MTStemmer: A multilevel stemmer for effective word pre-processing in Marathi

Virat Giri[1*], M M Math[2], U P Kulkarni[3]

[1]Department of Computer Science and Engineering, Sanjay Ghodawat Polytechnic, Kolhapur, India
[2]Department of Computer Science and Engineering, KLS, Gogte Institute of Technology, Belgaum
[3]Department of Computer Science and Engineering, SDM College of Engineering and Technology, Dharwad, India
virat.giri@gmail.com[*1]

Article History: Received: 10 November 2020; Revised: 12 January 2021; Accepted: 27 January 2021; Published online: 05 April 2021

Abstract: In natural language processing, it is important that the context and the meaning of words are retained while also ensuring the efficacy of the data modelling process. During human-to-human interactions, special care is taken regarding the tense and phrasing of the words by taking into consideration the rules of grammar of the specific language. While this modification of words is necessary for framing consistent sentences, these appendages do not add significant value to the original meaning of the word. Stemming is the process of converting words back to their root form for efficient and accurate modelling of the data. In this paper, MTStemmer, a new stemmer for the Marathi language is proposed. It focuses on the stripping of suffixes for obtaining the root word form. The proposed stemmer applies a multilevel approach by taking into consideration both auxiliary verb-based suffixes and gender-based suffixes. The presented approach intends to improve upon the limitations of the previously proposed stemmers for this language. The stemming performed by the stemmer is found to be more accurate in terms of mapping to the root words. Stemming is often an important pre-processing step before processing the data further for the main task. The benefit of the proposed stemmer is demonstrated by using it for an extractive Marathi text summarization task. A significant improvement in the performance of multiple performance metrics is achieved owing to the stemming done by MTStemmer. The working of the proposed stemmer shows promising signs for the development of similar engines for other Indic languages.

Keywords: Stemming, Natural language processing, Text summarization, Marathi, Pre-processing, Low-resource NLP

1. Introduction

Natural language processing involves the modelling of human-readable sentences into mathematical form so as to automate a particular language understanding task. It is crucial that the modelling offers maximum retention of the semantic structure and meaning of the original data. Natural language processing (NLP) finds its application in information retrieval, data mining, knowledge extraction, linguistics, and other sub-domains of artificial intelligence. The two main sub-categories of NLP are natural language understanding (NLU) and natural language generation (NLG). NLU tasks involve analysing the provided text for reading comprehension. NLG involves generation of new text based on available data, including tasks like translation, summarization, and chatbots. In both types of tasks, a faster and effective conversion of text is desired. Stemming is one such method used for the effective resolution of words, especially as a pre-processing step. Stemming involves the conversion of the words back to their root form (Porter et al., 1980). Consider the word 'play', it has different forms like playing and played. While these forms are necessary for maintaining the grammatical correctness in human interactions, they do not add any extra value in terms of the meaning of the root word. Thus, a better approach would not consider these three words differently and would rather map them to their root word before modelling them (Dogra et al., 2013). Stemming does this job of stripping the words back to their root form.

In the past, multiple stemmers have been proposed, majorly for the English language (Porter et al., 1980; Porter, 2001). Recent years have also seen a rise in the stemmers being developed for non-English text, especially for Indian languages (Makhija, 2016; Kaur and Buttar, 2019; Kumar et al., 2020). Marathi is an Indian language derived from the Devanagari script and is spoken prominently in the Maharashtra state and nearby areas. While stemmers are available for Devanagari script and Hindi language, the work in the Marathi language has been paltry. Also, a transfer learning or fine-tuning based approach for stemming may turn out to be risky as there are some peculiar characteristics of the Marathi language.

In this paper, MTStemmer, a stemmer for the Marathi language is proposed, which performs stemming using a multilevel approach. The stemming is performed in two phases: In the first phase, the stemming in terms of auxiliary verb suffixes is done, while in the second phase, the gender-based suffixes are removed. Both these suffixes are not found to be adding significant value to the root meaning of the word. Through this paper, the following contributions have been made:

1. A new stemmer for the Marathi language has been created which takes into consideration all the nuances of the language and still offers better performance than existing solutions.
2. The proposed stemmer not just considers tense based suffix forms, but also gives importance to both gender-based suffixes and auxiliary verb-based suffixes which are prevalent in the Marathi language.
3. The benefit of using MTStemmer is demonstrated by showing the improvement in results obtained for the task of extractive text summarization when using the stemmer as a pre-processing step.

