An Improved Rule based Iterative Affix Stripping Stemmer for Tamil Language using K-Mean Clustering

M.Kasthuri,
Asst.Professor, Dept. of Com.Science,
Bishop Heber College (Autonomous),
Tiruchirappalli, Tamil Nadu, India

S.Britto Ramesh Kumar,
Asst.Professor, Dept. of Com.Science,
St.Joseph's College (Autonomous),
Tiruchirappalli, Tamil Nadu, India

ABSTRACT
Stemming is an important step in many of the Information Retrieval (IR) and Natural Language Processing (NLP) tasks. Stemming is usually done by removing any attached suffixes and prefixes (affixes) from index terms before the actual assignment of the term to the index. Stemming is a pre-processing step in Text Mining applications and basic requirement for many areas such as computational linguistics and information retrieval work for improving their recall performance. This paper proposes improved rule based iterative affix stripping 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 and Extraction. The experimental analysis clearly shows that the words stemmed after clustering gives better result compared to words stemmed before clustering.

Keywords
Tamil morphology; Transliteration; Tamil stemmer; Improved affix stemmer; Natural Language Processing

1. INTRODUCTION
A process that attempts to map a derived form of word to its root is referred as stemmer. For example words such as tests, tested and testing all will reduce to stem word "test". Stemming plays an important role in Information Retrieval System for improving their performance [1]. For example when the user enters the query word computing, user most likely wants to retrieve documents containing the terms computer and computation as well. Thus using stemmer user can improves recall performance reducing the size of the index as user need not index all the morphological variants of a word. Since many terms are mapped to one. This is especially true in case of a morphologically rich language like Tamil, where a single word may take many forms. The aim of the stemming algorithm is to ensure that related words are mapped to common stem. Stemmers for different languages have been developed and evaluated for various Indian languages such as Hindi, Gujarathi, Punjabi, Bengali, Urdu, Marathi, Malayalam, Kannada etc [1-7] in the recent years. This paper proposes an improved iterative rule based affix stripping stemmer for Tamil language with K-Mean cluster technique. An overview of the proposed model is projected in Figure 1.

2. RELATED WORK
Stemmer was primarily developed for English Language; there was an increased demand from the research community to develop stemmers for other languages. But such studies on Indian languages are quite limited. The earliest work reported by Ramanathan and Rao [1] to perform longest match stripping for building a Hindi stemmer. Juhi Ameta et al. developed a light stemmer for Gujarathi language [2] in 2011 for removing inflectional and derivational endings. Then similar research work had started for other language such as Bengali [4], Urdu [5], Malayalam [7] and Punjabi [3]. There is a paper published by Vivek Anandan Ramachandran and Ilango Krishnamurthi [15] on an iterative suffix stripping stemming algorithm for Tamil. Steinbach and et al. developed a comparison of document clustering techniques [17] for improving the English document clustering technique and used it for information retrieval system. The work reported by M.Thangaraju et.al., and Dr.R.Manavalan in 2013 to perform suffix stripping stemming with clustering analysis [16, 17]. However, this section expresses the research experience in developing improved iterative rule based affix stripping Tamil stemmer with cluster technique.

3. 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 consists of a stem word attached to zero or more derivational prefix and zero or one suffix, which together form a word. Tamil is a morphologically rich language so Tamil Language has very 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. Tamil is the agglutinative language. One or more affixes are attached to the Tamil lexical root word. Most of the Tamil words affixes are suffixes. Suffixes of the Tamil Language can be derivational suffixes or inflectional suffixes. Derivational suffixes are either changes the part of speech of the word or its meaning. Inflectional suffixes are attached at the end of the root word. Proposed stemming algorithm for Tamil is used to strip extra constituents' available at prefixes and suffixes, and map them to a stem corresponding to the root word.

4. PRE-PROCESSING
Pre-processing has been traditional in setting up Information Retrieval System (IRS) to discard the stop words during indexing. 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. Stop words could be removed before the stemming algorithm is applied. A stemming algorithm is a process of linguistic normalization to strip unwanted constituents available at prefix or suffix of the stem word. This research work mainly deals with the problem of plural resolutions in Tamil language.

