Volume 3, Issue 7, July 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Design and Development of Stemmer for Tamil Language: Cluster Analysis

M.Thangarasu*
Department of Computer Science and Application
K.S.Rangasamy College of Arts and Science
Tiruchengode, India

Dr.R.Manavalan
Head, Department of Computer Science and Application
K.S.Rangasamy College of Arts and Science
Tiruchengode, India

Abstract*— Stemming is a technique to transform different inflections and derivations of the same word to one root word "stem". Stemming means removal of both prefix and suffix. It is useful in many areas of computational linguistics and information retrieval work for improving their performance. This paper proposes improved light stemming algorithm for getting stemmed Tamil word with less computational steps. Further K-Means clustering algorithm utilized to cluster the stemmed Tamil Words in order to improve the performance of Tamil language Information Retrieval System. The experimental results clearly shows that the words stemmed after clustering gives better result compared to words stemmed before clustering. This may increases performance of the Tamil language Information Retrieval System.

***Keywords*— Tamil morphology, Tamil stemmer, Light stemmer, Improved stemmer, Natural Language Processing**

I. INTRODUCTION

A process that attempts to map a derived form of word to its root is referred as stemmer. For example words such as removes, removing and removed all will reduce to stem remove. Stemming plays an important role in Information Retrieval System (IRS) for improving their performance [1] for example when user enter query word stemming, user most likely wants to retrieve documents containing the terms stemmer and stemmed as well. Thus using stemmer improves recall (i.e.) the number of documents retrieved in response to a query. Since many terms are mapped to one. Stemmer serves to decrease the size of the index files in the IRS. This is especially true in case of a morphologically rich language Tamil, where a single word may take many forms. The aim is to ensure that related words map to common stem. So many stemmer algorithm have been proposed and evaluated for various Indian languages such as Hindi [1], Marathi [2], Bengali [4], Urdu [5], Malayalam [6] etc. This paper proposes an improved light stemmer for Tamil language with cluster technique. An overview of the proposed model projected Fig. 1.

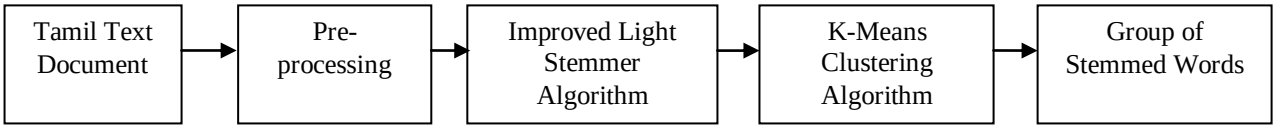

Fig. 1 Overview of Proposed Model for Tamil Stemmer Algorithm

This paper organized as follows: Section 2 describes the related work done in development of stemmer in Indian languages. Section 3 is a detailed description about the Tamil language. Pre-processing technique discussed in section 4. Section 5 presents the proposed system improved light stemmer algorithm and K-Means clustering technique for grouping the stemmed Tamil words. Sections 6 illustrate an experimental analysis and proposed work concluded in Section 7.

II. RELATED WORK

Stemming has been the most widely applied morphological technique for information retrieval. With stemming, the searcher does not need to worry about the correct truncation point of search keys. Earlier, Stemmer was primarily developed for English, but later due to the corpus growth of languages other than English, there was an increased demand from the research community to develop stemmers for other languages too. In case of Indian languages, the earliest work reported by Ramanathan and Rao [1] (2003) to perform longest match stripping for building a Hindi stemmer. Juhi Ameta et al. developed a light stemmer for Gujarati language [2] in 2011 for removing inflectional and derivational endings in order to reduce word forms to common stem. Slowly investigation started for other language such as Bengali [4], Urdu [5], Malayalam [6] and Punjabi [3]. Steinbach and et al. developed a comparison of document clustering techniques [15] for improving the English document clustering technique and used it for information retrieval system. However, this section expresses the research experience in developing Tamil stemmer with cluster technique. The most common approaches used for developing stemmers are Brute force and Affix strip. Among all the existing approaches,

this research tries to implement an improved light stemmer for Tamil language for inherent to develop a stemmer algorithm in an easier and faster way.

