ijecs
open access www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 5 Issues 6 June 2016, Page No. 17113-17126

Iterative Success Variety Method for Joined words in Tamil Language

D.Umamageswari[1], K.Kumanan[2]

[1]D.Umamageswari
Computer Science & Engineering,
Sri Venkateswara College Engineering,
Sriperumbuthur, India
umamageswari.devaraji@gmail.com

[2]Mr.K.Kumanan, M.Tech
Computer Science & Engineering,
Sri Venkateswara College Engineering,
Sriperumbuthur, India
kkumanan@svce.ac.in

Abstract— ***Stemmer plays a vital role in Natural Language Processing applications, where it is used to improve the accuracy of applications like Information Retrieval System(IRS) for indexing based on recall/precision factors, POS Tagger, Search engines and so on. Stemmer is a pre-processing step to squeeze out the root or stem or base of the words. Stemmer for class of words like nouns i.e plural form or verbal form of simple words is available. This paper presents a stemmer for joined words or compound words which is again a class of words formed as, the beginning character of the next word, formed with some form of the ending character of first word. It uses the dictionary of root or stem or base words to squeeze out the vital part of the word. It produces better accuracy when there are large numbers of root or stem or base word in the dictionary. It is working as context-based stemmer where it selects the second root or stem or base of the word based on context. So, it needs to collect the vital part of the words in the class of nouns and verbs.***

Keywords— Stemming, Stemmer, Natural Language Processing, Tamil

1. Introduction

Processing of Natural Language is a systematic approach, deals with the interaction of human and machine, which analyses and processes the words to produce meaningful information. It mainly focuses on the processing of text with the language structure and requires knowledge about language in order to develop an automated system. Natural Languages are ambiguous, requires a fine-tuned pre-processing task like Stemmer of NLP. It is a pre-processing task in many NLP like POS Tagging, machine translation, text repository in teaching and learning process, morphology, synthesizer, information retrieval and extraction, parser, and disambiguation.

The Stemmer is the process of squeezing out the root or stem or base of the given word. The words are classified as simple and compound words, where the compound words are made up of two or more words combined together. Here this paper considers the compound words that are formed as, the beginning character of the second word is the ending character of the first word with slight modifications.

Example: “uyirinangaL”, here we have two root words.

They are “uyir” and “inam”. One is noun and another is verb. If these words are available in the dictionary, then they can also be collected from the joined words.

2. An Overview

In this context, describes a brief summary of various Stemming approaches for Tamil is being analyzed.

Tamil is an agglutinative language, where the words are created by affixing root or stem with either suffixes or prefixes called derivational words or inflected or compound words.

The major complexity in the process of stemming, is in identifying the root or stem word in joined words, over-stemming and under-stemming.

Tamil language also follows free-word order, and due to highly inflectional and morphological form of words, it is better to use Rule-based affix stripping method. General types of Stemmer are,

1. Rule-based Approach.
2. Statistical Approach.

2.1 Rule-Based Approach

In Tamil language, where the words are created by affixing root or stem with either suffixes or prefixes.

Rules should be framed to extract the key part of the given word. It is using a pre-defined large set of well-formed rules, intended to be language-specific and deep knowledge of the language.

Advantages

a. Does not require any Dictionary.
b. Faster.

Disadvantages

a. Extensive knowledge of the language is required.
b. Requires considerable amount of memory to store rules.
c. It has over-stemming and under-stemming problem.

2.2 Statistical Approach

Alternative solution for stemming uses statistical information of the large trained set. It yields better results than rule based.

Advantages

a. Language Expertise is not required.

Disadvantages

b. Requires good algorithm.

c. Requires Large Corpus.

Various algorithms used are,

1. Affix Stripping or Suffix Removal method.
2. Success variety method.
3. Table look-up method.
4. N-Gram method.

Some of the algorithms are explained below,

2.2.1 Suffix Removal Method

It does not use any look-up table. It uses a set of rules to find the root or stem word for the given set of words.

