# TelStem:An Unsupervised Telugu Stemmer with Heuristic Improvements and Normalized Signatures

A.P. Siva Kumar[1], P. Premchand[2], A. Govardhan[3]
[1]Department of Computer Science and Engineering
JNTU, Anantapur, India
sivakumar.ap@gmail.com
[2]Department of Computer Science and Engineering
Osmania University
Hyderabad, India
p.premchand@uceou.edu
[3]Department of Computer Science Engineering
JNTUHCE
Nachupalli, India
govardhan_cse@yahoo.co.in

***ABSTRACT****: Stemming is a technique for reducing variant forms of a word to their roots (or stems) by enabling extraction of common suffixes. Stem need not correspond to the linguistic root of a word. Stemming is predominantly used in IR system to enrich retrieval effectiveness and to reduce the size of index for information retrieval task. This paper presents a systematic way of algorithm and implementation to develop an unsupervised Telugu stemmer using Take-All-Splits heuristic and improved paradigms. The performance of this stemmer is evaluated by taking two sets of 500 randomly extracted words. The trained Telugu corpus is taken from large set of documents containing 129066 words from CIIL Mysore. The accuracy is found to be 85.40% after applying normalization heuristic. The F-score is 92.94%, the recall is 91.97% and the precision is 96.91%. As the algorithm does not require any language specific information, it can be applied to other languages that are morphologically rich. The percentage reduction in index size for Telugu information retrieval task is also evaluated before and after normalization heuristic.*

**Keywords:** Stemming, Corpus, Precision, Recall, Unsupervised, Morphology

**Received:** 19 October 2010, Revised 19 November 2010, Accepted 28 November 2010



## 1.Introduction

A large volume of text is available in electronic form in multiple languages including Telugu which is an Indian language, spoken by more than 50 million people in the country. Linguist experts conveyed that a single Telugu word verb may be available in different forms. Language is very rich in literature and morphology, and it requires advancements in computational approaches. To access this information there is a need of multi-lingual (cross-lingual) text retrieval system. Language processing system has become an active area of research due to this. Earlier a lot of research was done on English. However, from the last decade research on Asian language processing has come into action. There are so many tools and other lexical resources available for English and other major European languages. No such standard tools are available for Indian languages. This paper focuses on development of one such tool, namely stemmer for Telugu language.

Stemming is a process that associates morphological variants of a word. Stemmers are used in query systems, indexing, web search engines, machine translation and information retrieval systems. Stemming also offers the benefits of minimizing storage requirements by eliminating redundant terms and reduces word forms to approximation of a common morphological root called stem. Thus, the relevant documents can be retrieved with improved precision and recall even if the morphemes of the query words are present in the documents.

The stem is not necessarily a valid linguistic root of the word. It is used to map the variant forms of the same stem. For example, the words పెలుతునాం, పెలాలి, పెలాము are reduced to same stem, పెల.

As explained earlier, Telugu words may have many morphological variants. For example,అధికారము, అధికారాలు, అధికారి, అధికారుడు are reduced into అధికార . These variants are mostly generated by adding suffixes to the root word. For example, Telugu plural nouns are generated by adding suffixes like ◌ం, లు, రు, ము Similarly, following inflected wordsరామ, రాము, రాముడు, రాములు are reduced to same stem రామ

English and other European languages have standard stemmers but Telugu and most of the other Indian languages do not have such standard stemmers. The most widely known stemmers have been developed by Lovins and Porter for English. Both Porter's and Lovin's stemming algorithms depend on linguistic knowledge.

Mainly there are two approaches to design stemming algorithm. They are linguistic approach which is based on the prior knowledge of the morphology of the specific language or non-linguistic approach which does not need any prior knowledge of morphology. The linguistic approaches are likely to be more powerful because the quality of the morphological analysis has been guaranteed by linguistic experts but it is hard to develop and codify that type of rules if a new language is to be added to an information retrieval system. This process is a time taking process and it is not always possible to have an expert for each language. In recent times, corpus based approaches without any linguistic knowledge have been emerged as promising alternatives to traditional linguistic approaches. Therefore, this paper focuses on the development of unsupervised approach which only relies on corpus for learning powerful suffixes.

