ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol4, Issue 6, June 2017

Statistical Stemmer for Roman Konkani

[1] V. Pugazhenthi, [2] Sagar Naik,[3] Neha Sawant,[4] Raghavendra Gawas,[5] Shradha Vainganker,
[6] Snusha Shirodkar,[7] Charitra Chandekar
[1][2] Assistant Professor, Computer Engineering, Agnel Institute of Technology and Design, Goa University, India
[3][4] [5] [6] [7] BE Students, Computer Engineering, Agnel Institute of Technology and Design, Goa University, India

***Abstract*— Stemming aims at obtaining the stem of a word by removing its suffixes and prefixes. It is widely used in Information retrieval and Natural Language processing systems. Few stemmers have already been developed for Konkani, the official and widely spoken language in the state of Goa but in Devanagari script. This paper develops a stemmer for Konkani in Roman script, in w hich, each Devanagari letter is resembled by a Roman character, equal to its Devanagari equivalent. A statistical stemmer is developed which clusters the morphological variants of a word into a single cluster, based on string distance measure. The Stem words are extracted from each cluster by using Longest Common Prefix algorithm. In case of singleton cluster, the stem words are extracted by applying suffix stripping methodology.**

***Index Terms*— Information Retrieval (IR), Stemming, Clustering, Rule-based stemmer, Distance measure, Statistical Based Stemming, Morphology, Root word, Inflection, Hierarchical clustering.**

I. INTRODUCTION

Stemming is the process of stripping affixes like prefixes and suffixes, from words to form a stem word. Thus it is a basic pre-processing step in applications like Natural Language processing, Language Modelling and Information Retrieval (IR). In Information Retrieval (IR) systems this process is often applied to words so that the words with almost the same meaning are grouped together as the same concept. 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. This reduces the size of the index files to a very great extent as different words are grouped under the same stem requiring only one entry in the index file. This increases the space efficiency and retrieval time of the IR system.

The Stemming process is also called as conflation which maps different variants of the words to its stem. Need for stemming arises to avoid mismatch between the query asked by the user and the stems present in the index files. For example if a user enters a query "pustokam" meaning "books" he might not get all the expected results. But if the query is stemmed, so that "pustokam" becomes "pustok",then the retrieval results will be more accurate, successful and relevant. The general idea is shown in below figure

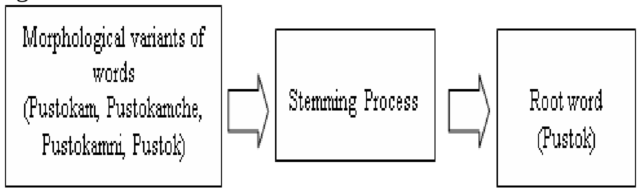
Morphological variants of words (Pustokam, Pustokamche, Pustokamni, Pustok)
Stemming Process
Root word (Pustok)

Figure 1: Stemming process

There exist many stemmers for various Indian Languages like Hindi, Marathi, Bengali, Kannada etc. A little work has been carried out for stemming Konkani regional language only in Devanagari script. Till today there does not exits a stemmer which will stem Konkani in roman script. So there is a need to develop a tool for stemming Konkani in roman script. Thus the main focus of this paper is to extract stem words from the given Konkani document scripted in Roman alphabets using the proposed approach. For determining the performance effectiveness, the stemmer was implemented by using complete linkage clustering algorithm [1] and the proposed threshold-based clustering techniques. This paper is organized as follows. Section I contains the introduction of the stemming process. Section II discusses in detail the works that has been carried out related to stemming for some of the regional languages of India. Section III provides the morphological characteristics of Konkani language with respect to roman scripts. Section IV presents the fundamental concept of the proposed stemming technique. Section V presents the obtained experimental results. Finally, section VI concludes the main achievements and also the limitations of this paper.

II. RELATED WORK

YASS: Yet Another Suffix Stripper [1] is purely an unsupervised clustering based stemming algorithm designed for Bengali language. This stemmer defined a set of string distance measures D1, D2, D3, D4 and the lexicon for the given text collection is clustered using this distance measures to identify their equivalence classes. This string distance measure is used to check the similarity of two words by calculating the distance between two words. To make the string lengths equal, the shorter string is appended with null characters. Edit distance counts the minimum number of edit operations required to transform one string to the other. Then the most effective distance measure out of four is chosen along with an appropriate threshold value. This stemmer being language independent can be applied to any language without having much knowledge about its morphological characteristics. Since complete linkage clustering technique was used, determining the stopping criterion is difficult which resulted in computationally expensive clustering step. In a paper titled, "Statistical Stemming for Kannada" [2], Suman Bhat developed a statistical stemmer for Kannada in which a set of string distance measure were used to check the similarity between two strings by calculating the distance between them and determine a threshold that results in good stemming performance. Clustering was performed using average linkage clustering and the obtained results were compared with traditional stemmers.

