A COMPREHENSIVE ANALYSIS OF STEMMERS AVAILABLE FOR INDIC LANGUAGES

Harshali B. Patil, B. V. Pawar, and Ajay S. Patil

School of Computer Sciences, North Maharashtra University, Jalgaon, India

ABSTRACT

Stemming is the process of term conflation. It conflates all the word variants to a common form called as stem. It plays significant role in numerous Natural Language Processing (NLP) applications like morphological analysis, parsing, document summarization, text classification, part-of-speech tagging, question-answering system, machine translation, word sense disambiguation, information retrieval (IR), etc. Each of these tasks requires some pre-processing to be done. Stemming is one of the important building blocks for all these applications. This paper, presents an overview of various stemming techniques, evaluation criteria for stemmers and various existing stemmers for Indic languages.

KEYWORDS

Stemming, light weight, affix removal, HMM, n-gram, Indian languages

1. INTRODUCTION

In 21st century the e-data in regional languages appearing on internet increased drastically. But due to the unavailability of NLP tools for these languages, access to this data is limited. The scarcity of tools has attracted the attention of researchers and industry persons towards the development of efficient systems for IR, text summarization, opinion mining, clustering, classifications etc., for regional languages. The inflected nature of a language poses several challenges for the automated processing of natural language data. In an information retrieval system that does not use any word normalization, search results are directly affected as many relevant documents are missed during the retrieval process. For instance, in case of the search query “stemmer”, the system will not return documents containing “stemming” or “stem” if the documents do not contain the term “stemmer”. Term conflation is the solution to these types of problems. In many natural languages several words sharing the same morphological variant (root) can be related to the same topic. Stemming is one of the popular term conflation techniques used as a preliminary step in many natural language processing tasks. In linguistics stem is the form that unifies the elements in a set of morphologically similar words. Stemming is the process which determines the stem of the given word. The goal of a stemming algorithm is to reduce variant word forms to a common morphological root, called stem [1]. India is a multilingual country where there are 22 official languages which belongs to 4 different families of languages. Among these 22 languages, 15 languages belong to Indo-Aryan, 4 are Dravidian, 2 are Tibeto-Burman, and 1 belongs to Munda family.

This paper presents a comprehensive analysis of stemmers available for Indic languages. The paper is organized as follows: related work is provided in section 2. Section 3 describes stemming process in detail as well as existing stemmers for Indic languages and Section 4 concludes the paper.

DOI: 10.5121/ijnlc.2016.5104

2. RELATED WORK

Stemming is a well known research problem; but in early days of stemming it was studied only for English language. Lovin's stemmer is one of the oldest stemmer developed for English using context sensitive longest match technique. Other most notable stemmers for English includes: Porter's stemmer, Dawson stemmer, Paice and Husk stemmer. Poter's stemmer has became the de-facto standard for English language. Some non-English stemmers were presented during 1990-2000, but study related to stemming for Indic languages has been started after 2000. The subsequent section discusses the work done related to stemmer development for Indic languages. Most of the work related to stemmer development has most recently initiated. Sarkar et.al (2012) surveyed stemmers for Bengali and has inferred that rule-based stemmers would be more suitable for Bengali, whereas suffix strippers may not be sufficient for Bengali. The author also concludes that POS tagged data and lexicon may improve performance in Bengali stemmer and accuracy based evaluation techniques should be appropriate for measuring stemmer performance [21]. Lakshmi et. al. (2014) reviewed literature related to stemming algorithms for Indian and Non-Indian languages and found that there is a need to develop a language independent stemmer for all languages [27]. Madhurima et.al (2013) analyzed popular stemming algorithms supporting information retrieval system and has shown that no perfect stemmer has been designed so far to match all the requirements [22]. Kasthuri et. al (2014) comprehensively analyzed stemming algorithms for Indian and non-Indian languages, but the analysis focuses on only recent stemmers developed during 2010 to 2014 [8]. Bijal et. al overviewed stemming algorithms for Indian and non-Indian languages and discussed 9 stemmers related to Indian languages [24]. Rakholia et. al. (2014) presented comparative analysis of different stemmers and character recognition algorithms for Indian Gujarati script where the author discussed the stemmers available for Gujarati language only [25]. Sethi el.al (2014) presented a literature survey related to stemming algorithms for Odia language [26]. Patil et.al (2014) presented a part-of-speech tagger for Marathi language using limited training corpora and obtained 78.82% accuracy with the rule-based technique [41]. The study related to developing links of compound sentences for parsing through Marathi link grammar parser was carried out by Vaishali Patil et. al (2014) [42,43]. Nita Patil et.al (2016) surveyed the name entity recognition (NER) systems with respect to Indian and foreign languages and concludes that very less work on NER is reported for Indian languages like Marathi and Guajarati [44]. Juhi et.al (2013) improved the quality of Gujarati-Hindi machine translation through part-of-speech tagging and stemmer assisted transliteration and achieved 93.09% overall efficiency of the transliteration scheme [45]. Kridanta analysis for Sanskrit has been done by Murali et.al (2014) and achieved 92.75% and 95.37% precision and recall for Kridanta analyzer [46].

