STEMMING ALGORITHMS FOR TAMIL LANGUAGE: AN OVERVIEW

Dr. J. Indumathi[1], Amala S.P[2]

[1]indumathi@annauniv.edu, [2]amalasp21@gmail.com

Department of Information Science and Technology,
College of Engineering, Anna University, Guindy, Chennai -25, TamilNadu, India.

Abstract -- *Stemming is a process of linguistic normalization that attempts to extract a root word from the given inflection word. It maps all the derived forms of a word to a single root, the stem, a common form. Stemming algorithms are widely used in computational linguistics, information retrieval – query based systems such as the web search engine, and text mining. Stemming algorithms for Tamil language is still at its infancy. As a basis for evaluation this paper discusses the different types of stemming algorithms performed on Tamil language such as the rule based suffix stripping stemmer algorithm, rule based iterative affix stripping stemming algorithm, light stemmer and then points out the common linguistic problem in stemming algorithms with suitable solution. The performance of the stemming algorithms are evaluated based on the stemming errors whose metrics involve the under-stemming index (UI), and the over-stemming index (OI).*

Keywords – stemming algorithms; rule based suffix stripping, rule based iterative affix stripping, light stemmer algorithms, errors in stemmers.

1. INTRODUCTION

Stemming is an important feature for indexing and searching whereby it provides ways of finding morphological variants of search terms. For example, when a user sends a query stating "மரம் வளர்ப்பது எப்படி?", all the relevant results such as "மரங்களை வளர்ப்பது எப்படி?", மரத்தினை வளர்க்கும் வழிமுறைகள்" can be got only when stemming is used. Because in stemming, words like "மரங்களை, மரத்தை, மரங்கள், மரத்தினை, மரத்தின்" all are stemmed to a single root word or stem called "மர".

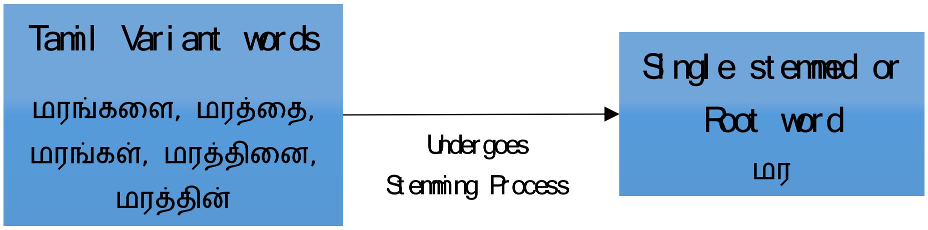

Figure 1. The Stemming Process

Stemming algorithms are not new, they have been studied [Lovins, J. B. (1968)] in Computer Science since 1968 till now to the Improved Rule based Iterative Affix Stripping Stemmer using K-Mean clustering [Kasthuri, M., et al., (2014)]. To increase the retrieval accuracy in the Information retrieval systems Stemming has been used.

Moreover, Stemming greatly aids in improving the performance of information retrieval by reducing the size of the index thus increasing the recall performance. As a single stem corresponds to a number of full terms, a single root word or stem is stored in place of a number of terms, thereby the index size is reduced. This holds good for a morphologically rich language like Tamil, where a single word in Tamil takes many forms.

This paper is organized as follows: Section 2 describes the related work done in development of stemmer for Tamil language. Section 3 gives a detailed description about the rule based suffix stripping stemmer algorithm, and the rule based iterative affix stripping stemming algorithm and discusses about the light stemmer. Section 4 presents the common linguistic problem in stemming algorithms with a suitable solution and Sections 5 gives the evaluation criteria for the stemming algorithms based on the stemming error metrics. The concluding remarks are given in Section 6.

2. RELATED WORK

In the earlier days, stemmers were developed for the English language. The growth of other languages has led to the development of the stemming algorithms for other languages too. Among Indian languages, stemming was first reported [Ramanathan. A., et al.,(2003)] followed by the development of a lightweight stemmer for Bengali [Islam, M. Z., et al.,(2007)] . Akram, Q. U. A., et al.,(2009) developed Assas-Band, an affix- exception-list based stemmer for the language Urdu. A morphological analyzer which can simultaneously serve as a stemmer too was first proposed [Vikram, T. N., et al., (2007)] for the language Kannada. Kumar, D., et al., (2010) designed and developed a stemmer for Punjabi. Ram, V. S., et al., (2010) introduced Malayalam stemmer for information retrieval. Patel, P., et al., (2010) proposed a Hybrid Stemmer for Gujarati. Saharia, N., et al., (2012) adopted a suffix stripping stemming approach along with a rule engine for Assamese. Dhabal Prasad Sethi (2013) developed a lightweight stemmer with derivational suffixes.