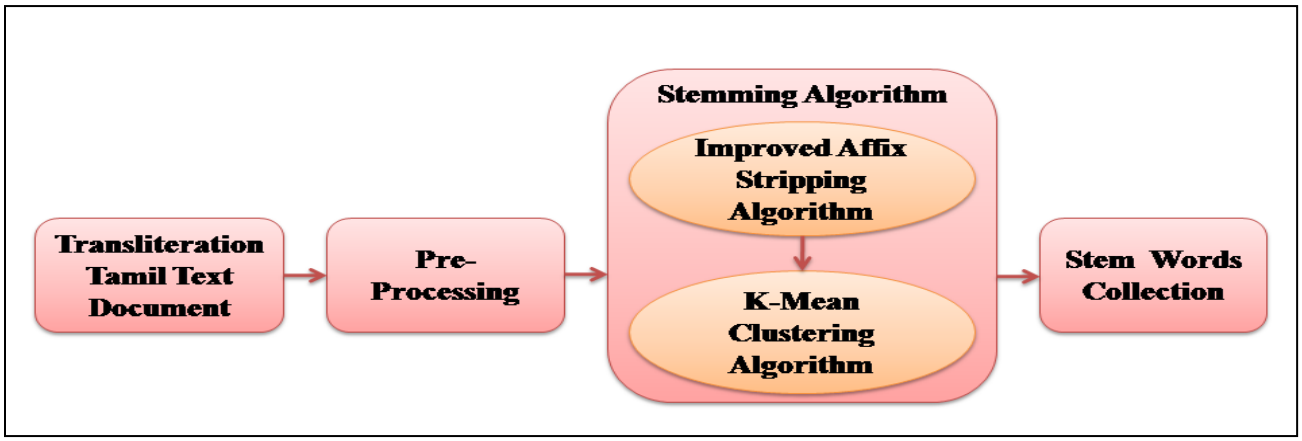

Fig 1: Proposed Model for Tamil Stemmer Algorithm

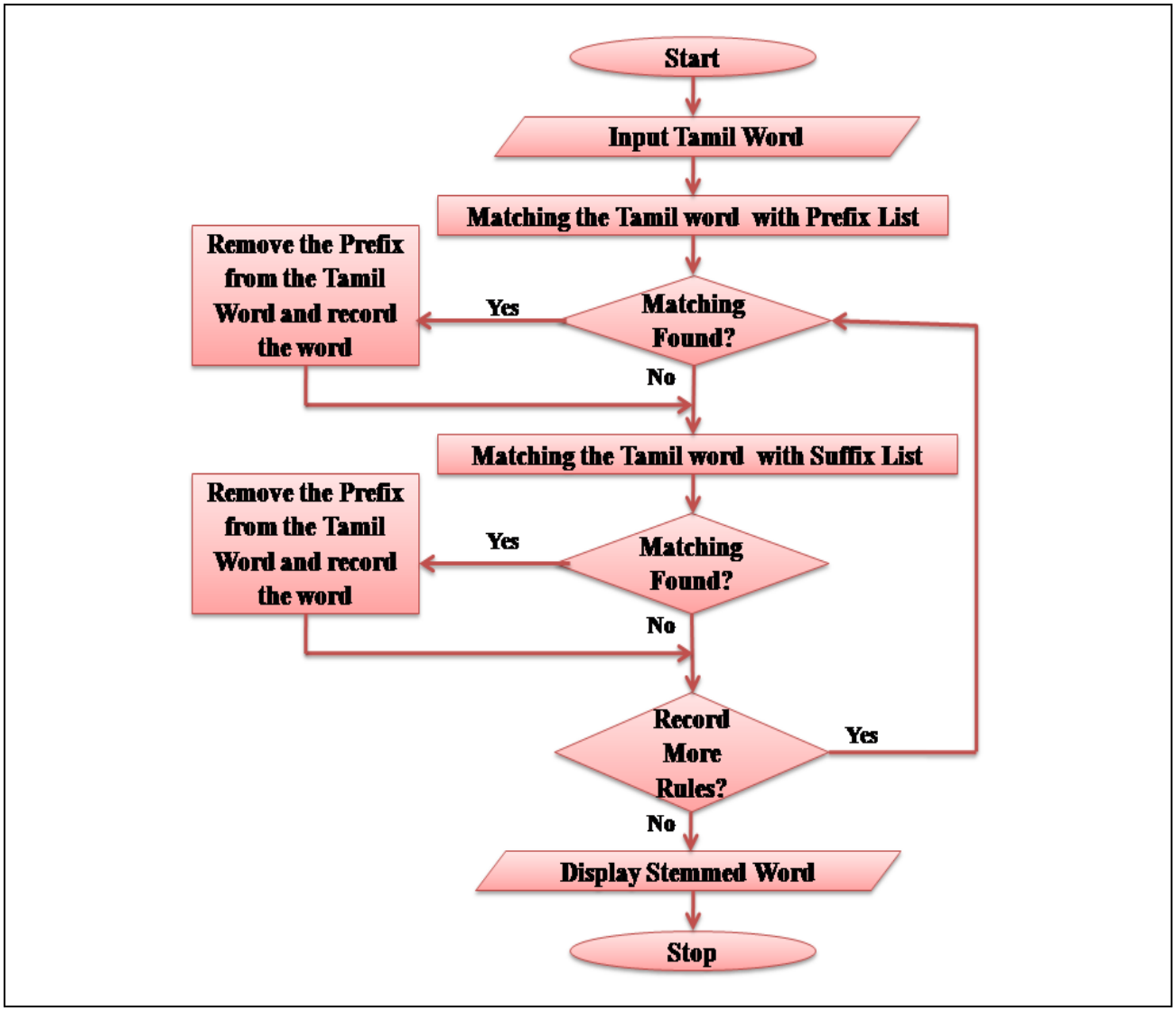

Fig 2: Improved Iterative Rule Based Affix Stripping Stemmer Algorithm for Tamil Language

(4.1) Example:

பேசுவார்கள், பேசுகிறார்கள், பேசினார்கள் and பேசிகொண்டிருக்கிறார்கள் all are stemmed to பேசுதல்.

5. IMPROVED AFFIX STRIPPING STEMMER ALGORITHM FOR TAMIL LANGUAGE

Affix removal algorithms remove suffixes and/or prefixes from terms leaving a stem. The flowchart and algorithm of proposed improved iterative affix stripping stemming algorithm is projected in Figure 2.

5.1 Prefix Removal

There are two routines in the algorithm to handle prefixes. First one is for handling the prefix in the questions. Eg. (எக்காலம்) (which period?) - (காலம்). And another one is for removing the pronoun prefixes, அ, இ and உ. Eg. (இக்காலம்) (this period) - (காலம்). After removing the prefixes another routine handles fixing the start of the word. The above prefixes introduce (வ்) when the root word starts with a vowel (வ) in the start of the word cannot combine with certain vowels. In such cases this routine substitutes with appropriate vowel as the starting. Notice that most of Tamil words use (திரு, திருமதி) prefix as a declarative term (e.g: திரு. Dr.A.P.J.அப்துல்கலாம்) therefore, proposed two new major categories in classifying of the designed algorithms; Without-Thiru (stemmer accepts the non-stemmed words after removing the prefixed Thiru) and With-Thiru (stemmer acquires the whole non-stemmed word). The improved light stemming algorithm is used to