The outline of the paper is as follows: Section 2 explains the background work. Section 3 describes the challenges faced when working with Marathi language. The proposed methodology is explained in Section 4. Dataset description is provided in section 5 while the obtained results are shown in Section 6. Finally, the conclusion of the work is presented in Section 7.

2. Background and Related Work

The initial methods of natural language processing involved taking into consideration statistical features such as word count, position, and part of speech tagging, etc. which were then fed to either mathematical functions or machine learning algorithms. However, later years saw a rise in the emphasis given on pre-processing methods for improving the accuracy in the modelling stage. Gradually, methods like sequence padding, lemmatization, stop word removal and stemming started to become prevalent as common pre-processing steps.

The stemmer proposed by Martin Porter became one of the standard adopted stemmers in the community for the English language and is still used in many applications (Porter et al., 1980). Porter also went ahead to write Snowball, a language for stemming algorithms that tried to address the two main issues, lack of standard stemmers for non-English text, and the extrapolation by developers when implementing the Porter stemmer (Porter, 2001). Porter and Snowball stemmers continue to be the most used stemmers for the English language. Recent years have also seen a rise in the development of stemmers for non-English languages, especially those used in a linguistically diverse country like India (Harish and Rangan, 2020). Ramanathan and Rao proposed one of the first stemmers for Hindi which deployed a suffix stripping and lookup table approach (Ramanathan and Rao, 2003). Mishra et al. (Mishra and Prakash, 2012) combined suffix stripping with brute force to propose a stemmer “Maulik” for Hindi. Some work has also been done for inflectional languages (Paik and Paruj, 2008). Unsupervised learning has also been used for this cause (Husain, 2012). A hybrid approach was presented by Sharma et al. (Sharma et al., 2016) for stemming to improve the efficiency of information retrieval. Some previous works for sentiment analysis have also made use of in-built stemmers offered by contemporary transformer architectures (Malte and Ratadiya, 2019; Ratadiya and Mishra, 2019). The Hindi language is based on the Devanagari script and there have been efforts to make stemmers for some other dialects of the Devanagari script as well.

Makhija proposed an affix removal-based stemmer for the Sindhi language (Makhija, 2016). Recently, Nathani et al. (Nathani et al., 2020) presented an unsupervised learning-based stemmer to better the performance. Desai and Dalwadi came up with a stemmer for Gujarati text which involved the removal of both prefixes and suffixes (Desai and Dalwadi, 2016). A stemmer for verbs in the Punjabi language was developed by Kaur and Buttar which followed a rule-based approach (Kaur and Buttar, 2019). The first stemmer in Maithili was proposed by Priyadarshi and Saha using a hybrid approach (Priyadarshi and Saha, 2019). Barman et al. (Barman et al., 2019) came up with a similar rule-based stemmer for the Assamese language. For the Marathi language, the first stemmer was presented by Patil et al. (Patil et al., 2017) using a dictionary-based conflation. They also extended their work for morphological analysis-based stemming (Patil and Patil, 2017).

The last couple of years have seen further improvement in the range and depth of the work being done in Indic languages. Quite recently, Kumar et al. (Kumar et al., 2020) designed a rule-based stemmer for the Hindi language. Patil et al. went on to improve it and come up with a new stemmer which takes into consideration the stacking of both suffixes and prefixes (Patil and Patil, 2020). Adhikari and Neupane came up with a method to improve the performance on Nepali language by extending the use of Sanskrit words (Adhikari and Neupane, 2020). Gaikwad and Haribhakta (Gaikwad and Haribhakta, 2020) proposed the adaptive merger of GLoVe and Fasttext vectors for Hindi word embeddings. Kaur and Buttar also extended their previous stemmer work for Punjabi adjectives (Kaur and Buttar, 2020). Memon et al. (Memon et al., 2020) presented a thorough overview of the different truncating and statistical method-based stemming algorithms. Steps have been taken towards creating a Part of Speech tagger for Maithili language as well (Priyadarshi and Saha, 2020). (Shah et al., 2020) worked on Marglish – code mixed Marathi text in a way that improves the performance on opinion mining. With increasing work on various NLP tasks, efforts are also being taken to ensure that accountability of these models does not go unnoticed (Verma and Verma, 2020).