While there are many design and developments for stemming algorithms in other Indian and Foreign languages, stemming algorithms for Tamil is still at its infancy. It started late in 2013 where a Rule Based Iterative Affix Stripping Stemming Algorithm [Rajalingam, D. (2013)] was proposed, the algorithm here was implemented using a string processing language called Snowball and the correctness of the algorithm was also evaluated. A cluster analysis based stemmer [Thangarasu, M., Manavalan, R. (2013)] was also designed and developed in 2013 with a performance analysis [Thangarasu, M., & Manavalan, R. (2013)] of the stemmers in Tamil language. An Iterative Suffix Stripping Tamil Stemmer [Ramachandran, V. A., & Krishnamurthi, I. (2012)] and Iterative stemmer [Ramachandran, V. A., & Krishnamurthi, I. (2012)] was proposed by Ramachandran et al.,. Kasthuri, M., et al., (2014)] provided an Improved Rule based Iterative Affix Stripping Stemmer using K-Mean clustering.

3. RULE BASED SUFFIX STRIPPING AND ITERATIVE AFFIX STRIPPING STEMMER ALGORITHM AND THE LIGHT STEMMER

Rule based suffix stripping stemmer algorithm truncates the suffix of Tamil inflectional word. Based on the rules given, it converts the inflectional Tamil word into a single stemmed

Tamil word. Affix removal algorithms remove the suffixes and/ or prefixes from the inflectional word. The possible suffixes for a word in Tamil can be referred from the flowcharts. [Tamil Noun Flow Chart] and [Tamil Verb Flow Chart]

Light stemmer is proposed to overcome the issue of infinite verb generation by the rule based suffix stripping stemmer algorithm for certain Tamil words. Light stemmer is also a form of the rule based stemming algorithm. The representative indexing forms of a given word is found by the light stemming by truncating the suffixes.

In both the rule based and the light stemmer algorithms, the entire complex plural is eliminated in the first step. The main difference is that while the former deals with identifying the next possible affix according to the identified affix using the rules, the later uses a plural to singular conversion and adjective and tense word to the main word conversion.

4. COMMON LINGUISTIC PROBLEMS IN DEVELOPING TAMIL STEMMING ALGORITHMS

There are many difficulties while developing a stemming algorithm. Homographs should be taken into account for a morphologically rich language like Tamil. Homographs are those words that have a similar pronunciation but differ in the meaning with respect to the context. Moreover the irregular verb forms regarding the tenses also impact the design and development of an efficient stemmer.

For applying a stemming algorithm, one needs to have an extensive language expertise. There are mainly two errors that occur commonly in stemming – over stemming and under stemming. Over-stemming occurs when two words with different stems are stemmed to the same root word. This is a criteria of false positive. Under-stemming occurs when two words that should be stemmed to the same root word are not stemmed. This is a criteria of false negative.

5. EVALUATION OF THE STEMMING ALGORITHMS

The precision of the stemming algorithms can be understood by the type of errors, their conditions and the impact of the errors on the system. There are mainly two errors namely, under-stemming and over-stemming. These errors are aggressive as the mis-stemming leads to the loss of semantic information thus making the language unclear.

When a word is under-stemmed, there is a difficulty in identifying if two words are related based on their morphology, as their obtained stems are mostly equal, but not totally because of a suffix that has not been deleted. The opposite of this takes place in case of over-stemming. The main problem here is that it becomes more possible for two words to be wrongly detected as related but in fact are not related and share part of their morphological root as their stems are equal.

Paice has proved that light-stemming reduces the over-stemming errors but increases the under-stemming errors [Chris, D. P. (1990)] [Paice, C. D. (1994)]. On the other hand, heavy stemmers reduce the under-stemming errors while increasing the over-stemming errors. The under-stemming and over-stemming errors are calculated as follows.

The under-stemming index and over-stemming index [Ramachandran, V. A., et al.,(2012a.2012b))] is calculated by the formula:

Under-stemming = (Number of variants under-stemmed / Total variants)*100%
Over-stemming = (Number of variants over-stemmed / Total variants)*100%

Stemmer Effectiveness is calculated as

Stemmer Effectiveness = 100% - [Over-stemming % + Under-stemming %]

Another evaluation criteria is based on how much the stemming process compresses the input received and reduces the storage space needed. This factor is called the Index Compression factor (ICF) which is based on the conflation ratio.

6. CONCLUSION

The focal point of stemming is to reduce different grammatical forms / word forms of a word like its noun, adjective, verb, adverb etc. to its root form. In fact a perfect stemmer is very important in most of the Information Retrieval systems. A lot of research work to develop stemmers that improve recall as well as precision is the need of the hour. Research with respect to natural language processing for a morphologically rich language like Tamil is still at its infancy.

In this paper synopsis of different types of stemming algorithms (such as the rule based suffix stripping stemmer algorithm, rule based iterative affix stripping stemming algorithm, light stemmer etc.,) proposed for Indian languages ; with special emphasis for Tamil language. It also elaborates on the common linguistic problems arousing in the stemming algorithms along with suitable solutions. The performance of the stemming algorithms are evaluated based on the stemming errors whose metrics involve the under-stemming index (UI), and the over-stemming index (OI).

This paper serves as a catalyst to researchers emphasizing the need for development of a method and a system for efficient stemming that will reduce the heavy tradeoff between false positives and false negatives. It also gives an insight of development of a stemmer that uses the syntactical as well as the semantical knowledge to reduce stemming errors.

