] Garima Joshi, Kamal Deep Garg, "Enhanced Version of Punjabi Stemmer Using Synset", International Journal of Advanced Research in Computer Science and Software Engineering (ISSN: 2277-128X), Volume 4, Issue 5, pp. 1060-1065, May 2014

[8] Puneet Thapar, "A Hybrid Approach used to Stem Punjabi Words", International Journal of Computer Science and Mobile Computing (ISSN 2320–088X), Vol. 3, Issue. 11, pp.1 – 9, November 2014

[9] Rajeev Puri, R. P. S. Bedi, Vishal Goyal, "Punjabi Stemmer Using Punjabi WordNet Database", Indian Journal of Science and Technology, Vol 8(27), DOI:10.17485/ijst/2015/v8i27/82943, pp. 1-5, October 2015

Authors Profile

Mr. Harjit Singh received the MCA (Master in Computer Applications) degree from IGNOU (Indira Gandhi National Open University), New Delhi, India and M.Phil.(CS) degree from Global Open University, Nagaland, India. He is working as Assistant Professor (Senior Scale) in Computer Science at Punjabi University Neighbourhood Campus Dehla Seehan, Sangrur, India. He is pursuing Ph.D. degree from RIMT University, Mandi Gobindgarh (Punjab). His current research interests include Natural Language Processing, Machine Translation, Artificial Intelligence.

Dr. Ashish Oberoi is working as Professor in Computer Science and Engineering at School of Engineering, RIMT University, Mandi Gobindgarh, Punjab, India. He completed his Ph.D. in Computer Science and Engineering from Maharishi Markandeshwar University, Mullana, India. He is skilled and expertise in Image Processing, Medical Imaging, Image Segmentation, Diagnostic Imaging and Image Reconstruction.