III. TAMIL LANGUAGE

Tamil is a Dravidian language, mainly spoken predominantly by Tamil people of Indian subcontinent. Tamil words have more derivational forms than English words. Tamil word contains constituent parts: a stem, which can be thought of as responsible for the nuclear meaning of verb, attached to which may be zero or more derivational prefix and zero or one suffix, which together form a word. Tamil is a morphologically rich language resulting in its relatively high inflectional forms. Normally most of the Tamil words have more than one morphological suffix. The number of suffix is ranging from 3 to 13. The description with example nouns, compound nouns and verbs are given in the following sequel.

A. Noun

Tamil has an extensive case system. Root nouns can assume eight different morphological shapes depending on their roles in a sentence. Singular and plural forms are also distinguished through inflections. Suffixes are attached to stem word of the noun. TABLE I shows example for different Tamil language stem noun.

TABLE I: NOUNS IN TAMIL LANGUAGE

Singular	PeN(¦Àñ)/ girl	maram(ÁÃõ) /tree
Oblique stem	PeN(¦Àñ)-	maram(ÁÃõ) -
Nominative stem	PeN(¦Àñ)	maram(ÁÃõ)
Accusative stem	PeN(¦Àñ)-ai(³)	maram(ÁÃõ) -ai
Dative stem	PeN(¦Àñ)-ukku(¯ìÌ)	maram(ÁÃõ) -ukku(¯ìÌ)
Sociative stem	PeN(¦Àñ)-odu(µÎ)	maram(ÁÃõ) -odu(µÎ)
Genitive stem	PeN(¦Àñ)-udaiya(¯¨¼Â)	maram(ÁÃõ) -udaiya(¯¨¼Â)
Instrumental stem	PeN(¦Àñ)–aal(¬ø)	maram(ÁÃõ) -aal(¬ø)
Locative stem	PeN(¦Àñ)-idam(þ¼õ)	maram(ÁÃõ) -marath(þø)
Ablative stem	PeN(¦Àñ)–idamirunthu(þ¼Á¢ÕóÐ)	maram(ÁÃõ) -ilirunthu(þÄ¢ÕóÐ)
Vocative stem	PeN(¦Àñ)-e(±)	maram(ÁÃõ) -e(±)
Plural	PeNgaL(¦Àñ¸û)/ girls	marangal(ÁÃí¸û)/ trees
Oblique stem	PeNgaL(¦Àñ¸û)-	marangal(ÁÃí¸û)-
Nominative stem	PeNgaL(¦Àñ¸û)	marangal(ÁÃí¸û)
Accusative stem	PeNgaL(¦Àñ¸û)-ai(³)	marangal(ÁÃí¸û)-ai(³)
Dative stem	PeNgaL(¦Àñ¸û)-ukku(¯ìÌ)	marangal(ÁÃí¸û)-ukku(¯ìÌ)
Sociative stem	PeNgaL(¦Àñ¸û)-odu(µÎ)	marangal(ÁÃí¸û)-odu(µÎ)
Genitive stem	PeNgaL(¦Àñ¸û)-udaiya(¯¨¼Â)	marangal(ÁÃí¸û)-udaiya(¯¨¼Â)
Instrumental stem	PeNgaL(¦Àñ¸û)-aal(¬ø)	marangal(ÁÃí¸û)-aal(¬ø)
Locative stem	PeNgaL(¦Àñ¸û)-idam(þ¼õ)	marangal(ÁÃí¸û)-il(þø)
Ablative stem	PeNgaL(¦Àñ¸û)-idamirunthu(þ¼Á¢ÕóÐ)	marangal(ÁÃí¸û)-ilirunthu(þÄ¢ÕóÐ)
Vocative stem	PeNgaL(¦Àñ¸û)-e(±)	marangal(ÁÃí¸û)-e(±)

B. Compound Nouns

A noun also occurs in various compound forms as well. It can be made up of several units where each unit expresses a particular grammatical meaning. The Tamil noun, "pAdikkoNdurunhthavanai (À¡Êì¦¸ñÎÕó¾Å¨É)", which translates as, "the male who was singing", gives information on tense, number, gender, person and case. This noun is actually derived from the full non-infinite verb, "pAdikkoNdu (À¡Êì¦¸ñÎ)", which means, "singing". In English, deriving nouns from verbs is seen too. The full finite verb, "sing", for instance could be changed into a noun by adding the suffix "er" to its stem, so that it becomes "singer". But, while Tamil is an agglutinating language, English is not.