There are certain limitations in the previous work which leave ample scope for research in this domain. Firstly, dictionary or lookup table-based stemmers are large in size and also slow in terms of retrieval. Hybrid stemmers proposed in the past have primarily focused on brute force as one of the involved approaches, thus keeping the window open for manual errors (Dogra et al., 2013). In the Marathi language, the work has been quite limited. Further, there are certain challenges in using the existing approach for the Marathi approach, which are described in the next section.

3. Challenges When Dealing with The Marathi Language

Table 1. Examples of challenges when working with Marathi language

Challenge	Example	Desired behavior
Prefix terms to create opposites	उचित: Appropriate अनुचित: Inappropriate	Prefix should not be removed as meaning of the word changes completely
Prefix terms used independently	अनुकूल: Suitable प्रतिकूल: Challenging	Removal of prefixes will give -kul which does not have any meaning of its own
Gender based suffixes	ती खेळते: She plays तो खेळतो: He plays	The suffixes can be removed as they only indicate the gender.

There are some peculiar features of the Marathi languages which makes it difficult to deploy a cross-language transfer learning concept. This is owing to some language-specific grammar rules which need to be addressed accurately. Some of the challenges when processing Marathi language, especially for a stemmer are as follows:

3.1. Opposite words formed using prefixes

In Marathi, often the opposite of a word is formed by using a prefix term. Consider the word Uchit (appropriate). The opposite of this word is Anuchit (inappropriate) which can be seen is formed by appending the prefix 'An-'. Similarly, many other such prefixes are used to create contrasting terms. As a result, a stemmer that directly strips of prefixes may lead to a complete change in the meaning of the term. This problem is analogous to including the word" not" in the list of stop words when processing the English language.

3.2. Prefix terms used independently

The terms which are used as prefixes to create opposites are also found to be used independently in some words. Consider the previous example of 'An-'. The word 'Anukul' means suitable, but by removing the prefix, the remaining word does not have any meaning on its own. In this case, 'An-' does not act as a prefix. Thus, the removal of prefixes in a stemmer is not recommended as in some cases they create opposite effect while in some cases they are also an important part of the word itself and not just used for grammatical correctness.

3.3. Variation in suffixes based on gender and auxiliary verb tenses

Unlike English where verbs do not have different forms for genders, in Marathi there are suffixes appended to the verbs only to indicate the gender of the subject. These suffixes do not add much to the meaning of the action denoted by the verb and can be removed. While in English, the stemming takes place only based on the tense of the verbs, in Marathi, the stemmer should also take into consideration these gender-based suffixes to further improve the efficiency of the system (Patil et al., 2017).

Table 1 gives examples for each of these challenges to show the problems which need to be addressed by the stemmer. The proposed MTStemmer tries to tackle all these challenges while improving the results

4. Proposed Methodology

The proposed stemmer involves a two-stage stemming of the suffixes of the given word. As mentioned earlier, these two stages are dependent on the two types of suffixes which it intends to remove: gender-based suffixes and auxiliary verb-based suffixes.

4.1. Auxiliary verb-based stemming

In Marathi, many suffixes are added to the verb to indicate its tense or to ensure the grammatical correctness of the sentence. These suffixes could be added to a plethora of words and tokens, and it is a challenging task to determine the list of all possible suffixes. The aforementioned challenge of a suffix being used semantically also persists. To tackle this problem, the mapping of the length of the word to a list of suffixes is carried out. By Marathi grammar rules, it can be derived that for a particular length of words, there is only a specific set of characters that can act as a suffix of auxiliary verb.

Thus, for every token word, the length of the word is checked. Based on the length of the word, the terminal character of the word is compared with the mapping table of suffixes to decide whether the terminal character is to be stripped off or not. The list of suffixes is mapped not to a specific length, but concerning a length threshold. Table 2 denotes the list of suffixes based on this threshold of the length of the individual word. Remember that this mapping list is checked from top to bottom in the same sequence.