Example:

If a word ends with '¸¢È¡÷' remove '¸¢È¡÷'

If a word ends with '¾¡÷' remove '¾¡÷'

If a word ends with 'Å¡÷' remove 'Å¡÷'

It is a simpler form of Stemming requires the language expertise.

2.2.2 Success Variety Method

Successor variety stemmer works on structural form of words, in which it uses frequencies of the letter sequences in the word with the set of words beginning with same letter. Consider the next character of each word and selects the appropriate one and repeats the same until it gets the correct stem.

Example: to find 'ÀÄ÷' among the set of words, ÀÄ÷, ÀÄ¡, À¡ø

To find 'ÀÄ÷'	Success Variety count	Characters
À	3	{Ä, Ä, ¡}
ÀÄ	2	{÷, ¡}
ÀÄ÷	1	-

For segmenting it uses some more algorithms along with Success variety method like cutoff, peak plateau methods.

2.2.3 Related Works

For the language English, Julie Beth Lovins was developed a Dictionary-based Stemming algorithm in 1968, used Suffix list with 294 suffixes and 29 context rules for removing the suffixes if matches. In 1980, another algorithm called Suffix Stripping Algorithm used in Porter Stemmer for English was developed and it provides better results than Lovins Algorithm.

Then for the other foreign languages and Indian languages, the Stemmer was developed based on the Lovins and Porter Stemmer. Most of the work for the Indian languages like Hindi, Guajarati, Punjabi, etc., done either with inflected or derived words, but very few stemmers was developed with both the inflected or derived words.

For rich morphological and highly inflected language like Tamil, [1] says about various stemming algorithm for Indian and non-Indian languages. It is used to improve the effectiveness of retrieving the information. [2] They proposed rule-based light-stemmer to find the stem from inflection words. And the light-stemmer was performed much better than suffix removal and more effective in retrieving the information. [3] Stemming with K-Mean Clustering technique, is used in IRS System, where the most likely documents are retrieved by ensuring the related words are mapped to common stem. [4] For improving the performance of IRS System, it clusters stemmed words with less computational steps. It provides better performance than before clustering. It improves the recall, the number of documents retrieved in response to a Query. [7] He developed derivational improving the performance of Odia IRS, by using recall factor of the number of document retrieved for a query in response. [11] He used Naïve Algorithm for finding the root word. It makes use of look-up table to relate root and the inflected words. It is a concept of Artificial Intelligence called Naive Search. It produces better performance with more number of relations in a look-up table. [6] He developed the stemmer for the language Urdu and Marathi using Frequency and Length based Stripping Algorithm.

The retrieval effectiveness of the IRS System is improved by using Stemmer.

3. Problem Statement

Words may be of derived or inflected form or affixed form or compound form, where the stemmer is developed for finding the key part of the word. It may be a root or stem of the word. Compound words (joined words) may be formed of concatenation of two or more words.

In this paper, it presents the compound forms framed as a noun followed by the verb. Words can be of noun followed by a verb eg., À¾É¢Î¾ø, noun followed by a noun eg., ¸¨¾ôÒò¾¸õ, etc., It requires that nouns and verbs are stored in the dictionary separately. It provides better accuracy when there are large numbers of verbs and nouns are stored in the dictionary.