C. Verbs

Tamil verbs may be main or auxiliary. They also exist in finite and non-finite forms just as in English. Tamil finite verbs however give much more grammatical information than English finite verbs do, in that they mark mood, tense, number, person, gender, case etc… In the Table II and TABLE III below, can observe the different finite and non-finite morphological construction for the verb "padi(ÀÊ)" [study].

TABLE II : VERBS IN TAMIL LANGUAGE

	Past	**Present**	**Future**	**Future-Neg**
I singular	Padi(ÀÊ)-ththEn(ò§¾ý)	Padi(ÀÊ)-kkiREn(ì¸¢§Èý)	Padi(ÀÊ)-ppEn(ô§Àý)	Padi(ÀÊ)-kkamaattEn(ì¸Á¡ð§¼ý)
II singular	Padi(ÀÊ)- ththAi (ò¾¡ö)	Padi(ÀÊ)-kkiRaai(ì¸¢È¡ö)	Padi(ÀÊ)-ppaai(ôÀ¡ö)	Padi(ÀÊ)-kkamataai(ì¸Á¼¡ö)
III singular male	Padi(ÀÊ)-ththaan(ò¾¡ý)	Padi(ÀÊ)-kkiRaan(ì¸¢È¡ý)	Padi(ÀÊ)-ppaan(ôÀ¡ý)	Padi(ÀÊ)-kkamaattaan(ì¸Á¡ð¼¡ý)
III singular female	Padi(ÀÊ)- ththaaL (ò¾¡û)	Padi(ÀÊ)-kkiRaaL(ì¸¢È¡û)	Padi(ÀÊ)-papal(ôÀ¡û)	Padi(ÀÊ)-kkamaattaaL(ì¸Á¡ð¼¡û)
III singular hon	Padi(ÀÊ)-ththaar(ò¾¡÷)	Padi(ÀÊ)-kkiRaar(ì¸¢È¡÷)	Padi(ÀÊ)-ppaar(ôÀ¡÷)	Padi(ÀÊ)-kkamaatdaar(ì¸Á¡ð¼¡÷)
III singular inan	Padi(ÀÊ)-ththathu(ò¾Ð)	Padi(ÀÊ)-kkiRathu(ì¸¢ÈÐ)	Padi(ÀÊ)-kkum(ìÌõ)	Padi(ÀÊ)-kkaathu(ì¸¡Ð)
I plural	Padi(ÀÊ)-tthOm(ð§¾¡õ)	Padi(ÀÊ)-kkiROm(ì¸¢§È¡õ)	Padi(ÀÊ)-pPOm(ô§À¡õ)	Padi(ÀÊ)-kkamaatTOm (ì¸Á¡ð§¼¡õ)
II plural	Padi(ÀÊ)-ththIrkaL (ò¾£÷¸û)	Padi(ÀÊ)-kkiRIrkaL(ì¸¢È£÷¸û)	Padi(ÀÊ)-ppIRkaL(ôÀ£ü¸û)	Padi-(ÀÊ)-kkamaattaarkaL(ì¸Á¡ðË÷¸û)
III plural an	Padi(ÀÊ)-ththaarkaL (ò¾¡÷¸û)	Padi(ÀÊ)-kkiRaarkaL(ì¸¢È¡÷¸û)	Padi(ÀÊ)-ppaarKaL(ôÀ¡÷¸û)	Padi(ÀÊ)-kkamaattaarkaL(ì¸Á¡ð¼¡÷¸û)
III plural inan	Padi(ÀÊ)-ththana(ò¾É)	Padi(ÀÊ)-kkinRana(ì¸¢ýÈÉ)	Padi(ÀÊ)-kkum(ìÌõ)	Padi(ÀÊ)-kkaathu(ì¸¡Ð)