3. STEMMING

The following section presents the detail information about stemming in terms of techniques used to develop stemmers, various types of errors generated while stemming, evaluation criteria for stemmers, and the existing stemmers for Indic languages. The major advantages and limitations provided by stemming for information retrieval are as given below:

Advantages of stemming

- Use of stemmers in IR decreases index size because for all the terms that belongs to single conflation class are reduced to single term. [16, 20].
- Reduction in index ultimately reduces the storage space required to store the inverted index file. file [16, 47, 48].

- Instead of using term variation if the stem is used for IR it increases the recall of retrieval systems. [2, 20].
- It is used as an important component for pre-processing in many applications.

Limitations of stemming

- Exceptional cases are also grouped together e.g – university and universal – univers
- Sometimes decreases the retrieval performance

3.1. Classification of stemming techniques

The stemmers are broadly classified into two types: manual and automatic[48]. Manual stemmers stems the terms manually while automated stemmers can be developed by using various techniques. Simplest affix removal to a complicated technique like n-gram or Hidden Markov Model (HMM) has been used for development of stemmers for various languages. Fig. 1 classifies the major stemming techniques used for stemmer development.

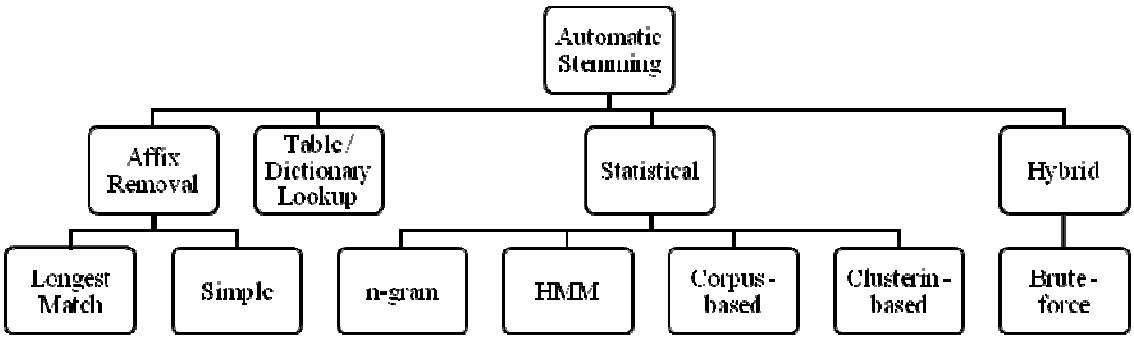

Figure 1. Some automatic stemming techniques

3.1.1. Affix removal Technique

These types of stemmers are also called as rule-based stemmers or suffix stripper. This is one of the oldest and simplest techniques used for stemmer development. These types of algorithms uses list of suffixes and with each suffix the criteria under which it may be removed from a word to leave a valid stem. The affix removals based on rules are either done based on longest match basis or in iterative manner. Fig 2 shows the procedure for affix removal stemmers. Some Indic stemmers developed using this technique includes [3, 5, 9, 12, 38, 39].