remove suffixes recursively and a single prefix non-recursively. The same defined affixation terms list were used but with the improved affix stripping algorithm, execution step via Suffix Prefix Suffix truncating process. So this algorithm is called SPS.

5.2 Fixing the ending

Only because of the following reasons suffixes are attached to the root word in Tamil

1. New letters are introduced
2. Some letters are removed
3. The letters are transformed
4. Joins naturally without addition/removal

Ending routine tries to handle these modifications before the next suffix removal routine is called. If the join had caused new letters to be introduced, this routine removes it. For example வல்லினம் consonants appear as conjunctions in many cases. A normal word will not end with a வல்லினம் consonant.

(5.2.1) Example:

Original word ➔ Suffix Stripped ➔ வல்லினம் consonant removed

நடக்க ➔ நடக் ➔ நட

5.3 Suffix removal

The improved iterative affix stripping stemming algorithm handles different types of suffixes such as Question Suffixes, Conjunction suffix, Common words, Case Suffixes, Plural Suffix, Imperative Suffixes and Tense Suffixes.

5.3.1 Question suffixes

This routine is used to remove the suffixes such as ஆ, ஏ, ஓ.

(5.3.1.1) Example:

குழந்தையா ➔ குழந்தை

Is it child ➔ Child

5.3.2 Conjunction suffix

This routine is used to remove conjunction suffix உம்.

(5.3.2.1) Example:

யானையும் ➔ யானை

It and ➔ it

5.3.3 Common words

This part of the algorithm tries to remove some of the common words that are attached to verbs or nouns. These are not suffixes and are proper words.