System Architecture

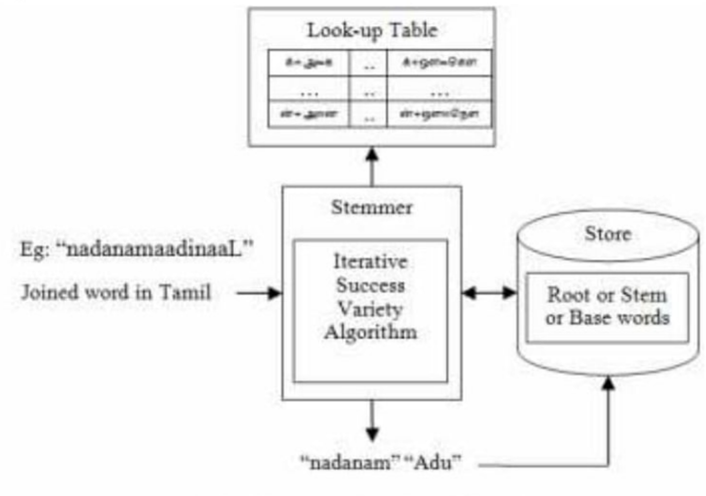


Fig.1 Stemmer for Joined Words

Description

In this context, reads an input compound word and splits it into two root word. It needs well-formed rules for the joined words which is framed by analyzing the words and gets stored. And stores the collected stem or root word in a Dictionary, so that it can be used to split the joined words into stem or root. It needs language-specific knowledge of word analysis, so that the key part can be identified from the compound form of words.

Stemmer

It reads the compound word and splits that word into root or stem word by applying rules.

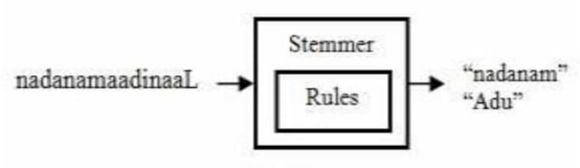


Fig.2 Stemmer

Algorithm: (Iterative Success Variety method)

1. Read the input string.
2. Collect the words (at least five) from the noun store which are closer to the input string and find the word length of the word which is maximum among those words.
3. Do the iterative success variety process for the word that matches at the most probable one.
   a. For getting the last character of the first word and the first character of the second word it makes use of look-up table.
   b. Eg., ¿¼ÉÁ¡ÊÉ¡û, it looks for the character "Á¡", and it replaces "Á¡" by "õ+¬", by looking up the table. So that we get the last character of the first word and the first character of the second word.
   c. And add it to the dictionary if it is a new one.
4. Do the next iteration by selecting the words (at least five from the verb store which are closer to the input string after splitting process and find the word length of the word which is maximum among those words) with character it finds with the look-up table.
5. Repeat until all the root words found.
6. And add second root or stem or base word to the dictionary if it is a new one.

Algorithm Description:

Sample Input-1: ¿¼ÉÁ¡ÊÉ¡û and a set of words beginning with ¿. (It requires a dictionary of root/stem/base words). Here those joined words are formed as noun followed by a verb.

Output: ¿¼Éõ, ¬Î

Iteration-1:

¿¼EA¡EE¡û

¿¼Eõ, ¿¡¼¸õ, ¿¼, ¿E, ¿¡û

Current character	Next Character
¿	{¼, ¡, E}
¿¼	{E}
¿¼E	{õ}
¿¼Eõ	{null}

Here it maintains a pointer to hold that the root word '¿¼Éõ' ends here (all the character matches except last character when compared to the root word '¿¼É_' '¿¼ÉÁ¡'). And it needs to look-up the table to get 'Á¡' to 'õ+¬' so that we get '¿¼Éõ'.

After the modification, next word starts with '¬'. So it picks the set of words from the dictionary which is beginning with '¬' and it is a verb selected from the store.

Iteration-2:

¬EE¡û

¬I ¬E ¬¨º

Current character	Next Character
¬	{I, E, ¨º}
¬E	{null}

It gives the second root word as "¬Î"

Sample Input-2: Åñ½Á¢Î and a set of words beginning with Å.

Iteration-1:

Åñ½Á¢Î

Åñ½õ, Å¡Éõ, Å¡¢¨º, ÅñÊ, ÅÚ¨Á

Current character	Next Character
Å	{ñ, ¡, ¡¢, Ú}
Åñ	{½, Ê}
Åñ½	{õ}
Åñ½õ	{null}

Output: Åñ½õ, þÎ

Here it maintains a pointer to hold that the root word 'Åñ½õ' ends here (all the character matches except last character when compared to the root word ' Åñ½_' 'Åñ½Á¢'). And it needs to look-up the table to get ' Á¢' to 'õ+þ' so that we get 'Åñ½õ'. After the modification, next word starts with 'þ'. So it picks the set of words from the dictionary which is beginning with 'þ' and it may be a verb selected from the store.