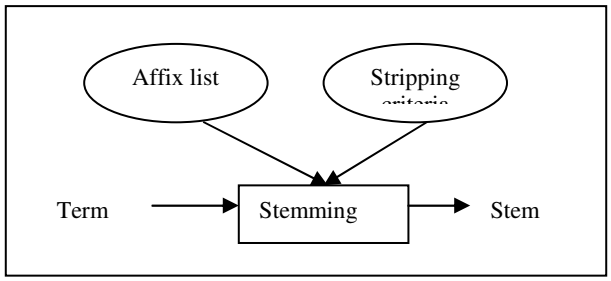

Figure 2. Affix removal stemming procedure

The sample rule related to English affix removal / rule-based stemmer is as given below:

Rule :- IES -> I
E.g. :- Ponies -> Poni

3.1.2. Table / Dictionary lookup

In this technique a table of corresponding terms along with their stem is used. The stemming is done by searching the corresponding term in the table and retrieving the stem related to that term [48]. This technique is not popular for stemmer development due to limitations related to it: like dictionary of term-stem are not available for many languages, the accuracy totally depends on the size of dictionary, and dictionary are domain dependent; but this technique can be combined with rule-based technique to develop hybrid stemmers and increase the stemming accuracy. Fig. 3 shows the process of stemming with help of dictionary / table lookup approach.

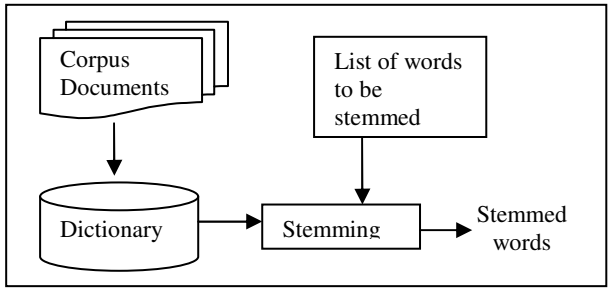

Figure 3. Stemming procedure for dictionary / table lookup

3.1.3. Statistical Techniques

These types of stemmers uses some statistical measure for stemmer development like n-gram, HMM, clustering-based method, corpus-based method. Some of them are briefly explained in the following section. In this approach first the system is trained with the large corpus and then inflected forms are submitted to the trained module for stemming.

N-gram:

N-gram is a set of n consecutive characters extracted from a word. In this technique the similar words will have a high proportion of n-grams in common. The n-gram based stemmer calculates the association measure between the pair of terms based on shared unique diagrams. Then similarity measures are determined for all pairs of terms in the database, forming a similarity matrix. After that terms are clustered using single link clustering. [40]

HMM:

The HMM is a finite sequence of states, and a set of transitions between states that are taken based on the input observations. Each character comprising a word is considered as a state. All possible states are divided into two groups (roots and suffixes) Word building process are defined by the transitions between states. [29]

Clustering-based:

In these types of stemmer first the Equivalence classes are discovered. Then a set of string distance measures are defined. The lexicon for a given text collection is clustered using complete linkage clustering technique to identify these equivalence classes. For some infrequent cases the author uses some post-processing for each cluster. [2]

Corpus-based:

First the set of potential suffixes are formed. Then the equivalence classes are generated by combining the common prefix and potential suffix information. Then the common prefix and potential suffixes help to recognize the better equivalence classes through mutual agreement. [14]

3.1.4. Hybrid

When any two or more approaches are combined for stemmer development then the stemmer becomes hybrid. Most hybrid stemmers were developed by combining lookup table with affix removal. They overcome the disadvantages of one technique by combining it with other; for example the lookup table based stemmer cannot able to stem if term is not present in the table so when this type of stemmer is combine with affix removal then it can be able to stem the term which are not present in dictionary. . The work related to hybrid stemmers for Indic languages has been reported by [4, 6, 15, 29, 30, 37].