TABLE III : NON-FINITE VERB IN TAMIL LANGUAGE

Conjunctive	Padi(ÀÊ)-thu (Ð)
Infinitive	Padi(ÀÊ)-kka(ì¸)
Neg. verbal participle	Padi(ÀÊ)-kkaamal(ì¸¡Áø)
Conditional	Padi(ÀÊ)-thaal(¾¡ø)
Neg. Conditional	Padi(ÀÊ)-kkaanittaal(ì¸É¢ð¼ø)
Neg.relative participle	Padi(ÀÊ)-kkaatha(ì¸¡¾)
Neg. verbal noun	Padi(ÀÊ)-kkaathathu(ì¸¡¾Ð)
Deverbal nouns	Padi-(ÀÊ)thal(¾¡ø);padi(ÀÊ)-ppu(ôÒ); padi(ÀÊ)-kkai(ì¨¸)

IV. PRE-PROCESSING

Pre-processing has been traditional in setting up Information Retrieval System (IRS) to discard the very commonest words of a language - the stop words - during indexing. A more modern approach is to index everything, which greatly assists searching for phrases for example. Stop words can then still be eliminated from the query as an optional style of retrieval. In either case, a list of stop words for a language is useful. Getting a list of stop words can be done by sorting a vocabulary of a text corpus for a language by frequency, and going down the list picking off words to be discarded. The stop word list connects in various ways with the stemming algorithm.

The stemming algorithm can itself be used to detect and remove stop words. One would add into the irregular forms something like this,

"am/is/are/be/being/been/" /* BE */

"have/has/having/had/" /* HAD */
"do/does/doing/did/" /* DID */
So that the words `am', `is' etc. map to the null string (or some other easily recognized value). Alternatively, stop words could be removed before the stemming algorithm is applied, or after the stemming algorithm is applied. In this latter case, the words to be removed must themselves have gone through the stemmer, and the number of distinct forms will be greatly reduced as a result. In Tamil for example, the four forms
PaaduvaarkaL(À¡ÎÅ¡÷¸û),PaadukiRaarkaL(À¡Î¸¢È¡÷¸û), paadinaarkaL(À¡ÊÉ¡÷¸û),
PaadikkoNdirukkiRaarkaL(À¡Êì¸¡ñÊÕì¸¢È¡÷¸û). Meaning 'that' all stem to
paaduthal (À¡Î¾ø)

In the data directory, each language represented in the stemming section has, in addition to a large test vocabulary, a useful stop word list in both source and stemmed form. The source form, in the file stop source, is carefully annotated, and the derived file, stop words, contains an equivalent list of sorted, stemmed, stop words.

Almost in all languages, words appear in several inflected forms. For example, in English, the verb 'to run' may appear as 'run', 'ran', and 'running'. The base form, 'run', that one might look up in a dictionary, is called the lemma for the word. In Tamil language the stem/root of a word is known as 'Ver'. The combination of the base form with the part of a speech is often called the lexeme of the word. A stemming algorithm is a process of linguistic normalization, in which the variant forms of a word are reduced to a common form, called the root (stem). This work mainly deals with the problem of plural resolutions in Tamil language.

V. THE IMPROVED LIGHT STEMMER ALGORITHM FOR TAMIL LANGUAGE

Light stemmer works by truncating all possible suffixes form and produce finite verb. Light stemming is used to find the representative indexing form of given word by the application of truncation of suffixes [10]. The core objective of light stemmer is to preserve the word meaning intact and so that it increases the retrieval performance of an IR system. The proposed improved light stemming algorithm in this system defines an algorithm (removes suffixes recursively and a single prefix non-recursively), that is called SP. The same defined affixation terms list were used but with a modified execution step via Suffix Prefix Suffix truncating process, that algorithm is called SPS.

Notice that most of Tamil words use (Thiru, Thirumathi) prefix as a declarative term (e.g., Thiru. Dr.A.P.J.AbdulKalam) therefore, proposed two new major categories in classifying of the designed algorithms; Without-Thiru (WOTH the stemmer accepts the non-stemmed words after removing the prefixed Thiru) and With Thiru (WTH stemmer acquires the whole non-stemmed word).This proposed work applies improved light stemming concept to replace plural terms, adjectives and tense words. The flowchart and algorithm of proposed improved light stemmer projected in Fig. 2 and Fig. 3.