"MAULIK: An Effective Stemmer for Hindi Language" [3] is a hybrid approach for stemming that combined brute force and suffix removal techniques. Brute force approach employed a lookup table which maintained large number of inflected words as well as their corresponding root word. If the word was not found in the lookup table, then suffix stripping methodology was used in which certain predefined rules were defined to remove the commonly used suffices from these inflected words. Problems of over-stemming and under-stemming were effectively controlled by using this approach.

III. ROMAN KONKANI

Konkani, the official language of Goa is generally written in Devanagari script. People having difficulty in using the Devanagari script, Roman Konkani is an alternative approach and it also becomes easier for the computers to process the Roman script. Konkani in Roman script or "Romi Konkani" actually refers to the process of writing the Konkani language in the roman script where each Konkani character is replaced by its Roman character, equal to its Devanagari equivalent. Much of the vocabulary of Konkani comes from Sanskrit. It uses a combination of 52 characters consisting of 10 vowels and 36 consonants as shown in Figure 2 and Figure 3 below.

अ	आ	इ	ई	उ	ऊ	ए	ऐ	ओ	औ
a[ə]	a	i	ī	U	ū	e	ai	o	au

Figure 2: List of Vowels

क k	ख kh	ग g	घ gh	ङ gna
च c`	छ c`h	ज j`	झ j`h	ञ ña
ट t◌	ठ t◌ h	ड d◌	ढ d◌ h	ण n◌
त t	थ th	द d	ध dh	न n
ऩ n	प ph	फ b	ब bh	भ m
म y	य r	र l	ल w	ळ ĕ
स s	ह h	ऱ l◌		

Figure 3: List of Consonants

IV. STEMMER FOR ROMAN KONKANI

The objective of this project is to obtain a root word by removing inflections and suffixes of the words. The input document in Roman Konkani is tokenized in order to obtain the tokens which can be handled individually. All the obtained tokens are then filtered to eliminate the stop words and special characters. Now all the valid tokens are clustered using the below mentioned string distance measure to find the similarity between the word.

$$\text{Distance (Str1, Str2)} = \frac{n-m+1}{m} * \sum_{i=m}^{n} \left(\frac{1}{2^{i-m}}\right) \quad \text{, if } m>0,$$

$$\text{, otherwise} \qquad \infty$$

where n is the length of the longest string between Str1 and Str2 and m is the position of the first mismatch.The available morphological variants of the root word will be grouped together, forming a cluster. Finally root words are obtained either by applying Longest Common Prefix Matching algorithm on the clusters or by applying inflection stripping rules in case of singleton clusters. Figure 4 depicts the proposed stemmer architecture.

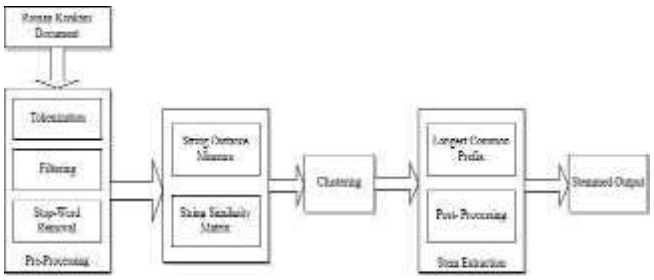

Figure 4: Proposed Stemmer for Roman Konkani

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol4, Issue 6, June 2017

The following steps are involved in extracting the stems from the Roman Konkani documents given as the inputs.

Tokenization

The inflated sentences in a input document are broken up into constituent pieces, called tokens. The blank space, tab and comma are used as delimiters in order to generate tokens.

Filtration

The presence of special characters like !@ # $ % ^ & * ( ) _ - + = { [ ] } : " < > ? ? : used in the input document degrades the performance of the system and hence needs to be removed. This process of removal of special symbols is called filtration of the document. In this step tokens containing punctuations and special characters are filtered.