(5.3.3.1) Example:

கடவுள்ளில்லாத ➔ கடவுள்

Whithout God ➔ God

5.3.4 Case suffixes

Tamil case suffixes are attached to the ends of nouns to express grammatical relations as well as meanings. This routine is to remove such kind of Case suffixes to find out stem words.

(5.3.4.1) Example:

கல்லூரியில் ➔ கல்லுரி

In college ➔ college

5.3.5 Plural suffix

This part of the algorithm to remove the plural suffix in Tamil is கள்.

(5.3.5.1) Example:

பறவைகள் ➔ பறவை

Birds ➔ Bird

5.3.6 Imperative suffixes

Imperative suffixes are used to command a person. Such kinds of suffixes are removed using this routine.

(5.3.6.1) Example:

காண்பி ➔ காண்

Show me ➔ see

5.3.7 Tense suffixes

This routine is used to remove tense indicating suffixes and Person suffixes.

(5.3.7.1) Example:

பாடுகின்றனர் ➔ பாடு

Singing ➔ sing
Apart from the standard suffixes the routine also removed கொண்டு and similar words.

5.4 Improved Light Stemmer Algorithm for Tamil Language

Input: Preprocessed Tamil words
Output: Stemmed (Root) words
Step 1: Eliminate the entire prefix based on prefix list and record the words. If it is found then there are four routines in the algorithm to handle the prefixes.
1.a) One is for handling the prefix in the questions.
1.b) Another one is for removing the pronoun prefixes.
1.c) Fixing the start of the word with substituting the vowel.
1.d) Eliminating the prefix 'Thiru'
Step 2: Eliminating the entire suffixes based on suffix list and record the words. If it is found, then suffix stripping routine is used to remove Question suffixes, Conjunction suffixes, common words, case suffixes, plural suffixes, complex plural suffixes, Imperative suffixes and Tense suffixes
Step 3: The word is also checked for adjective and tense. And they are replaced with its equivalent verb.
Step 4: if the given word is not match with prefix, suffix, adjective and tense list, then next possible suffix list is generated and add more rules
Step 5: Repeat the process from Step 1 until stem word is found

5.5 Minimum Length Criteria

Being a strong stemmer there is a tendency to over stem some words to single letters. Every suffix stripping routine checks for the length of the string before proceeding and after removing a suffix. The minimum length of the string is set as 4 characters. In Unicode a meaningful character can be represented by more than one code points, so the minimum length of the string is not 4 characters exactly. So the check made in the implementation actually only verifies the number of code points in the string than the actual meaningful characters. Also this routine which fixes the ending does not check for the length of the string. And this routine is still possible to get a stem with one character long.

5.6 K-Means Clustering Algorithm for Stemmed Tamil Documents

Clustering is a natural way to reduce the large dimensionality of the document vector space and helps in efficient relevant in Tamil document extraction. This research paper presents Novel approach for Stemming algorithm to extract Tamil text documents and apply K-Means algorithm to cluster the stemmed Tamil words. In statistics and machine learning, K-Means clustering is a method of grouping something, 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 (x1, x2… x*n*), where each stemmed Tamil word is a *d*-dimensional real vector, where ($k < n$) **S** = {*S*1, *S*2… *SK*} so as to minimize the within-cluster sum of squares

$$argmin\, s \sum_{i=1}^{k} \quad \sum_{x_j \in S_i} \quad \| X_j - \mu_i \|^2$$

Where, μ_i is the mean of points in Si.

5.6.1 K-Means Cluster Algorithm for Tamil Document

Input: Stemmed Tamil words from Improved Iterative Affix Stripping Stemmer Algorithm
Output: Group of Stemmed (Root) words
Step 1: Specify the number of cluster K.
Step 2: Determine the centriod from the given input.
Step 3: Find out 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 happened between the Clusters.
Step 5: Grouping takes place otherwise based on minimum distance between the stemmed Tamil words in cluster
Step 6: Repeat the execution from Step 2, until clusters are formed.

6. EXPERIMENTAL ANALYSIS

6.1 Test Data

The Tamil documents have been taken from the net and approximately 300 Tamil documents were taken and extracting more than 11,700 words. Text data has to go through several pre-processing stages in order to obtain clear and unambiguous data before stemming and cluster analysis. Given Tamil words are then converted into transliteration Tamil text. Then applied improved iterative affix stripping stemming algorithm to stem the article.

6.2 Experimental Result

The K-Means clustering algorithm is used to cluster the stemmed Tamil documents. Tamil stemmed words files are taken for clustering process. Initially the words in the item set table are taken and then searched the text files. The text file names, which contain the particular word, are stored in a vector.