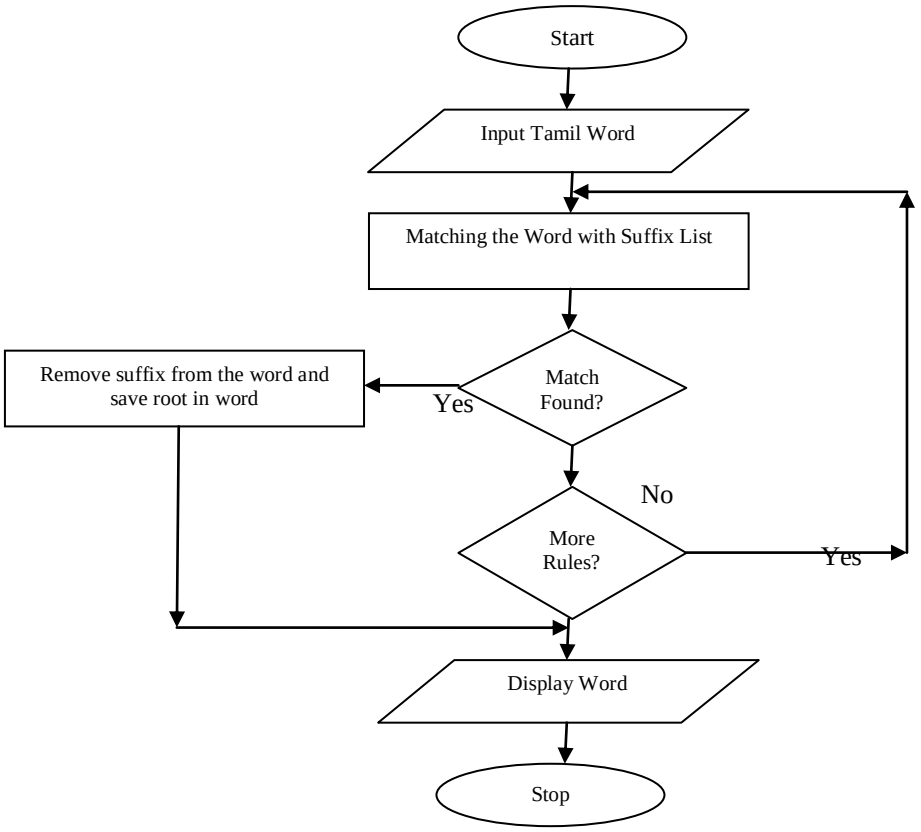

Fig. 2 Improved Light Stemmer Algorithm for Tamil Language

Improved Light stemmer Algorithm for Tamil Language
Input: List of preprocessed Tamil words Output: Stemmed(Root) words
Step 1: Eliminate the entire complex plural. Step 2: After the plural word is converted into singular word, during the iteration, the word is also checked for adjective; if it is found, then its equivalent verb is substituted. Example, the term 'diya(ÊÂ)' in Odiya(µÊÂ) will be changed to 'du(Î)' and the word is changed to 'Oodu(µÎ)'. Step 3: After the adjectives are converted to main word, the tenses are eliminated such that Paadiya(À¡ÊÂ), Paadukinra(À¡Î¸¢ýÈ) and Paadum(À¡Îõ) will be changed to Paadu(À¡Î). Step 4: According to the identified suffix, the next possible suffix list is generated and add more rules. Step 5: The Light algorithms are used for plural to singular conversion, and for adjective and tense words to main verbs conversion.

Fig. 3 Improved Light Stemmer Algorithm for Tamil Language

A. *K-Means Clustering Algorithm for Stemmed Tamil Documents*

The new system provides clustering in a natural way to reduce the large dimensionality of the document vector space and helps in efficient relevant in Tamil document extraction. This paper presents noval algorithm for stemming Tamil language texts and apply K-Means algorithm to cluster the stemmed Tamil words. The detailed description of K-Means clusters are given in the following sequel.

In statistics and machine learning, K-Means clustering is a method of cluster analysis which aims to partition *n* Stemmed Tamil words into *k* clusters in which each Stemmed Tamil words belongs to the cluster with the nearest mean. Consider the given set of Stemmed Tamil words (x_1, $x_{2\ldots}$ x_n), where each stemmed Tamil word is a *d*-dimensional real vector, K-Means clustering aims to partition the *n* stemmed Tamil word into *k* sets ($k < n$) $\mathbf{S} = \{S_1, S_2 \ldots S_K\}$ so as to minimize the within-cluster sum of squares (WCSS):

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

Where, $\boldsymbol{\mu}_i$ is the mean of points in S_i. The K-Means cluster algorithm projected in Fig. 4.