Iteration-2:

þI

þI, þE, þØ

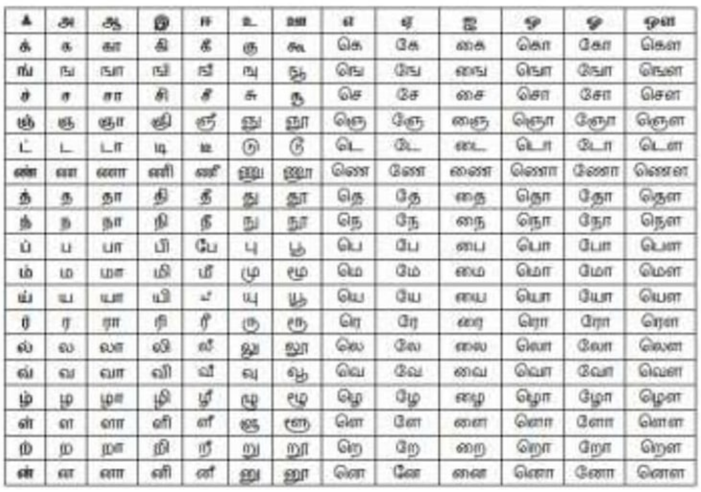
Current character	Next Character
þ	{I, E, Ø}
þI	{null}

It gives the second root word as "þÎ"

It stops finding the root/stem/base when it finds the pattern and needs to select the word, which is a verb. It needs to collect the verbs and nouns separately.

Look-up Table:

Table-1 Look-up Table.



ஃ	அ	ஆ	இ	ஈ	உ	ஊ	எ	ஏ	ஐ	ஒ	ஓ	ஔ
க்	க	கா	கி	கீ	கு	கூ	கெ	கே	கை	கொ	கோ	கௌ
ங்	ங	ஙா	ஙி	ஙீ	ஙு	ஙூ	ஙெ	ஙே	ஙை	ஙொ	ஙோ	ஙௌ
ச்	ச	சா	சி	சீ	சு	சூ	செ	சே	சை	சொ	சோ	சௌ
ஞ்	ஞ	ஞா	ஞி	ஞீ	ஞு	ஞூ	ஞெ	ஞே	ஞை	ஞொ	ஞோ	ஞௌ
ட்	ட	டா	டி	டீ	டு	டூ	டெ	டே	டை	டொ	டோ	டௌ
ண்	ண	ணா	ணி	ணீ	ணு	ணூ	ணெ	ணே	ணை	ணொ	ணோ	ணௌ
த்	த	தா	தி	தீ	து	தூ	தெ	தே	தை	தொ	தோ	தௌ
ந்	ந	நா	நி	நீ	நு	நூ	நெ	நே	நை	நொ	நோ	நௌ
ப்	ப	பா	பி	பே	பு	பூ	பெ	பே	பை	பொ	போ	பௌ
ம்	ம	மா	மி	மீ	மு	மூ	மெ	மே	மை	மொ	மோ	மௌ
ய்	ய	யா	யி	ீ	யு	யூ	யெ	யே	யை	யொ	யோ	யௌ
ர்	ர	ரா	ரி	ரீ	ரு	ரூ	ரெ	ரே	ரை	ரொ	ரோ	ரௌ
ல்	ல	லா	லி	லீ	லு	லூ	லெ	லே	லை	லொ	லோ	லௌ
வ்	வ	வா	வி	வீ	வு	வூ	வெ	வே	வை	வொ	வோ	வௌ
ழ்	ழ	ழா	ழி	ழீ	ழு	ழூ	ழெ	ழே	ழை	ழொ	ழோ	ழௌ
ள்	ள	ளா	ளி	ளீ	ளு	ளூ	ளெ	ளே	ளை	ளொ	ளோ	ளௌ
ற்	ற	றா	றி	றீ	று	றூ	றெ	றே	றை	றொ	றோ	றௌ
ன்	ன	னா	னி	னீ	னு	னூ	னெ	னே	னை	னொ	னோ	னௌ

Look-up table consist of the character which is to be replaced by a sequence of the characters, so that it gets the last character of the first word and the first character of the second root word.