Table 1. Performance Result Based on Stemmed Words

No of Documents	Words Stemmed after			
	Prefix Stripping	Suffix Stripping	Improved Affix Stripping	K-Mean Clustering
50	112	395	466	488
100	197	488	512	532
150	258	502	577	625
200	207	624	619	687
250	265	609	642	669
300	345	656	710	756

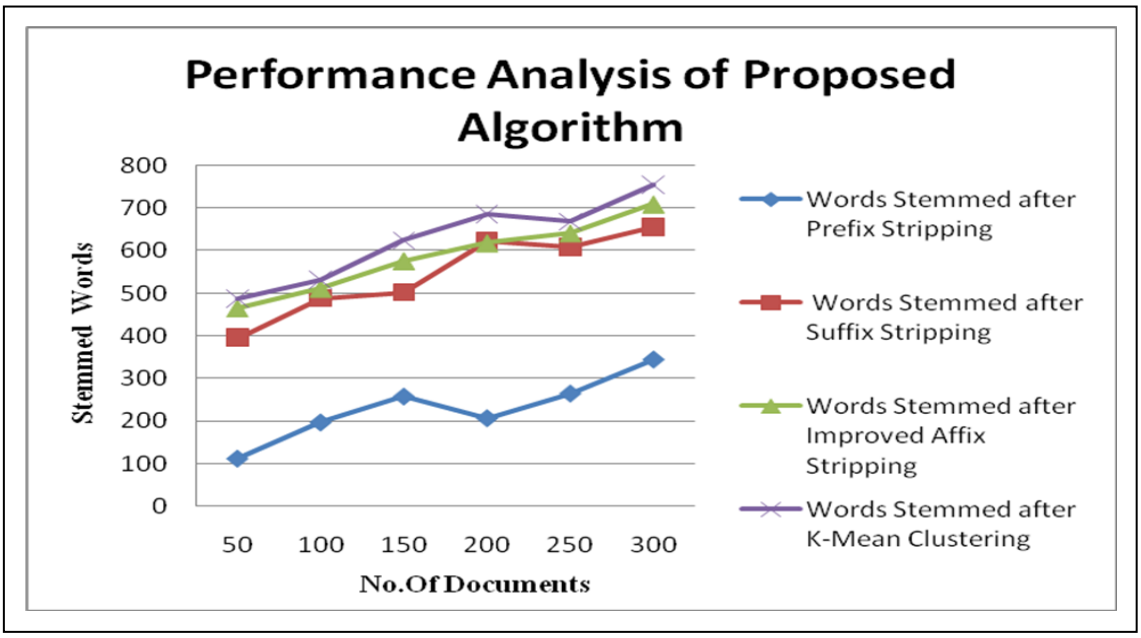

Fig 3: Performance Analysis for Proposed Tamil Stemmer Algorithm

Table 2. Performance Analysis Based on Time (Sec)

No of Documents	Time (Sec) Taken to find out Stemmed Words			
	Prefix Stripping	Suffix Stripping	Improved Affix Stripping	K-Mean Clustering
300	23.2	32.2	31.1	19.8
400	26.7	35.9	35.7	25.6
500	44.3	54.2	50.9	37.8
600	54.2	57.8	55.2	38.2
700	56.4	61.2	55.7	39.5
800	59.7	63.4	56.4	41.3