Table 2. List of auxiliary verb-based suffixes based on the length of the word

Word length 'L'	List of suffixes
L >5	शया
L >4	शे, शी, चा, ची, चे, हून
L >3	नो, तो, ने, नी, ही, ते, या, ला, ना, ऊण
L >2	◌े, ◌ी, स, ल, म, त, ◌ा

4.2. Gender-based suffix stemming

Once the verb-based stemming is done, a gender-based stemming strategy is deployed which is similar inflow as the previous one. The length threshold is mapped to a set of suffixes to determine the stemming order. Table 3 indicates the mapping for these gender-based suffixes, for whom the same sequence is followed as that of the rows in the table. In this case, there are more suffixes associated owing to multiple genders being addressed through these characters.

Table 3. List of gender-based suffixes based on the length of the word

Word length 'L'	List of suffixes
L >5	◌ुरडा
L >4	ढा
L >3	रु, डे, ती, ◌ान,, ◌ीण, डा, डी, गा, ला, ळा, या, वा, ये, वे, ती
L >2	◌ौ, ◌ै, ◌ा, ◌ी, अ, ◌े, ◌ि, ◌ु, ◌ू, त

For both these stemming levels, the word length considered is one less than its original length, to skip the terminal suffix character. After clearing the auxiliary suffixes, the gender suffixes are removed to obtain the final root form of the given word. Table 4 shows sample examples of stemming of some Marathi words using the proposed stemmer.

Table 4. Output of proposed stemmer for sample inputs

Original word	Stemmer output
खेळते	खेळ
जमीनीला	जमीन
विहिरीतून	विहिर
चमचाशी	चमचा
उठावे	उठा

The working of the proposed MTStemmer is described in Algorithm 1. While stemmer evaluation is done, it is also important to check its effectiveness on some tasks, especially those where stemming is looked at as an important preprocessing step.

4.3. Using stemmer as a preprocessing method

Stemming has always been used as an effective yet efficient preprocessing method for the modeling of data. While evaluating a stemmer by itself does not prove its effectiveness because the root words will always be obtained, the benefit of the stemmer can be checked by using it as a preprocessing step for a natural language processing task and comparing the impact on the obtained results. Figure 1 shows the flowchart of an input case in the system.

The proposed stemmer is used as a preprocessing step for the use case of performing the task of extractive text summarization. Extractive text summarization involves the ranking of sentences from the document and retaining only the top n% of those sentences as the summary of the document. There are various methods for performing extractive text summarization, but the most dominantly used technique is the textrank algorithm (Mihalcea and Tarau, 2004).

Textrank is a graph-based algorithm operating in an unsupervised way. The document sentences are used to calculate relative cosine similarity between each of them. For two sentences S_1 and S_2, the cosine similarity for their respective vectors is calculated as follows:

$$\text{Cosine similarity } (S1, S2) = (S1.S2)/(||S1||x||S2||) \quad (1)$$

Algorithm 1 Algorithm of MTStemmer

Input: Document D, Auxiliary suffix lookup table A (word_length, suffixes), Gender suffix lookup table G (word_length, suffixes)
Output: Stemmed Document D'

```
for each word w in D do
    word_len = len(w)
    if A(word_len) then
    w' = remove_auxiliary_suffix(w,A)
    end if
    updated_word_len = len(w')
    if G(updated_word_len) then
    w' = remove_gender_suffix(w',G)
    end if
    D'.append(w')
end for
return D'
```

A graph is constructed with each sentence indicating a node, and an edge indicating the co- sine similarity between the two sentences that it is connecting. Based on the user input, the top n sentences from this graph are then taken as the final summary of the document (Mihalcea and Tarau, 2004). Being an unsupervised method, the complexity is less and results are obtained immediately, thus making it a more reliable modeling technique.

5. Dataset Description

The extractive text summarization task is conducted on a news article dataset 1. The dataset consists of over 100 articles ranging on topics from the economy, finance, and politics. As it is an unsupervised method, the performance of the summarization technique with MTStemmer and without MTStemmer is evaluated over all these documents. A sample document text is shown in Table 5.