3.2. Comparison of stemming techniques

Table 1 compares the stemming techniques (rule based, dictionary lookup, statistical and hybrid) in terms of advantages and limitations.

Table 1. Comparison of stemming techniques.

Technique	Advantages	Limitations
Rule-based	Easy to implement, requires less memory, dictionary not required and fast	Rule preparation needs to be done manually, Over-stemming and under-stemming error rate is high.
Dictionary lookup	has capability to work with exceptional cases, under and over-stemming error rate is reduced and is fast	Dictionary / table is not available, storage and retrieval overhead, accuracy depends on dictionary size.
Statistical Technique	Applicable for wide variety of languages	Requires large corpus to train the system, takes more processing time, requires significant amount of memory.
Hybrid	Combination of more than one technique, overcomes the drawbacks of each other.	If dictionary lookup is combined then extra overhead for storage.

3.3. Evaluation criteria for stemmers

Stemmer is used as a pre-processing component for various NLP applications. There are several criteria that are used for evaluation of stemmers. Some of them include: correctness, retrieval effectiveness, and compression performance [48]. Generally stemmer is evaluated based on accuracy provided by it for stemming. Paice has given a method to evaluate stemmers based on two types of error counting [33]. Under-stemming errors and over-stemming errors are used to

evaluate the stemmer. Section 3.4 discusses these two types of errors. In IR few studies has evaluated stemmers as a method for index compression while some discuss the improvement in precision and recall for IR.

Frakes and Fox has given the stemmer strength metrics. [31,32] The degree to which a stemmer changes words that it stems is called stemmer strength. Some ways to measure stemmer strength are:

- The number of words per conflation class.
- Index compression factor.
- The number of words and stems that differ.
- The mean number of characters removed in forming stems.
- The median and mean modified Hamming distance between words ad their stems.

3.4. Types of errors in stemming

Generally two types of errors i.e over-stemming and under-stemming errors are generated while stemming. [33]

3.4.1. Over-stemming

Over-stemming errors refers to the words that should not be grouped together by stemming but they are grouped together. These types of errors will affect on precision of IR.

3.4.2. Under-stemming

Under-stemming refers to words that should be grouped together by stemming, but that are not grouped together. These type of errors tend to decrease the recall in an IR search.

3.5. Existing stemmers for Indic languages

Significant work related to stemmer development has been done for non-Indic languages like English, Arabic, etc. whereas stemming work is in progress for Indic languages. The work related to stemmer development for various Indic languages has started after 2000. Till 2010, not much work related to stemmer development has been reported for Indic languages. India is a multilingual country where there are 22 official languages. Among these languages stemmers are available for 13 Indic languages. Some language independent stemmers are also available. Table 2 presents the existing stemmers for Indic languages.

Table 2. Existing stemmers for Indic languages.

Sr. No.	Language	Author & Reference	Year	Approach	Accuracy / Results
1	Hindi	Ramanathan el al [9]	2003	Suffix removal	Results are favourable and can be used effectively in information retrieval
2		Pande et. al [36]	2008	Unsupervised	89.90%
3		Dolamic et.al [11]	2010	Light and aggressive	Improvements over no stemming: 19.3% with light stemmer, 27.6% with aggressive stemmer