This paper explains sequence of steps needed to develop an unsupervised Telugu stemmer. Proposed paradigm uses set of words to derive powerful suffixes that are used in stemming of strange words. Telugu corpus is taken from CIIL Mysore. This Telugu corpus consists of 129066 words. Normalization heuristic is applied to drop and to refine set of suffixes to avoid over stemming. In order to test the performance of proposed stemmer, two sets of 500 words are selected from Telugu daily newspapers and these words are manually segmented for creation of actual root. Two different experiments are conducted. First experiment shows the performance of proposed stemmer in terms of accuracy, precision, recall and F-score. Two test runs are performed in the first experiment. Accuracy, F-score are observed as 85.40% and 92.94% respectively. The performance of proposed stemmer is evaluated before and after normalization heuristic. By refining the suffixes with normalization heuristic the performance is improved. In the second experiment the performance of proposed stemmer is evaluated for the indexing task. The percentage reduction in index size is measured before and after applying normalization heuristic. Results show that proposed stemmer outperforms after applying normalization heuristic. The proposed paradigm does not rely on the linguistic knowledge. Hence, this approach can be used for other languages that are morphologically rich. Proposed paradigm is language independent and computationally inexpensive. Thus this approach is known as unsupervised. Before applying normalization heuristic the number of suffixes in the dropped list is 2746, after applying normalization heuristic the number of suffixes are 1583. Longest suffix matching algorithm is used for stemming of strange words.

The rest of the paper is organized as follows: In the section two previous works on stemming algorithms is explained. Proposed paradigm to develop Telugu stemmer is described in section three. Section four presents experimental investigation. Section five clarifies the accomplishments. Section six concludes the paper.

**2. Associated Work**

From the last five decades many stemming algorithms were proposed and implemented to improve information retrieval task. First stemming algorithm was written by Julie Beth Lovins of Massachusetts Institute of Technology in1968. This is single pass, context sensitive and uses longest suffix stripping algorithm. Lovins stemmer maintains 250 suffixes. These suffixes were used in stemming of strange words. This stemmer was remarkable for its early date and had great influence on later work in this area.

Later Martin Porter designed and implemented stemming algorithm for English language at the University of Cambridge in 1980. It is a five step procedure and some rules were applied under each step. If word is matched with suffix rule, then the condition attached to that rule is executed to get the stem. Porter stemmer is a linear process. It is widely used as it is readily available. J.L. Dawson of the Literary and Linguistics Computing Centre at Cambridge University developed a stemmer that is based on Lovins stemmer, by extending the suffix list to 1200 and modifying recoding rules. Bob Krovertz developed another stemming algorithm at the University of Massachusetts, in 1993. Krovertz stemmer is based on morphology and uses dictionary.

The techniques used in these stemmers were rule based. Preparing such rules for a new language is time consuming and also these techniques are not applicable to morphologically rich languages like Telugu.

Later unsupervised approaches came into picture. Brent et al proposed unsupervised morphology technique based on Minimum Description Length (MDL). This is based on Bayesian model for English and French languages. Goldsmith uses MDL framework to learn unsupervised morphological segmentation of European languages. Mathias Creutz presented language independent and unsupervised algorithm for word segmentation that is based on prior distribution of morpheme length and frequency. Freitag proposed stemming algorithm based on automatic clustering of words using co-occurrence information.

Previous work on morphology of Indian languages includes lightweight stemmer for Hindi, statistical stemmer for Hindi, unsupervised stemmer for Hindi, unsupervised morphological analyser for Bengali, hybrid stemmer for Gujarati and rule based Telugu morphological generator (TelMore) for Telugu. Hybrid stemmer is not completely unsupervised. This stemmer uses hand-crafted suffix list. TelMore generates the morphological forms of verb and noun by using the rules of C.P. Brown and H. Krishnamurthy.

Word segmentation heuristic is based on Goldsmith's approach. This heuristic is based on split-all method that uses Boltzmann distribution. After applying normalization heuristic, the performance of proposed stemmer is found to be increased. The proposed paradigm presented in this paper focuses on the development of an unsupervised Telugu stemmer for effective Telugu information retrieval task.

**3. Proposed Paradigm**

Proposed paradigm is moderately in Goldsmith's approach. It is based on taking all splits of each and every word from corpus. Goldsmith's heuristic does not require any linguistic knowledge, thus it is completely unsupervised. For unsupervised studying, words from Telugu corpora are considered. To refine the learned suffixes normalization heuristic is applied. Each word from corpus is considered in l-1 different ways, splitting the word into stem + suffix after i letters, where 1<=i<=l -1 and l is length of the word. Then frequencies of stems and suffixes of each split of words are computed. These frequencies are used to get optimal split of word. The optimal segment unifies stem and suffix. Making analogous signatures and removal of spurious signatures are applied to get regular suffixes. Figure 1 shows the layout of proposed paradigm.