Table 5. Sample document text

नवी दिल्ली - देशाला आता अल्पबचत योजनांवरील व्याजदर कमी करण्यास सुरवात करावी लागणार आहे. व्याजदरात झालेली कपात न्याय्य असून हा दैनंदिन प्रक्रियेचा भाग असल्याचे सांगत केंद्रीय अर्थमंत्री अरुण जेटली यांनी या व्याजदर कपातीचे समर्थन केले आहे. अल्पबचत योजनांचे व्याजदर बाजारतील सध्याच्या दरांशी सुसंगत करण्यासाठी हा निर्णय घेतल्याचे त्यांनी

सांगितले.

The Textrank algorithm is run on these documents set two times- once without using MTStemmer, and the second time while using MTStemmer as a preprocessing step before forwarding it to the algorithm. The results obtained in both the instances across multiple performance metrics are elucidated in the next section.

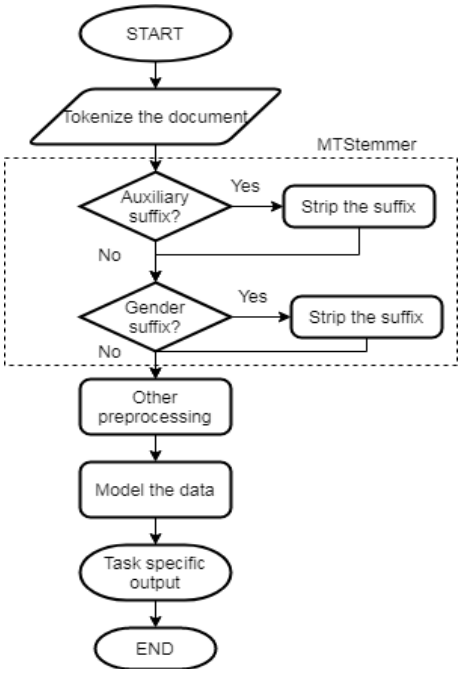

Figure 1. Flowchart of data in the complete modeling pipeline

6. Result and Analysis

For extractive text summarization, the ROUGE metric is considered as a standard performance metric. ROUGE takes into consideration the number of overlapping words present in the human summary and the predicted summary document. Considering this approach, the precision, recall, and F1 scores under ROUGE for a document *HS* and predicted summary *PS* are defined.

The F1 score considers harmonic mean of the previously mentioned metrics of precision and recall. Based on these metrics, the results obtained on the textrank algorithm are tabulated in Table 6. It can be seen that the use of the proposed stemmer has led to healthy gains of up to 2-3% in the performance of the system without changing any other configuration. To further solidify the claims, results are also evaluated on ROUGE-1, ROUGE-2, and ROUGE-L precision, which take into consideration the number of overlaps of one word (unigram), two words (bigram), and whole sentence respectively. The obtained results are indicated in Table 7.

Table 6. Comparison of performance on text summarization with and without using MTStemmer

Metric	**Textrank**	**MTStemmer (proposed) + Texrank**
Precision	0.596	**0.706**
Recall	0.722	**0.806**
F1 score	0.653	**0.753**

Table 7. Comparison of performance on precision across ROUGE variants

Metric	**Textrank**	**MTStemmer (proposed)+ Texrank**
ROUGE-1 precision	0.694	**0.712**

ROUGE-2 precision	**0.661**	0.653
ROUGE-L precision	0.707	**0.734**

While the proposed approach focuses on the development of stemmer, it has also been proved that the use of the proposed stemmer helps achieve improvement in performance across most of the variants of the ROUGE metric as well. A similar trend can be expected when using the proposed MTStemmer for other language processing tasks on other performance metrics. To further solidify the claims, the performance of the proposed stemmer is also compared with two other standard stemmers for Devanagari script. These include the Snowball stemmer2 and the Marathi stemmer package from the Indic stemmer library3. The comparison of the average precision, recall, and F-1 score over the 100 documents is tabulated in Table 8. It can be seen that the proposed stemmer has comfortably outperformed the two standard stemmers in this domain, owing to its consideration of various kinds of suffixes and appropriate handling of the nuances in the Marathi language. As a part of ablation studies, the results obtained by these three stemmers on F-1 score of ROUGE-1, ROUGE-2, and ROUGE-L metrics are shown in Table 9.