Eg., ¿¼ÉÁ¡ÊÉ¡û, where 'Á¡' is replaced by 'õ+¬', so that it gets the last character 'õ' of the first word and the first character '¬' of the second root word.

4. Future and Conclusion

Stemmers are used to find the root or stem or base words of the given words. The stemmers are mainly used in other NLP applications like search engine or information retrieval system or in POS tagger etc. The accuracy of these applications depends on the accuracy of the stemmer it uses. For the joined words, stemmers are not available.
In this paper, it considers the joined words where the second word start with the last character of the word in some form of that character. And it considers those words as the noun followed by a verb. Likewise it may be used to find the root words for the joined words like noun followed by a noun, words formed with consonants between two words, etc.
This paper is submitted as an Article and which will be implemented later.

References

[1] Ubeeka Jain and Jasbir Kaur, "A Review on Text Chunker for Punjabi Language", International Journal of Advanced Research in Computer and Communication Engineering, Vol.4, Issue.7, pp.116-119, July-2015.

[2] M.Kasthuri and S.Britto Ramesh Kumar, "An Improved Rule-based Iterative Affix Stripping Stemmer for Tamil Language using K-mean Clustering", International Journal of Computer Applications(0975-8887) Vol.94, No.13, P.No:902-908, May-2014.

[3] R.Vijay Lakshmi and Dr.S.Britto Ramesh Kumar, "Literature Review: Stemming Algorithm for Indian and non-Indian Languages", International Journal of Advanced Computer Science & Technology, ISSN: 2347-8446, Vol.2, Issue.3, June-Sept-2014.

[4] M.Thangarasu and Dr.R.Manavalan, "Stemmers for Tamil Language: Performance Analysis", International Journal of Computer Science & Engineering Technology, ISSN: 2229-3345, Vol.4, No.7, P.No:902-908, July-2013.

[5] M.Thangarasu et. al., and Dr.R.Manavalan, "Design and Development of Stemmers for Tamil Language: Cluster Analysis", International Journal of Advanced Research in Computer Science & Software Engineering, ISSN: 2277-128X, Vol.3, July-2013.

[6] Dhabal Prasad Sethi, "Design of Light Weight Stemmer for Odia Derivational Suffixes", International Journal of Advanced Research in Computer & Communication Engineering, ISSN: 2319-5940, Vol.2, No.12, December-2013.

[7] Mohd. Shahid Husain, "An Unsupervised Approach to develop Stemmer", International Journal of Natural Language Computing (IJNLC), Vol.16, No.2, August-2012.

[8] G. Akilan.R Prof.E.R.Naganathan, "Morphological Analyzer for Classical Tamil text – A Rule-based Approach", International journal of Innovative Science, Engineering & Technology, Vol.1, Issue.5, July-2014.

[9] Puneet Thapar, "A Hybrid Approach used for stem Punjabi Words", International Journal of Computer Science and Mobile Computing, Vol.3, Issue.11, pp.1-9, November-2014.

[10] Willett.P (2006). "The Porter Stemming Algorithm, then and now", Program: Electronic Library and Information System, 40(3), pp.219-223.

[11] Thomas Lehman, "A grammar of modern Tamil", Pondicherry Institute of Linguistic and Culture.

[12] Weblink:www.languagesgulper.com/eng/Tamil.html

[13] Weblink://snowball.tartarus.org/algorithms/porter/stemmer.html Vol.14, Issue.3, pp.130-137, July-1980.

[14] http://www.slideshare.net/siva518/tamil-grammerineasyenglish

Author Profile

D.Umamageswari, studying II-Year M.E(CSE), in Sri Venkateswara College of Engineering, Pennalur, and I completed my B.E(CSE) 1998-2001, in Periyar Maniammai College of Technology for Women, Tanjavur.