The details of the proposed paradigm are described below:

**3.1 Word Segmentation**

Telugu corpus is given as input to this process. Corpus consists of unique words. 129066 words are considered as corpus. Take-all-splits heuristic is used for word segmentation and it uses all cuts of a word of length 'l' into stem+suffix $w_{1,i} + w_{i+1}$,l, where 'w' refers to the word and $1 = i \leq l$. This heuristic assigns a value to each split of word 'w' of length 'l': $w_1$, i +$w_{i+1}$, l. This heuristic takes all splits of a word wl into stems and suffixes as:

$$w_1 = \{stem_{11}, suffix_{21}, stem_{12}, suffix_{31}, ..., stem_{1i}, suffix_{1l}\}$$

Where $stem_{1i}$ refers to the prefix of word $w_1$ with length i and $suffix_{i+1l}$ refers to the suffix of same word with length l-i. For example, the word అధికారము is split into following stems and suffixes:

'అధికారము' { అ, ధికారము; అధ, ికారము; అధి, కారము; అధిక, ారము; అధికా, రము; అధికార, ము; అధికారమ, ు }

Next, stems and suffixes are stored in database and heuristic value of each split is calculated by using following equation 1:

$$\text{Heurisitc value at split i} = i \log \text{freq}(\text{stem}=w_{1,i}) + (l - i) \log \text{freq}(\text{suffix} = w_{i+1,l}) \quad (1)$$

As the value of i changes the heuristic value also changes. Heuristic value mainly depends upon the frequency of stem and suffix in corpus. The frequency of stem and suffix are varies as the length of stem and suffix is increases. The frequencies of shorter stems and suffixes are very high when compared to the slightly longer stems and suffixes. Thus the multipliers i (length of stem) and l-i (length of suffix) are introduced in this heuristic in order to compensate for this disparity.

### 3.2 Procuring Optimal Split

The output of first heuristic is set of heuristic values for each word. The split with highest heuristic is treated to be optimal split of the word. This segmentation mainly depends upon the frequency distribution of stems and suffixes of the word, language and corpus. Some restrictions are considered while taking optimal split. First, the word length is restricted to a minimum of three. Following this condition small words like ఏం, అంక, అంగ, అంచ cannot be split. The second restriction is based on heuristic value of each split. If any heuristic value of the split is zero, it is not taken into account. By this restriction lengthy word like అంగసంపదకార్మికులు, అంకురాద్యవస్థయందే are not considered.

The characterization of heuristic values of all splits of word 'అక్షరమ్యత' is shown in Figure 2.