K-Means Cluster Algorithm for Tamil Document
Input: List of Stemmed Tamil words from Improved Light Stemmer Output: Group of Stemmed(Root) words
Step 1: Specify the number of cluster K. Step 2: Determine the centriod from a given input. Step 3: Find the distance of Stemmed Tamil Words using Eculidean distance metrics. Step 4: If there is no minimum distance between the Stemmed Tamil Words and no stemmed Tamil words movement happen between the cluster Step 5: Otherwise grouping takes place based on minimum distance between the stemmed Tamil words in cluster Step 6: Repeat the process from Step 2, unit form until form cluster.

Fig. 4 K-Means clustering Algorithm for Tamil Document

VI. EXPERIMENTAL ANALYSIS

The goal of experimental analysis is to calculate the accuracy of the proposed stemmer system can achieve. Parameter that can be used for evaluating Tamil improved light stemmer algorithm in this proposed model is number of cluster produced by it. The detailed description of dataset used for experiment and analysis of experiment, results and their discussions are mentioned below.

A. *Test Data*

For finding the efficiency of the proposed system, it should be tested by the proposed system by giving the input as Tamil documents. The documents have been taken from the net and given input of 200 Tamil documents which approximately consist of 7,000 words. Text data has to go through several pre-processing stages in order to obtain clear and unambiguous data before stemming and cluster analysis. 200 documents considered for evaluating the proposed system that are simple enough for stemming from the textual analysis point of view. The documents were pre-processed, and applied improved light stemming to stem the article.

B. *Experimental Result*

The K-Means clustering algorithm is used to cluster the stemmed Tamil documents and the result presented in TABLE IV. The text files are taken for clustering. First the words in item set table are taken and searched in text files.

The text file names, which contain the particular word, are stored in a vector. Likewise all the text file names that contain the words are stored in separate vectors. Then the intersection is made in such a way that the text files contain two words. The iteration continues for three words, four words and so on. The test dataset can be evaluated according to the number of Tamil documents. When 100 documents are taken the words stemmed before clustering reaches 483 stemmed words and words stemmed after clustering reaches up to 526. Likewise when 200 documents are taken the words stemmed clustering reaches 617 stemmed words and words stemmed after clustering reaches 684. The result shows that the number of words stemmed after clustering is better than number of words before clustering.

TABLE IV: TEST DATA

Number of documents	Paragraphs taken	Words stemmed before clustering	Words stemmed after clustering
100	119	483	526
120	135	509	515
140	154	547	623
160	170	580	670
180	196	598	669
200	218	617	684

Fig. 5 shows the performance analysis of the proposed system. The computational results from the TABLE IV and Fig. 5 clearly proved that improved light stemmer algorithm is achieving better performance in words stemmed after clustering compare than the words stemmed before clustering.