REFERENCES

- **[Akram, Q. U. A., et al., (2009)]** Akram, Qurat-ul-Ain, Asma Naseer, and Sarmad Hussain. "Assas-Band, an affix-exception-list based Urdu stemmer." In *Proceedings of the 7th Workshop on Asian Language Resources*, Association for Computational Linguistics, 2009. pp. 40-46.
- **[Chris, D. P. (1990)]** Chris, D. Paice. “Another Stemmer.” In *ACM SIGIR Forum*, 1990. vol. 24, no. 3, pp. 56-61.
- **[Dhabal Prasad Sethi (2013)]** Dhabal Prasad Sethi, **“**Design of Lightweight Stemmer for Odia Derivational Suffixes.” *International Journal of Advanced*

Research in Computer and Communication Engineering, vol. 2, Issue 12, December 2013.

- **[Harman, D. (1991)]** Harman, Donna. "How effective is suffixing?." *Journal of American Society for Information Sciences*, 42, no. 1, pp. 7-15, 1991.
- **[Islam, M. Z., et al., (2007)]** Islam, Md Zahurul, Md Nizam Uddin, and Mumit Khan. "A light weight stemmer for Bengali and its Use in spelling Checker.", 2007.
- **[Kasthuri, M., et al., (2014)]** Kasthuri, M., Ramesh Kumar, and S. Britto. "An Improved Rule based Iterative Affix Stripping Stemmer for Tamil Language using K-Mean Clustering." *International Journal of Computer Applications* 94, 2014.
- **[Kumar, D., et al., (2010)]** Kumar, Dinesh, and Prince Rana. "Design and Development of a Stemmer for Punjabi." *International Journal of Computer Applications* 11, no. 12, pp. 18-23, 2010.
- **[Lovins, J. B. (1968)]** Lovins, Julie B. *Development of a stemming algorithm*. MIT Information Processing Group, Electronic Systems Laboratory, 1968.
- **[Paice, C. D. (1994, August)]** Paice, Chris D. "An evaluation method for stemming algorithms." In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*. Springer-Verlag New York, Inc., 1994. pp. 42-50.
- **[Patel, P., et al., (2010)]** Popat, Pratikkumar Patel Kashyap, and Pushpak Bhattacharyya. "Hybrid Stemmer for Gujarati." In *23rd International Conference on Computational Linguistics*, 2010. pp. 51-55.
- **[Rajalingam, D. (2013)]** Rajalingam, Damodharan. "A Rule Based Iterative Affix Stripping Stemming Algorithm For Tamil." *12th International Tamil Internet Conference.* 2013. pp. 28-34.
- **[Ram, V. S., et al., (2010)]** Ram, V. S., and S. L. Devi. "Malayalam stemmer." *Morphological Analysers and Generators*, 2010. pp. 105-113.
- **[Ramachandran, V. A., et al., (2012, January)]** Ramachandran, Vivek Anandan, and Ilango Krishnamurthi. "An Iterative Suffix Stripping Tamil Stemmer." In *Proceedings of the International Conference on Information Systems Design and Intelligent Applications 2012 (INDIA 2012) held in Visakhapatnam, India, January 2012,* Springer Berlin Heidelberg, 2012. pp. 583-590.
- **[Ramachandran, V. A., et al., (2012, January)]** Ramachandran, Vivek Anandan, and Ilango Krishnamurthi. "An iterative stemmer for tamil language." In *Intelligent Information and Database Systems*, Springer Berlin Heidelberg, 2012. pp. 197-205.
- **[Ramanathan, A., et al., (2003, April)]** Ramanathan, Ananthakrishnan, and Durgesh D. Rao. "A lightweight stemmer for Hindi." In *the Proceedings of EACL*, 2003. pp. 43-48.
- **[Saharia, N., et al., (2012, August)]** Saharia, Navanath, Utpal Sharma, and Jugal Kalita. "Analysis and evaluation of stemming algorithms: a case study with Assamese." In *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*. ACM, 2012. pp. 842-846.
- **[Thangarasu, M., Manavalan, R. (2013, July)]** Thangarasu, M., and R. Manavalan. **"**Design and Development of Stemmer for Tamil Language: Cluster Analysis." *International Journal of Advanced Research in Computer Science and Software Engineering,* Volume 3, Issue 7, ISSN: 2277 128X, July 2013.
- **[Thangarasu, M., et al., (2013)]** Thangarasu, M., and R. Manavalan. "Stemmers for Tamil Language: Performance Analysis." In *International Journal of Computer Science & Engineering Technology (IJCSET),* 2013, Vol. 4 No. 7, pp. 902-908.

- **[Vikram, T. N., et al., (2007)]** Vikram, T. N., and Shalini R. Urs. "Development of Prototype Morphological Analyzer for the South Indian Language of Kannada." In *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*. Springer Berlin Heidelberg, 2007. pp. 109-116.
- Tamil Noun Flow Chart, http://www.aukbc.org/research_areas/nlp/projects/morph/NounFlowChart.pdf
- Tamil Verb Flow Chart, http://www.au-kbc.org/research_areas/nlp/projects/morph/VerbFlowChart.pdf