It can thus decisively be concluded that the proposed stemmer is a better alternative to the existing standard stemmers for Marathi language, as demonstrated from the results across multiple performance metrics for a sample task of extractive text summarization. A similar trend can be expected when extending the use case across other natural language understanding tasks.

Table 8. Comparison with performance of standard stemmers for extractive text summarization using Textrank approach

Metric	**Snowball stemmer**	**Indic stemmer**	**MTStemmer (proposed)**
Precision	0.689	0.692	**0.706**
Recall	0.762	0.761	**0.806**
F1 score	0.708	0.709	**0.753**

Table 9. Comparison of the performance of stemmers combined with Textrank approach in terms of various F-1 score metrics

F1 score	**Snowball stemmer**	**Indic stemmer**	**MTStemmer (proposed)**
ROUGE-1	0.663	0.663	**0.694**
ROUGE-2	0.602	0.605	**0.637**
ROUGE-L	0.682	0.684	**0.716**

For better interpretability and visualization of the results, we also present the comparisons of the MT Stemmer vs other stemmers, and MT Stemmer's standalone boost in text summarization approach in Figures 2, 3, 4, and 5 respectively.

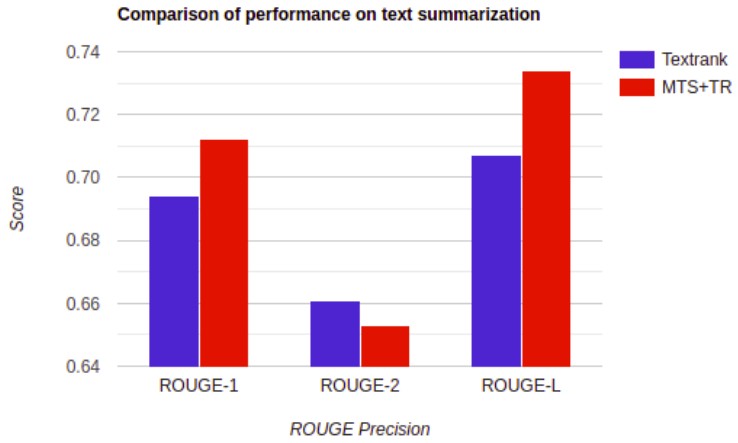

Figure 2. Comparison of ROUGE precision of regular textrank algorithm against addition of MTStemmer for extractive text summarization. MTS+TR indicates MTStemmer+Textrank combined approach.

2https://pypi.org/project/PyStemmer
3https://github.com/anubane/indic_stemming

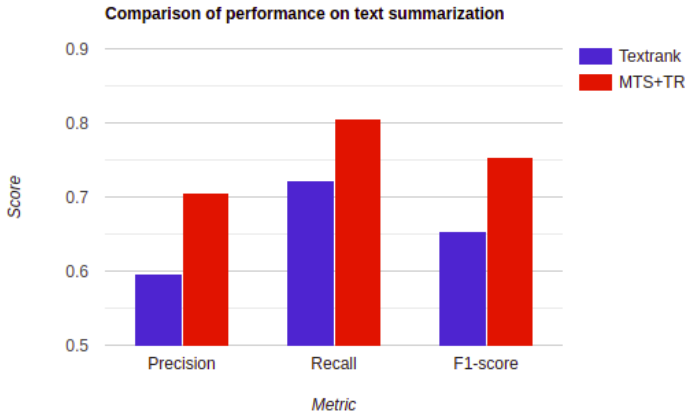

Figure 3. Comparison of performance on other metrics of regular textrank algorithm against addition of MTStemmer for extractive text summarization. MTS+TR indicates MTStemmer+Textrank combined approach.