4		Mishra et. al [37]	2012	Hybrid	91.59%
5		Gupta [5]	2014	Rule –based	83.65%
6	Bengali	Sarkar et. al [39]	2008	Rule-based	89% and above
7		Zahurul et. al [10]	2008	Lightweight	90.80%
8		Dolamic et.al [11]	2010	Light and aggressive	Improvements over no stemming: 13.7% with light stemmer , 17.7% aggressive stemmer
9		Das et. Al [12]	2011	Rule-based	0.4748 (Mean Average Precision)
10	Punjabi	Kumar et. al [13]	2010	Brute force algorithm	80.73%
11		Gupta et.al [38]	2011	Rule-based	87.37%
12		Joshi et.al [6]	2014	Hybrid	95.6%
13	Marathi	Dolamic et. al [11]	2010	Light and aggressive	Improvements over no stemming: 13.9% with light stemmer , 41.6% with aggressive stemmer
14		Majgaonkar et. al [3]	2010	Rule -based and unsupervised	80.7% for rule-based 82.5% for unsupervised
15	Tamil	Ramchanrdan et. al [19]	2012	Suffix stripping	84.79%
16		Thangarasu et. al [20]	2013	Light stemming	83.28%
17	Odia	Chaupattnaik et. al [17]	2012	Suffix stripping	88%
18		Sethi [18]	2013	Light weight	85%
19	Assamese	Saharia et. al. [29]	2013	Hybrid (rule-based and HMM)	92%
20	Gujarati	Patel el.al [4]	2010	Hybrid	67.86%
21		Suba et.al [30]	2011	Hybrid and rule-based	90.70% for hybrid 70.70% for rule-based
22	Kannada	Bhat [16]	2013	Statistical technique	88.82%
23	Kokborok	Patra et. al [23]	2012	Rule –based	80.02% for minimum suffix stripping, 85.13% for maximum suffix stripping
24	Malayalam	Prajitha et. al [35]	2013	Suffix stripping	Computationally inexpensive and domain independent.
25	Manipuri	Meitei et. al.[15]	2015	Hybrid	86.29%
26	Telugu	Kumar[34]	2013	Unsupervised	85.40%

27	Urdu	Gupta et.al[28]	2013	Rule-based	86.50%
28	Language independent	Majumder et. al [2]	2007	Clustering - based	Improvement in the recall for information retrieval
29		Paik et. al [14]	2011	Statistical technique	Outperformance observed over the well known rule-based stemmers for all languages under study
30		Husain [7]	2012	Length-based and frequency-based unsupervised approach	Length based 79.63% (Urdu), 82.6% (Marathi), frequency based 84.27% (Urdu), 63.5% (Marathi)

Various techniques were used for the evaluation of stemmers as mentioned in section 3.3. It is observed that the percentage accuracy provided by the stemmer was generally used to evaluate the stemmers. Form the literature review related to Indic languages stemmer development it is found that there is a difference in the accuracy levels provided by various types of stemmers. Table 3 presents the percentage accuracy obtained by using various approaches for Indic languages stemmer development.

Table 3. Accuracy ranges for stemming techniques

Sr. No.	**Technique**	**Accuracy range (%)**
1	Rule –based	80.02 – 89
2	Statistical	63.5 - 89.9
3	Hybrid	67.86 - 95.6

From table 3 it is observed that the maximum accuracy level provided by rule-based and statistical techniques are nearly same but there is lots of difference between the minimum accuracy levels achieved by both of these techniques. The maximum accuracy reported for Indic languages are obtained by the use of hybrid technique.

4. CONCLUSION

Stemmer is an important and basic component in many natural language processing applications. The accuracy of the stemmer strongly affects the results of the system in which it is used. Various stemmer development techniques are being explored and studied for different languages across the world. This paper surveyed the stemmers available for Indic languages. From literature survey it is observed that the work related to stemmer development for some of the Indic languages like Mizo, Santhali, etc has not been reported. The rule-based and light stemming techniques are widely used for stemmer development for languages like Hindi, Marathi, Tamil, etc. Hybrid approaches are also used by some researchers to avoid limitations of using single technique. Hybrid technique has reported more accuracy than the other techniques for some of the Indic languages. Though some work related to Indic stemmer development has been reported still much work needs to be done. The techniques like dictionary lookup or hybrid approach needs to be evaluated for the languages like Marathi, Hindi, etc. We intend to develop a stemmer for Marathi language that will be used for efficient information retrieval. The survey of the existing

techniques will be helpful in deciding the applicability of these techniques for Marathi stemmer development.