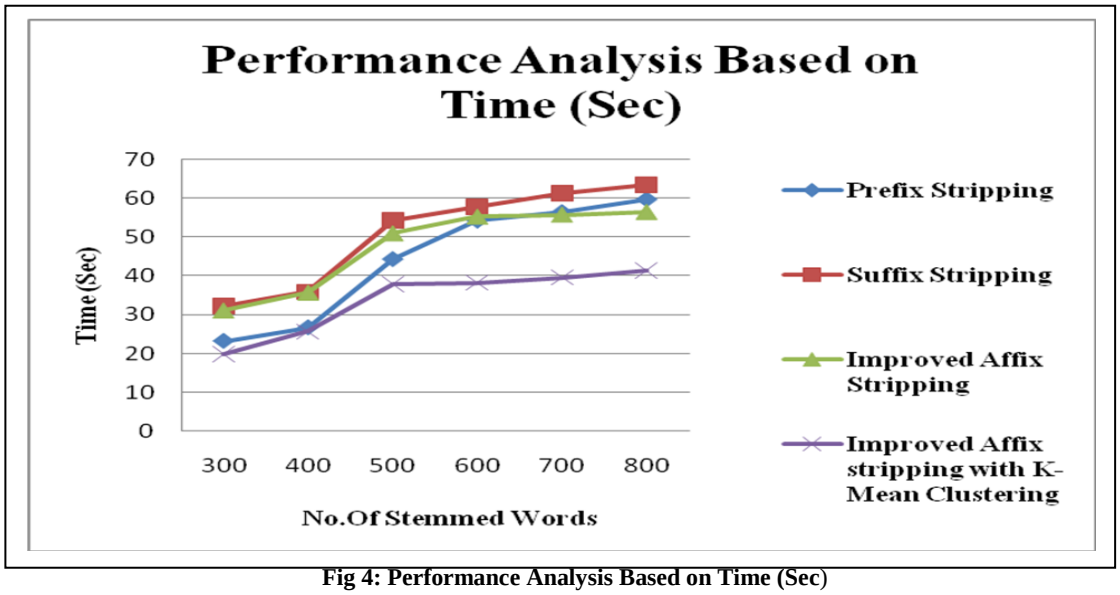

Fig 4: Performance Analysis Based on Time (Sec)

Likewise the entire text file's words are stored in separate vectors. Then the intersection is made and the text files contain two words. The iteration continues to find out 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, then words stemmed before applying proposed algorithm will reach 488 stemmed words, words stemmed after applying proposed algorithm up to 512 and words stemmed after clustering reaches up to 532. Similarly 300 documents are taken, then words stemmed before applying proposed algorithm will reach 656 stemmed words, words stemmed after applying proposed algorithm up to 710 and words stemmed after clustering reaches up to 756. The result shows that the number of words stemmed after clustering is better than number of words before clustering. Figure 3 shows the performance analysis of the proposed Tamil Stemmer Algorithm. The computational results from the Table 1 and Figure 3 clearly proved that improved affix stripping stemmer algorithm is achieving better performance in words stemmed after clustering compare than the words stemmed before clustering. The computational results from the Table 2 and Figure 4 clearly proved that improved iterative rule based affix stripping stemmer algorithm is achieving better performance based on Time (Sec) in words stemmed after clustering compare than the words stemmed before clustering.

7. CONCLUSION

This paper described about improved iterative rule based affix stripping algorithm. Stemming plays vital role in Tamil information retrieval, compared with all other languages. Improved Iterative Rule Based Affix Stripping Stemmer is best one since it removes stop words, definite documents, beginning of words and suffixes. Tamil text contains many definite articles that one could obtain to claim 99% tokenization accuracy simply by removing 'thiru' from the beginnings of words. Improved Iterative Affix Stripping Stemming is robust. The values obtained in the Table 1 and Table 2 indicates that the proposed algorithm is a strong stemmer and it provides good index compression ratio. An Improved Iterative Affix Stripping Stemmer algorithm is suitable for Information Retrieval of Tamil text documents.

8. FUTURE ENHANCEMENT

Sometimes unwanted clusters are formed. So there is a need for fine tuning the proposed algorithm once again. This is still inferior in terms of speed. Intend to test this feature as our future work to enhance the quality of the stemmer algorithm using clustering.

9. 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] 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, pages 35–39,Chiang Mai, Thailand, November 8, 2011.

[4] Khan, "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, 2007.

[5] Sajjad Ahmad Khan1, Waqas Anwar1, Usama Ijaz Bajwa1, Xuan Wang, "A Light Weight Stemmer for Urdu Language: A Scarce Resourced Language", 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", Second International Conference on Universal Digital Library Alexandria, Egypt, November 17-19, 2006.

[9] Frakes, William B. and Christopher J. Fox., "Strength and similarity of affix removal stemming algorithms", ACM, SIGIR Forum 37 (2003): 26-30.

[10] D. Freitag, "Morphology induction from term clusters", Ninth conference on computational natural language learning (CoNLL), pp. 128–135, 2005.

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

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

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

[14] Ramachandran, Vivek Anandan and Krishnamurthi, Ilango, "An Iterative Suffix Stripping Tamil Stemmer", Proceedings of the International Conference on Information Systems Design and Intelligent Applications: Volume 132, 583-590, 2012.

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

[16] M.Thangarasu et. al., and Dr.R.Manavalan, "Design and Development for Stemmer for Tamil Language: Cluster Analyses", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 7, ISSN: 2277 128X, July 2013.

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