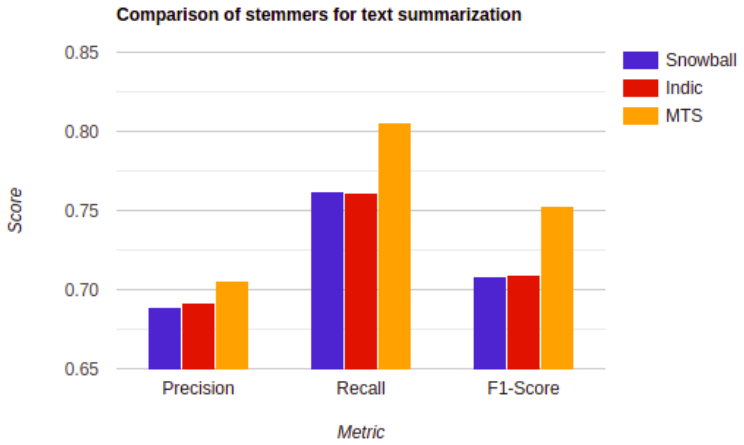

Figure 4. Comparison of performance on other metrics of MTStemmer against other stemmers for extractive text summarization.

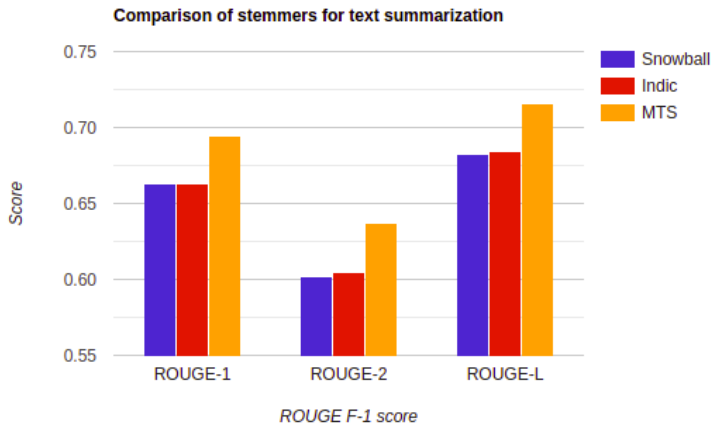

Figure 5. Comparison of performance on ROUGE F-1 score of MTStemmer against other stemmers for extractive text summarization.

7. Conclusion

A new stemmer for Marathi language, MTStemmer has been proposed in this paper. It deploys a multilevel suffix stemming approach using a lookup table for reducing the words to their root forms. The proposed stemmer is used as a preprocessing method for the use case of extractive text summarization and significant gains are observed in the performance of the system over multiple performance metrics. The proposed stemmer is lightweight in nature and still manages to be effective, while not violating the grammatical nuances of the Marathi language. Both kinds of suffixes, verb-based, and gender-based are addressed by the stemmer. Future work includes extending the use case of this stemmer for other language processing tasks such as sentiment analysis, question, and answering, etc. The use of contextual vectors for tokenization can also help improve the performance of the system for the mentioned text summarization task. As the digital world provides access to more data across multiple languages, efficient, accurate methods for processing and modeling such data will help extend the scope of smart systems to areas and domains where English or other global languages may not necessarily be used. The proposed MTStemmer marks a positive step taken in this direction for the Marathi language.

References

1. Adhikari, M., & Neupane, A. (2020). A vowel based word splitter to improve performance of existing Nepali morphological analyzers on words borrowed from Sanskrit. *Kathmandu University Journal of Science, Engineering and Technology*, *14*(1).
2. Barman, A. K., Sarmah, J., and Sarma, S. K. (2019). Development of assamese rule-based stemmer using wordnet. In *Wordnet Conference*, page 135.
3. Desai, N. and Dalwadi, B. (2016). An affix removal stemmer for gujarati text. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 2296–2299. IEEE.
4. Dogra, M., Tyagi, A., and Mishra, U. (2013). An effective stemmer in devanagari script. In *Proceeding of the International Conference on Recent Trends In Computing and Communication Engineering-RTCCE*, pages 22–25.
5. Gaikwad, V., & Haribhakta, Y. (2020). Adaptive GloVe and FastText Model for Hindi Word Embeddings. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD* (pp. 175-179).
6. Giri, V. V., Math, M. M., & Kulkarni, U. P. (2016). A Survey of Automatic Text Summarization System for Different Regional Language in India. *Bonfring International Journal of Software Engineering and Soft Computing*, *6*(Special Issue Special Issue on Advances in Computer Science and Engineering and Workshop on Big Data Analytics Editors: Dr. SB Kulkarni, Dr. UP Kulkarni, Dr. SM Joshi and JV Vadavi), 52-57.
7. Harish, B. S., & Rangan, R. K. (2020). A comprehensive survey on Indian regional language processing. *SN Applied Sciences*, *2*(7), 1-16.
8. Husain, M. S. (2012). An unsupervised approach to develop stemmer. *international journal on natural language computing*, *1*(2), 15-23.
9. Kaur, P. and Buttar, P. K. (2019). A rule-based stemmer for punjabi verbs.
10. Kaur, H., & Buttar, P. K. (2020). A RULE-BASED STEMMER FOR PUNJABI ADJECTIVES. *International Journal of Advanced Research in Computer Science*, *11*(6).