ACKNOWLEDGEMENTS

The authors are thankful to University Grants Commission (UGC), New Delhi, India, for financial assistance for the research study under the scheme of Special Assistance Programme (SAP) of Departmental Research Support (DRS) Phase I.

REFERENCES

[1] M. Bacchin, N. Ferro, & M. Melucci (2002) "The Effectiveness of a Graph-based Algorithm for Stemming", E.-P. Lim et. al (Eds.): ICADL 2002, LNCS, 2555, pp 117-128.
[2] P. Majumder, M. Mitra, S. K. Parui, and G. Kole , P. Mitra and K. Datta, (2007) "YASS: Yet Another Suffix Stripper", ACM Transactions on Information Systems, Vol. 25, No. 4, Article 18.
[3] M. M. Majgaonkar & T. J. Siddiqui, (2010) "Discovering Suffixes: A Case Study for Marathi Language", Int. Journal on Computer Science and Engineering Vol. 02, No.08, pp 2716-2720.
[4] P. Patel, K. Popat & P. Bhattacharyya, (2010) "Hybrid Stemmer for Gujarati", Proc. of the 1st workshop on South & Southeast Asian Natural Language Processing, pp 51-55.
[5] V. Gupta, (2014) "Hindi Rule-based Stemmer for Nouns", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 4, No.1, pp 62-65.
[6] G. Joshi & K. D. Garg, (2014) " Enhanced Version of Punjabi Stemmer using Synset", Int. Journal of Advanced Research in Computer Science and Software Engineering, Vol. 4, No.5, pp 1060-1065.
[7] M. S. Husain, (2012) "An Unsupervised Approach to Develop Stemmer", Int. Journal on Natural Language Computing Vol. 1, No.2, pp 15-23.
[8] M. Kasthuri & S. B. R. Kumar, (2014) "A Comprehensive Analysis of Stemming Algorithms for Indian and non-Indian languages", Int. Journal of Computer Engineering and Applications, Vol. 7, No. 3, pp 1-8.
[9] A. Ramanathan, D. D Rao, (2003) "A Lightweight Stemmer for Hindi", Proc. of the EACL.
[10] M. D. Zahurul Islam, M.D.N. Uddin & M. Khan, (2007) "A Light Weight Stemmer for Bengali & Its Use in Spelling Checker", Proc. of Ist Int. Conference on Digital Communications & Computer Applications.
[11] L. Dolamic, J. Savvo, (2010) "Comparative Study of Indexing and Search Strategies for the Hindi, Marathi and Bengali Language", Special Issue of ACM Transaction on Asian Language Information Processing on IR for Indian Languages Vol. 9 No. 3.
[12] S. Das, P. Mitra, (2011) "A Rule Based Approach of Stemming for Inflectional and Derivational Words in Bengali", Students' Technology Symposium IEEE No. Date: 14-16 Jan. 2011 pp 134 - 136 Kharagpur.
[13] D. Kumar, P. Rana, (2011) "Stemming of Punjabi Words by Using Brute Force Technique", Int. Journal of Engineering Science and Technology Vol. 3 No. 2 pp. 1351-1358.
[14] J.H. Paik, & S. K. Parui, (2011) "A Fast Corpus Based Stemmer", ACM Transactions on Asian Languages Information Processing Vol. 10, No. 2, Article No. 2.
[15] S. P. Meitei, B. S. Purkayastha & H. M. Devi, (2015) "Development of a Manipuri Stemmer: A Hybrid Approach", Int. Symposium on Advanced Computing & Communication 2015.
[16] S. Bhat, (2013) "Statistical Stemming for Kannada", The 4th Workshop on South and Southeast Asian NLP, Int. Joint Conference on Natural Language Processing, pp 25-33.
[17] S. Chaupattnaik, S. S. Nanda, S. Mohanty, (2012) "A Suffix Stripping Algorithm for Odia Stemmer", Int. Journal of Computational Linguistics and Natural Language Processing.
[18] D. P. Sethi, (2013) "Design of Lightweight stemmer for Odia Derivational Suffixes", Int. Journal of Advanced Research in Computer and Communication Engineering, Vol. 2,No. 12.
[19] V. A. Ramchandran and I Krishnamurthi, (2012) " An Iterative Suffix Stripping Tamil Stemmer", Proceedings of the International Conference on Information Systems Design and Intelligent Applications, Advances in intelligent and Soft Computing, Vol 132. pp 583-590.
[20] M. Thangarasu & R. Manavalan, (2013) "Stemmers for Tamil language: Performance Analysis", M. Thangarasu et. al./ Int. Journal of Computer Science & Engineering Technology, Vol. 4, No. 7.