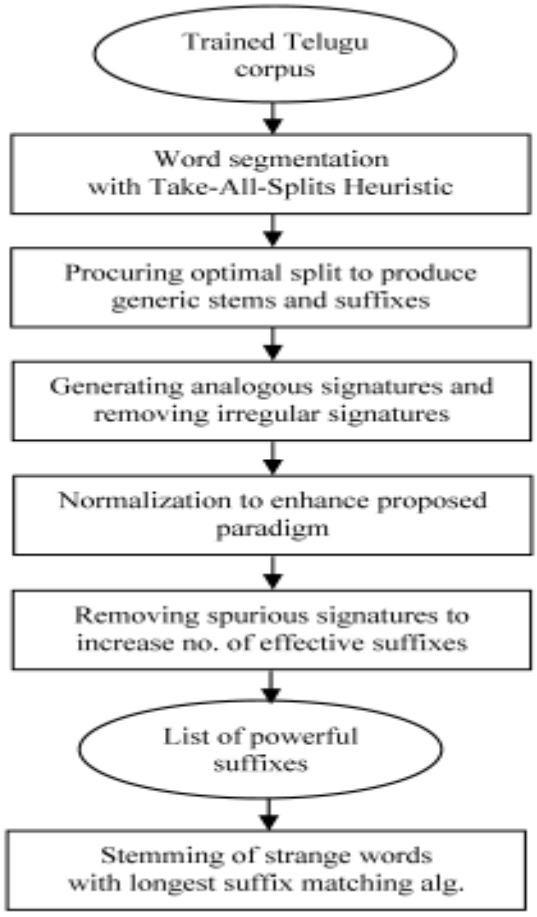



Figure 1. Proposed paradigm

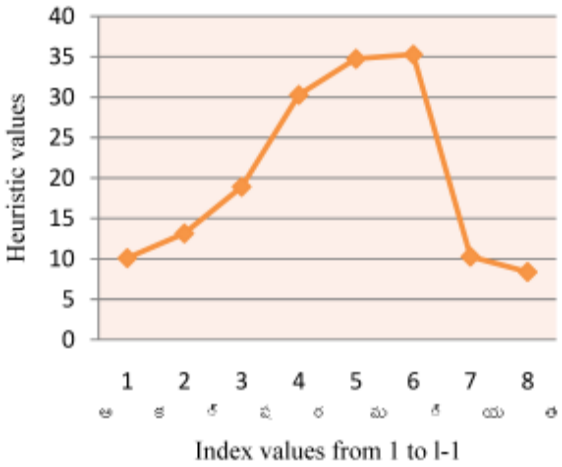



Figure 2. Heuristic values of all splits of word

Index value is considered in the x-axis and heuristic values are considered in the y-axis. By looking into graph, the heuristic values of word is increased first and then decreased. The point where the heuristic value starts decreasing is considered as optimal split of the word. As the stem length increases up to a certain limit, the frequency of stem starts decreasing. That is the reason why heuristic value starts decreasing. This step gives the generic stems and generic suffixes of given corpus.

**3.3 Analogous Signatures**

An optimal split is assigned to every word of corpus into generic stem and generic suffix by the above step. After getting generic stems and generic suffixes, the suffixes that are having same stem are grouped together. This is called stem's signature. This structure is shown below:

$$\{stem_1\} \begin{Bmatrix} suffix_1 \\ suffix_2 \\ suffix_3 \\ suffix_4 \end{Bmatrix}$$

Where suffix1, suffix2, suffix3…. are having similar stem, stem1. By considering Telugu word example, కుండా, వచ్చు, తారు suffixes have same stem అంతరించిపో. By grouping suffixes with same stem is shown below:

$$\{\text{అంతరించిపో}\} \begin{Bmatrix} \text{కుండా} \\ \text{వచ్చు} \\ \text{తారు} \end{Bmatrix}$$

Signature can also be referred to the set of suffixes along with the associated set of stems. Stems that are having the same set of suffixes are grouped. Here stem1, stem2 have the same set of suffixes suffix1, suffix2, suffix3 and suffix4. The presence of signature structure is shown below:

$$\begin{Bmatrix} stem_1 \\ stem_2 \end{Bmatrix} \begin{Bmatrix} suffix_1 \\ suffix_2 \\ suffix_3 \\ suffix_4 \end{Bmatrix}$$

Here there are at least two members present in each set, and all combinations present in this structure are available in the corpus and each stem is found with no other suffix but suffix may well appear in other signatures.

By considering Telugu words example,అంతస్తు, అంతరిక్షనౌక and అకాడమీ have same set of suffixes లో, లు and కి. So by grouping set of stems that are having same set of suffixes is shown below:

$$\begin{Bmatrix} \text{అంతస్తు} \\ \text{అంతరిక్షనౌక} \\ \text{అకాడమీ} \end{Bmatrix} \begin{Bmatrix} \text{లో} \\ \text{లు} \\ \text{కి} \end{Bmatrix}$$

All signatures that are associated with only one stem and only one suffix are discarded. These are called irregular signatures. The suffixes that are generated with these set of signatures can be used directly in stemming of strange words.

**3.4 Normalization**

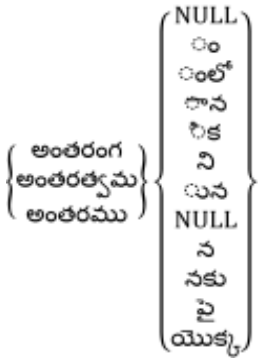
It is perceived that in some of the analogous signatures, set of suffixes having similar stem involve some common set prefixes. Thus, to enhance the performance of proposed paradigm, another heuristic is applied. This heuristic normalizes the prefixes of suffixes that are common to same stem. This is shown by the following expressions:

$$suffix_i = a_1b_1suffix_1$$
$$suffix_j = a_1b_1suffix_2$$
$$suffix_k = a_1b_1suffix_3$$
$$suffix_l = c_1d_1e_1suffix_4$$
$$suffix_m = c_1d_1e_1suffix_5$$

$Suffix_i$, $suffix_j$, $suffix_k$, $suffix_l$ and $suffix_m$ are having same stem $stem_i$. The application of this heuristic concatenates $stem_i$ with prefixes of suffixes $a_1b_1$, $c_1d_1e_1$ and modifies stem and suffixes. The stems and suffixes after applying normalization heuristic are $stemia_1b_1$, stemic1d1e1, $suffix_1$, $suffix_2$, $suffix_3$, $suffix_4$ and $suffix_5$.