11. Kumar, R., Ramotra, A. K., Mahajan, A., and Mansotra, V. (2020). Design and implementation of rule-based hindi stemmer for hindi information retrieval. In *Smart Trends in Computing and Communications*, pages 115–122. Springer.
12. Makhija, S. D. (2016). A study of different stemmer for sindhi language based on devanagari script. In *2016 3rd International Conference on Computing for Sustainable Global Develop- ment (INDIACom)*, pages 2326–2329. IEEE.
13. Malte, A. and Ratadiya, P. (2019). Multilingual cyber abuse detection using advanced transformer architecture. In *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*, pages 784–789. IEEE.
14. Memon, S., Mallah, G. A., Memon, K. N., Shaikh, A. G., Aasoori, S. K., & Dehraj, F. U. H. (2020). Comparative Study of Truncating and Statistical Stemming Algorithms.
15. Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.
16. Mishra, U. and Prakash, C. (2012). Maulik: an effective stemmer for hindi language. *International Journal on Computer Science and Engineering*, 4(5):711.
17. Nathani, B., Joshi, N., and Purohit, G. (2020). Design and development of unsupervised stemmer for sindhi language. *Procedia Computer Science*, 167:1920–1927.
18. Paik, J. H., & Parui, S. K. (2008). A simple stemmer for inflectional languages. In *Forum for Information Retrieval Evaluation*.
19. Patil, H. B., Mhaske, N. T., and Patil, A. S. (2017). Design and development of a dictio nary based stemmer for marathi language. In *International Conference on Next Generation Computing Technologies*, pages 769–777. Springer.
20. Patil, H. B., & Patil, A. S. (2017, July). MarS: A rule-based stemmer for morphologically rich language Marathi. In *2017 International Conference on Computer, Communications and Electronics (Comptelix)* (pp. 580-584). IEEE.
21. Patil, H. B. and Patil, A. S. (2020). A hybrid stemmer for the affix stacking language: Marathi. In *Computing in Engineering and Technology*, pages 441–449. Springer.
22. Porter, M. F. (2001). Snowball: A language for stemming algorithms.
23. Porter, M. F. et al. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137.
24. Priyadarshi, A. and Saha, S. K. (2019). A hybrid approach to develop the first stemmer in maithili.In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pages 42–46.
25. Ramanathan, A. and Rao, D. D. (2003). A lightweight stemmer for hindi. In *the Proceedings of EACL*.
26. Priyadarshi, A., & Saha, S. K. (2020). Towards the first Maithili part of speech tagger: Resource creation and system development. *Computer Speech & Language*, *62*, 101054.
27. Ratadiya, P., & Mishra, D. (2019, November). An attention ensemble based approach for multilabel profanity detection. In *2019 International Conference on Data Mining Workshops (ICDMW)* (pp. 544-550). IEEE.
28. Shah, S. R., Kaushik, A., Sharma, S., & Shah, J. (2020). Opinion-mining on marglish and devanagari comments of youtube cookery channels using parametric and non-parametric learning models. *Big Data and Cognitive Computing*, *4*(1), 3.
29. Sharma, A., Kumar, R., and Mansotra, V. (2016). Proposed stemming algorithm for hindi information retrieval. *Int. J. Innov. Res. Comput. Commun. Eng.(An ISO Certif. Organ.)*, 3297(6):11449–11455.
30. Verma, P., & Verma, A. (2020). Accountability of NLP Tools in Text Summarization for Indian Languages. *Journal of Scientific Research*, *64*(1).