[21] S. Sarkar, & S. Bandyopadhyay, (2012) "On the Evolution of Stemmers: A Study in the Context of Bengali Language", Int. Journal of Computational Linguistics and Natural Language Processing, Vol. 1, No. 2, pp 51-59.
[22] Madhurima V., T. V. N. Rao, & L. S. Bhargavi, (2013) " Analysis of Popular Stemming Algorithms Supporting Information Retrieval system", Int. Journal of Computer Organization Trends, Vol. 3, No. 9, pp 377-385.
[23] B. G. Patra, K. Debbarma, S. Debbarma, D. Das, A. Das, S. Bandyopadhyay (2012) "A Light Weight Stemmer in Kokborok", Proc. of the 24th Conference on Computational Linguistics and Speech Processing, pp -318-325.
[24] D. Bijal, & S. Sanket, (2014) "Overview of Stemming Algorithms for Indian and Non-Indian Languages", Int. Journal of Computer Science and Information Technologies, Vol. 5, No. 2, pp 1114-1146.
[25] R. M. Rakholia & J. R. Saini, (2014) "A Study and Comparative analysis of Different Stemmers and Character Recognition Algorithms for Indian Gujarati Script", Int. Journal of Computer Applications, Vol. 106, No. 2, pp 45- 50.
[26] D. P. Sethi, & S. K. Barik, (2014) "A Literature Survey: Stemming Algorithm for Odia Language", Int. Journal of Advanced Research in Computer Engineering & Technology, Vol. 3. No. 1 pp 9- 11.
[27] R. V. Lakshmi, & S. B.R. Kumar, (2014) " Literature Review: Stemming Algorithms for Indian and Non-Indian Languages", Int. Journal of Advanced Research in Computer Science & Technology, Vol. 2, No. 3, pp. 349-352.
[28] Gupta V., Joshi N., Mathur I. (2013), "Rule Based Stemmer in Urdu", Proc. of 4th Int. Conference on Computer and communication Technology, pp 129-132.
[29] N. Saharia, K. M. Konwar, U. Sharma, J. K. Kalita, (2013) "An Improved Stemming Approach using HMM for a Highly Inflectional Language", A. Gelbukh (Ed). CICLing 2013, part I, LNCS 7816, pp 164-173.
[30] K. Suba, D. Jiandani, & P. Bhattacharyya (2011) "Hybrid Inflectional Stemmer and Rule-based Derivational Stemmer for Gujarati", Proc. of 2nd Workshop on south & Southeast Asian Natural Language Processing, pp 1-8.
[31] S. R. Sirsat, V. Chavan, & H. S. Mahalle (2013), "Strength and Accuracy Analysis of Affix Removal Stemming Algorithms", Int. Journal of Computer Science and Information Technologies, Vol. 4, No.2, pp 265-269.
[32] Frakes W. B., & Fox C. J. (2003) "Strength and Similarity of Affix Removal Stemming Algorithms", ACM SIGIR Forum, Vol. 37, No.1, pp 26-30.
[33] Paice C. D. (1996) "Method for Evaluation of Stemming Algorithms Based on Error Counting", JASIS, Vol. 47, No.8, pp 632-649.
[34] A. P. Siva Kumar, Dr. P. Premchand, & Dr. A. Govardhanv (2011) "TelStem:An Unsupervised Telugu Stemmer with Heuristic Improvements and Normalized Signatures". Int. Journal of Computational Linguistics and Applications ,Vol. 2, No. 1 , pp 13-23.
[35] Prajitha U., Sreejith C, P. C. R. Raj (2013) "LALITHA: A Light Weight Malayalam Stemmer using Suffix Stripping Method", Int. Conference on Control Communication and Computing, pp 244-248.
[36] A. K. Pandey ,T. J Siddiqui, (2008) "An Unsupervised Hindi Stemmer with Heuristic Improvements", Proc. of the Second Workshop on Analytics for Noisy Unstructured Text Data ACM, pp. 99-105.
[37] U. Mishra, C. Prakash (2012) "MAULIK: An Effective Stemmer for Hindi Language" Int. Journal on Computer Science and Engineering Vol. 4. No. 5 pp. 711 – 717
[38] V. Gupta, G. S. Lehal (2011) "Punjabi Language Stemmer for Nouns and Proper Names", Proc. of the 2nd Workshop on South and Southeast Asian Natural Language Processing, pp. 35-39.
[39] S. Sarkar, S. Bandyopadhyay (2008) "Design of a Rule-based Stemmer for Natural Language Text in Bengali", Proc. of the IJCNLP – 08 Workshop on NLP for Less Privileged Languages, pp 65 -72.
[40] J. Mayfield ,P. Mcnamee, (2003) "Single N-Gram Stemming", Proc. of the 26th Annual Int. ACM SIGIR Conference on Research and Development in Information Retrieval.
[41] H. B. Patil, A. S. Patil and B. V. Pawar (2014) " Part-of-Speech Tagger for Marathi Language using Limited Training Corpora", IJCA Proc. on National Conference on Recent Advances in Information Technology NCRAIT(4) , pp 33-37.
[42] Vaishali. B. Patil & B. V. Pawar (2015) "Modeling Complex Sentences for parsing through Marathi Link Grammar Parser", Int. Journal of Computer Science Issues, Vol. 12, Iss. 1, No 2, pp 108-113.