The normalization process by considering Telugu words is displayed below:


After applying normalization heuristic to the word 'అంతర' the stems and suffixes is displayed below:

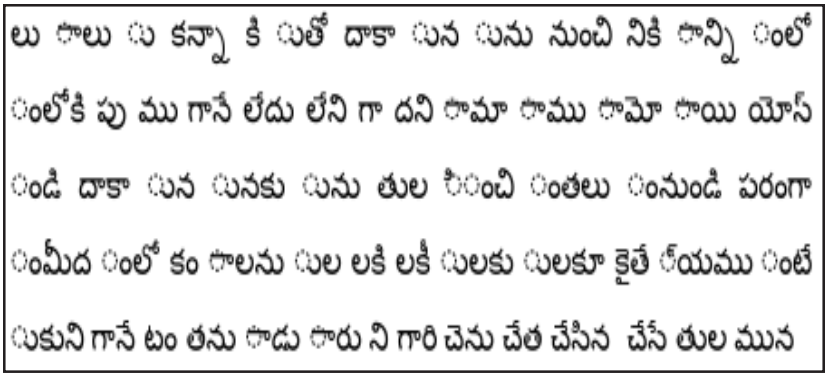



The suffixes that are obtained after normalization process are refined suffixes. This heuristic is applied to control over stemming. These refined suffixes can be used in stemming of strange words.

**3.5 Removing Spurious Signatures**

The output of the above step gives set of signatures. Again all signatures that are associated with only one stem and only one suffix are discarded. These signatures are called spurious signatures. To enrich the performance of proposed paradigm these signatures are removed. The remaining signatures are called refined signatures and the suffixes that are dropped from them are called refined suffixes. The refined suffixes are not the suffixes that can be used to establish the morphology, but they are very good for approximation and useful for stemming process.

Some of refined suffixes produced by proposed paradigm are shown below:



### 3.6 Stemming of Strange Words

The output of unsupervised paradigm is a list of powerful suffixes. Here two sets of suffixes are available. First suffix list is dropped before normalization heuristic. Second suffix list is generated after applying normalization process. Each suffix approximately seizes some morphological variation. The longest suffix matching algorithm is used in stemming of strange words. To alleviate stemming of strange words the suffixes are organized in decreasing order of their lengths. The longest possible suffix of the strange word that is matching with some suffix in list is dropped. This method is called longest suffix matching algorithm. Matching is started from first suffix to last suffix, wherever a match is found, the strange word is segmented into stem and suffix. Figure 3 shows the sample output of proposed paradigm.

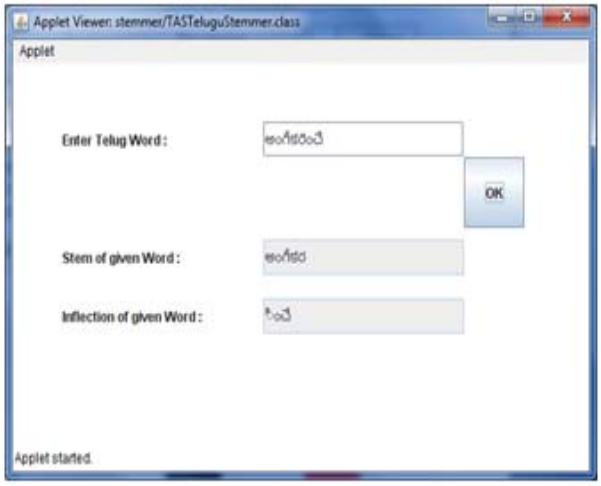



Figure 3. Sample output of proposed paradigm

### 4. Experiments

There is always an argument on effectiveness of stemming; Frakes reported that different outcomes and results are possible for stemming process. Harman reported that influence of stemming on overall performance is little, if effectiveness is measured by traditional precision and recall measures. It is true, if language is with simple morphology like English. But Krovertz, Popovic and Willett proved that stemming process significantly improves the retrieval effectiveness, mainly precision, for short queries or for languages with high complex morphology, like romance language.