Stop-Word Removal

Stop words are the most frequently occurring words in the roman script that has no meaning for keyword based information retrieval system. The presence of stop words results in more space and more retrieval time degrading the performance of the system. Thus their removal is important. The most commonly occurring stop words in Roman Konkani are listed below in the table 1

zo	he	te	ti	vo	hanga	tor	na
tem	the	khay	phati	hacho	tacho	odhik	mekant
sokot	voi	tea	taka	vont	ki	hem	ho
re	toso	ani	tum	magir	hanv	ee	kon
kenna	naka	punn	oxe	favtt	mhaka	mat	khub
khatir	to	hea	legit	teo	koxe	magir	

Table 1: Commonly used Stop words in Roman Konkani

String Distance Measure and Similarity Matrix

This step includes mapping of each pair of lexicons to a real number indicating the similarity between them.

Clustering

To compare the performance effectiveness, the stemmer was implemented in two ways. First by using the complete linkage clustering technique [1] and then using the threshold based clustering technique [2]. All the valid tokens are clustered which groups the morphological variants of a word in a single cluster. For threshold-based clustering the value of the threshold T is computed as

$$T = \begin{cases} \left(\frac{(MIN+MAX)}{2}\right) + \text{Distance }(X, Y), & \text{if } MAX > \text{length}(X, Y) \\ \left(\frac{MIN}{2}\right), & \text{Otherwise.} \end{cases}$$

where MIN and MAX are the minimum and maximum distance value between any pair of strings in the String similarity matrix.

Stem Extraction

Root words are obtained either by applying Longest Common Prefix Matching algorithm on the clusters or by applying suffix stripping rules in case of singleton clusters [4].

V. EXPERIMENTAL RESULTS

Both the approached were tested on three different Roman Konkani documents (data sets). The following Table 2 shows the comparison results of the two approaches.

Data set		Complete Linkage Clustering		Proposed Threshold based Clustering	
Name	No of Inflected words	Accurate words after stemming	Accuracy %	Accurate words after stemming	Accuracy %
DS 1	89	74	83.1%	55	61.8%
DS 2	101	81	80.2%	74	73.3%
DS 3	145	101	70%	87	60%

Table 2: Comparison results of both methods

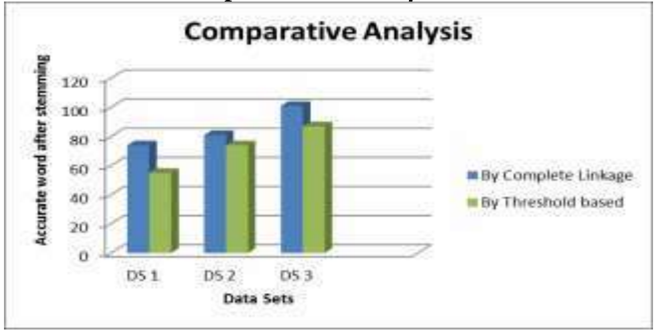

Figure 5: Comparative analysis

VI. CONCLUSION

The Stemmer proposed in this project employs a statistical stemming technique for Konkani in roman script. It uses a clustering based approach wherein the variants of the same word are clustered into the same cluster and then the stem word is obtained by applying string longest common prefix algorithm on that cluster. Though the stemmer was developed with the intent of employing the statistical based strategy which would work for any language without knowing the morphological characteristics of the language, there are few cases where the clusters might contains just a single word. In such cases, these clusters are post processed by removing the common suffices from the words by simplified rules, to extract the stem word. These techniques reduce the problem of under-stemming and over-stemming. The limitation of the System is the lack of human interference in the final steps. In future more accurate results could be obtained by ensuring involvement of the user and by adding extra suffix stripping rules in case of singleton clusters. Though the accuracy of the threshold based stemmer is less than complete linkage, it does not requires human intervention, and it automatically terminates when the required clusters are formed.

REFERENCES

[1] P. Majumdar, M Mitra, S.K. Parui, G. Kole, P. Mitra and K. Datta, “YASS: Yet Another Suffix Stripper,” Association for Computing Machinery Transactions on Information Systems, 25(4):18-38, 2007.

[2] Suma Bhat, “Statistical Stemming for Kannada”, Workshop on South and Southeast Asian NLP (WSSANLP), 2013.

[3] Upendra MishraM, “MAULIK: An Effective Stemmer for Hindi Language”, International Journal on Computer Science and Engineering (IJCSE), Vol. 4 No. 05 May 2012, 711-717.

[4] Mudassar M. Majgaonker , “Discovering suffixes: A Case Study for Marathi Language”, International Journal on Computer Science and Engineering Vol. 02, No. 08, 2010,
2716-2720.

[5] Vishal Gupta, “ Hindi Rule Based Stemmer for Nouns”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 1, January 2014.