[43] Vaishali. B. Patil & B. V. Pawar (2014) "Developing Links of Compound Sentences for Parsing Through Marathi Link Grammar", Int. Journal on Natural Language Computing, Vol. 3, No 5/6, pp 108-113.

[44] Nita Patil, Ajay S. Patil & B. V. Pawar (2016) "Survey of Named Entity Recognition Systems with respect to Indian and Foreign Languages", Int. Journal of Computer Applications, Vol. 134, No. 16, pp 21-26.

[45] Juhi Ameta, Nisheeth Joshi & Iti Mathur (2013) "Improving the Quality of Gujarati-Hindi Machine Translation through Part-of-Speech Tagging and Stemmer Assisted Transliteration", Int. Journal on Natural Language Computing, Vol. 2, No.3, pp 49-54.

[46] N. Murali, Dr. R.J. Ramasreee & Dr. K.V.R.K. Acharyulu (2014) "Kridanta Analysis for Sanskrit", Int. Journal on Natural Language Computing, Vol. 3, No.3, pp 33-49.

[47] C. Moral, A. Antonio, R. Imbert & J. Ramírez (2014) "A Survey of Stemming Algorithms in Information Retrieval", Information Research, Vol. 19, No. 1.

[48] William B. Frakes and Ricardo Baeza-Yates (1992) Information Retrieval: Data Structures & Algorithms, Prentice Hall