Mainly the performance of a stemmer is measured in terms of its contribution to enrich the performance of an IR system. In order to evaluate proposed unsupervised Telugu stemmer, two different experiments were conducted. The first experiment evaluates the performance of stemmer in terms of accuracy, precision and recall. In the second experiment, the performance of stemmer is measured in terms of its capability to reduce the size of index in an information retrieval task. For these two experiments, two sets of suffixes are used. The performance of proposed stemmer is evaluated and compared before and after normalization heuristic. The training data is extracted from CIIL Mysore containing 129066 words from 200 documents. Table I shows an overview of characteristics of the trained corpus. Two different test runs are produced by randomly extracted words from Telugu daily newspapers.

Experiment 1: To measure the performance of this stemmer accuracy, recall, precision and F-score metrics are considered in the first experiment. Two different test runs run-1 and run-2 are performed. The evaluation metrics used in this experiment are described below:

Accuracy of proposed stemmer is defined as number of words stemmed correctly. This is calculated by comparing manually stemmed test words and stems of test data that are produced by proposed stemmer.

Recall is defined as the ratio of length of stem generated by proposed stemmer and the length of the actual stem that is identified manually. If the length of stem that is produced by the proposed stemmer is less than the length of actual stem then the recall is treated as 100% because stem produced by proposed stemmer is part of the actual stem. However, these types of words decrease

the precision value because the actual stem is having some extra characters. Mathematically, recall is calculated using the below equation 2 or 3:

$$\text{Recall} = 1, \text{if length of stem produced by proposed paradigm} < \textit{length of actual stem} \quad (2)$$

$$\text{Recall} = \frac{\sum_{i=1}^{n} \frac{\text{length of actual stem}}{\text{length of stem produced by proposed paradigm}}}{n}, \text{otherwise} \quad (3)$$

Precision is defined as the ratio of length of the actual stem that is identified manually and the length of stem generated by the proposed stemmer. If the length of stem that is produced by proposed stemmer is greater than the length of actual stem then precision is treated as 100% because all the characters present in actual stem are also present in the stem that is produced by proposed stemmer. Mathematically, precision is calculated using below equation 4 or 5:

$$\text{Precision} = 1, \text{if length of stem produced by proposed paradigm} > \text{length of actual stem} \quad (4)$$

| Description | Count |
|---|---|
| Total documents | 200 |
| Unique words | 129066 |
| Stem's signatures before normalization | 29501 |
| Unique suffixes before normalization | 22541 |
| Regular suffixes before normalization | 2746 |
| Stem's signatures after normalization | 42273 |
| Unique suffixes after normalization | 14834 |
| Refined suffixes after normalization | 1583 |

Table1. Trained Telugu corpus

$$\text{Precision} = \frac{\sum_{i=1}^{n} \frac{\text{length of actual stem}}{\text{length of stem produced by proposed paradigm}}}{n}, \text{otherwise} \quad (5)$$

Where n is total number of test words.

Harmonic mean of recall and precision is defined as F-score. Mathematically F-score is represented using the below equation6:

$$\text{F- Score} = \frac{(2*\text{Precision}*\text{recall})}{\text{Precision} + \text{recall}} \quad (6)$$

The two test runs of proposed stemmer before and after normalization are shown in Table II and Table III respectively.

| Test Data | Accuracy | Precision | Recall | F-score |
|---|---|---|---|---|
| **Run-1** | **34.80** | **82.90** | **96.92** | **89.37** |
| **Run-2** | **50.20** | **88.52** | **95.11** | **91.70** |

Table 2. Test results before normalization

| Test Data | Accuracy | Precision | Recall | F-score |
|---|---|---|---|---|
| **Run-1** | **69.89** | **93.94** | **91.97** | **92.94** |
| **Run-2** | **85.40** | **96.91** | **89.14** | **92.86** |

Table 3. Test results after normalization

Experiment 2: The performance of proposed paradigm in terms of reduction in index size of Telugu information retrieval is

measured in this experiment. For this experiment, documents from Telugu corpus are taken. The difference between number of index terms with and without stemming is considered as reduction in index size. Table IV describes the comparison of percentage reduction in index size of proposed stemmer before normalization and after applying normalization heuristic.