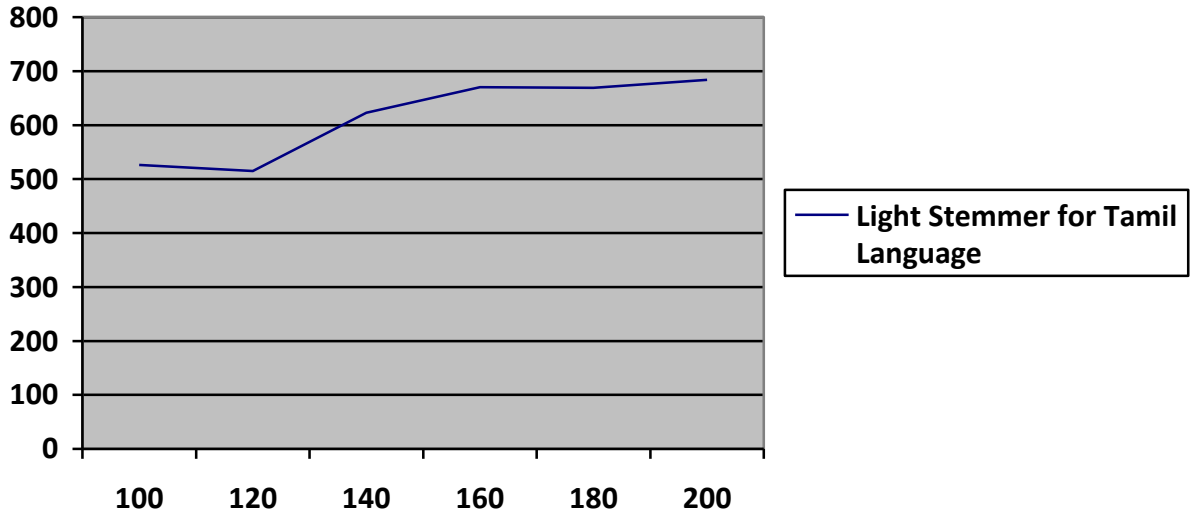

Fig. 5 Performance analyses for Tamil Stemmer Algorithm with Cluster

VII. CONCLUSION

Stemming plays vital role Tamil information retrieval, compared with all other languages. The stemming effects very large, compared to that found in review on other stemming algorithm. Improved light stemmer is best one since it removes stop words, definite documents, and ("and") from the beginning of words and suffixes from the ends of words. Note, however, that Tamil text contains many definite articles that one could obtain the claimed 99% tokenization accuracy simply by removing Thiru from the beginnings of words. Improved light stemming is robust. It does not require complete sentences and it does not try to handle every single case. Improved light stemmer algorithm is suitable for IRS of Tamil language. Since it perform efficiently for morphological rich language Tamil.

REFERENCES

[1] A.Ramanathan and 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 Computational linguistics for South Asian Language (Budapest, April) workshop, 2003.

[2] Juhi Ameta, Nisheeth Joshi and Iti Mathur, 2011, "A Lightweight Stemmer for Gujarati," 46th Annual National Convention of Computer Society of India. Organized by Computer Society of India Gujarat Chapter. Sponsored by Computer Society of India and Department of Science and Technology, Govt. of Gujarat and IEEE Gujarat Section.

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

[4] Khan. 2007. "A light weight stemmer for Bengali and its Use in spelling Checker," Proc. 1st Intl. Conf. on Digital Comm. and Computer Applications (DCCA07), Irbid, Jordan, March 19-23.

[5] "A Light Weight Stemmer for Urdu Language: A Scarce Resourced Language," Sajjad Ahmad Khan1, Waqas Anwar1, Usama Ijaz Bajwa1, Xuan Wang, Proceedings of the 3rd Workshop on South and Southeast Asian Natural Language Processing (SANLP), pages 69–78,COLING 2012, Mumbai, December 2012.

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

[7] Malayalam Stemmer - Computational Linguistic Research Group, nlp.au- kbc.org, Malayalam Stemmer.

[8] Madhavi Ganapathiraju and Levin Lori, TelMore: "Morphological Generator for Telugu Nouns and verbs," In the proceedings of Second International Conference on Universal Digital Library Alexandria, Egypt, November 17-19, 2006.

[9] D. Freitag, "Morphology induction from term clusters," In Proceedings of the ninth conference on computational natural language learning (CoNLL), pp. 128–135, 2005.

[10] Imed Al-Sughaiyer, Ibrahim Al-Kharashi. (2004). "Arabic morphological analysis techniques: a comprehensive survey," Journal of the American Society for Information Science and Technology, 55(3):189 – 213.

[11] M.F. Porter. 1980. An algorithm for suffix stripping Program, 14(3):130–137.

[12] Rajendran, S., Arulmozi, S., Ramesh Kumar, Viswanathan, S. 2001. "Computational morphology of verbal complex," Paper read in Conference at Dravidan University, Kuppam December 26-29, 2001.

[13] R. C. Dubes and A. K. Jain. Algorithms for Clustering Data. Prentice Hall, 1988.

[14] M.Thangarasu and Dr.R.Manavalan, "Stemmers for Tamil Language: Performance Analyses," International Journal for Computer Science & Engineering Technology, Vol. 4 No. 07 Jul 2013, ISSN: 2229-3345, 902-908.

[15] "A Comparison of Document Clustering Techniques," Michael Steinbach, George Karypis, Vipin Kumar.