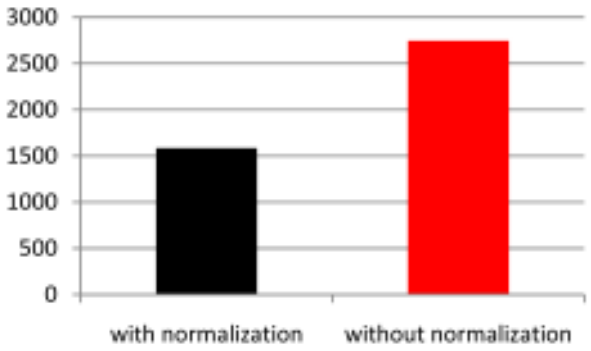



Figure 4. No of suffixes

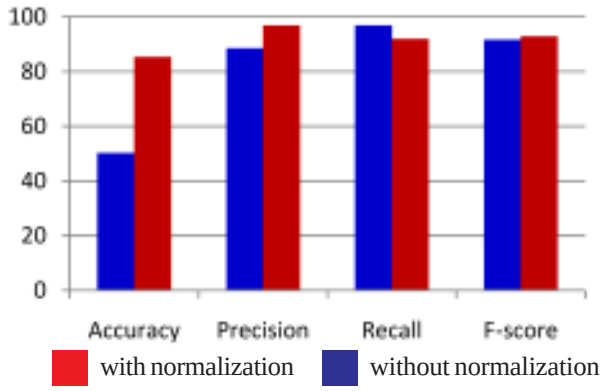



Figure 5. Compares the number of suffixes dropped before and after normalization heuristic

| Test parameter | No. of words without stemming | Before normalization heuristic | After normalization heuristic |
|---|---|---|---|
| No. of words | Reference | 77.14% | 67.24% |
| Index size(MB) | Reference | 83.68% | 76.05% |

Table 4. Percentage reduction in index size

The comparison of index size in terms of number of words with and without normalization heuristic is shown in Figure 6.

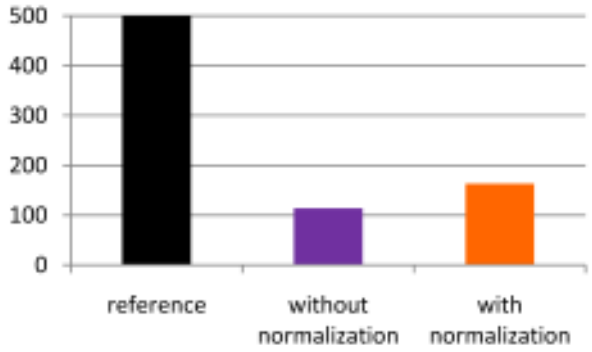



Figure 6. Index size comparison

**5. Accomplishments And Conversation**

Table 2 describes that maximum accuracy is found to be 50.20% and F-score is found to be 91.70% in run-2. Table 3 shows that the maximum accuracy is 85.40% in run-2 and F-score is 92.94% obtained after applying normalization heuristic in run-1. By seeing all test runs in terms of accuracy, precision and F-score, it is observed that the proposed stemmer performs better after normalization heuristic. Recall is high before normalization heuristic when compared to recall after normalization heuristic. The recall and precision values in two test runs indicate that over stemming is the main cause of difference in accuracy. This accomplishment is achieved with large set of refined powerful suffixes.

The result of reduction in the index size is shown in Table IV. The index size without stemming is considered as baseline.

According to the number of index terms the reduction is found to be 77.14% before normalization. 67.24% reduction is obtained after applying normalization. By considering index size (measured in MB), the percentage reductions are found to be 83.68% and 76.05% before applying normalization and after applying normalization respectively.

**6. Conclusion and Future work**

The results show that proposed paradigm outperforms after applying normalization heuristic. Proposed stemmer is unsupervised and language independent. Corpus is used to derive set of powerful suffixes and it does not need any linguistic knowledge. Hence, this approach can be used for developing stemmers of other languages that are morphologically rich.

The performance of proposed paradigm is evaluated in terms of accuracy, precision, recall and F-score and compared before and after normalization heuristic. Ability to reduce index size of information retrieval task is also measured by proposed stemmer before and after applying normalization. This shows that the percentage reduction in the index size is better for the proposed stemmer before normalization.

The most widely accepted measure to evaluate performance of stemmer by using IR system is Mean Average Precision (MAP) at overall relevant retrieved documents over fraction of retrieved documents. The effectiveness of proposed stemmer is not evaluated in terms of MAP due to unavailability of IR system and other resources for Telugu language. This work will be done in future.

**References**


[1] Amaresh Kumar Pandey, Tanveer J Siddiqui. (2008). An Unsupervised Hindi stemmer with heuristic improvements. *In: Proc. of the second workshop on Analytics for noisy unstructured text data*. New York: ACM.

[2] Sunitha, K.V.N., Kalyani, N. (2009). A Novel approach to Improve rule based Telugu Morphological Analyser. *In:* Proc. of 2010 Second World Congress on Nature and Biologically Inspired Computing. Japan: IEEE.

[3] Michela Bacchin. (2005). A probabilistic model for stemmer generation. *In:* Proc. of Information Processing and Management: an International Journal - Special issue: An Asian digital libraries perspective, 41 (1) 121-137. New York: ACM.

[4] Goldsmith, J. (2001). Unsupervised Learning of the Morphology of a Natural Language. *In:* Proc. of Computational Linguistics (p. 153-198). Chicago: Department of Linguistics, University of Chicago.

[5] Goldsmith, J. (2000). Lingouistica: An automatic morphological analyser. *In:* Proc. from the Main Session of the Chicago Linguistic Society's Thirty-sixth Meeting. p. 153-198. Chicago: Chicago Linguistic Society.

[6] Creutz, M., Lagus, K. (2005). Unsupervised Morpheme Segmentation and Morphology Induction from Text Corpora Using Morfessor 1.0.Tech. *In:* Proc. of ACM Transactions on Speech and Language Processing (TSLP). New York: ACM.

[7] Akshar Bharati, Rajeev Sangal, Bendre, S.M., Pavan Kumar, M.N.S.S.K.., Aishwarya. (2001). Unsupervised Improvement of Morphological Analyzer for Infectionally Rich Languages. *In:* Proc. of the Natural Language Processing Pacific Rim Symposium (NLPRS) (p. 685- 692). Tokyo: NLPRS.

[8] Santhosh Kumar, M., Kavi Narayana Murthy. (2007). Corpus-based Statistical Approaches for Stemming Telugu. *In:* Proc. of National Seminar on creation of Lexical Resources for Indian Language Computing and Processing. Mumbai: C-DAC.

[9] Douglas W. Oard, Gina-Anne Levow, Clara I. Cabezas. (2000). Statistical Stemming and Backoff Translation. *In:* Proc. of the Workshop on cross-language evaluation forum (CLEF) (p. 176-187), Portugal: CLEF.

[10] Jagadeesh, J., Kumaran, A. (2007). Cross-Lingual Information Retrieval System for Indian Languages. *In:* Proc. of Advances in Multilingual and Multimodal Information Retrieval (p. 80-87), Volume 5152/2008. Verlag Berlin: Springer.

[11] Mathias Creutz. (2003). Unsupervised segmentation of words using prior distributions of morph length and frequency. *In:* Proc. of ACL-03, the 41st Annual Meeting of the Association of Computational Linguistics p. 280-287. Japan: ACL.

[12] Ganapathiraju, Madhavi, Lori Levin. (2006). TelMore: Morphological Generator for Telugu Nouns and verbs. *In:* Proc. of Second International Conference on Universal Digital Library Alexandria. Egypt: ICUDL.

[13] Pratikkumar Patel Kashyap Popat, Pushpak Bhattacharyya. (2011). Hybrid Stemmer for Gujarati. *In:* Proc. of the 23rd International Conference on Computational Linguistics (COLING) (p. 51-55). Beijing: ICCL.

[14] Majumder, Prasenjit, Mandar Mitra, Swapan, K., Parui, Gobinda Kole, Pabitra Mitra, Kalyankumar Datta. (2007). YASS: Yet another suffix stripper. *In:* Proc. of Association for Computing Machinery Transactions on Information Systems (p. 18-38). New York: ACM.

[15] Prasad Pingali, Vasudeva Varma. (2006). Hindi and Telugu to English Cross Language Information Retrieval. *In:* Proc. of Working Notes of Cross Language Evaluation Forum Workshop (CLEF). Spain: CLEF.

[16]Julie Beth Lovins. (1968). Development of a stemming algorithm. *In: P*roc. of Mechanical Translation and Computational Linguistics.

[17] Martin Porter. (1980). An algorithm for suffix stripping program, *In:* Proc. of British Library Research and Development Report (pp 130H137). London: British Library.

[18] Krovertz. (2000). Viewing morphology as an inference process. *In:* Proc. of the 16th annual international ACM SIGIR conference on Research and development in information retrieval. New York: ACM.

[19] Snover, M.G., Brent, M.R. (2001). A Bayesian model for morpheme and paradigm identification, *In: Proc. of the 39th Annual Meeting on Association for Computational Linguistics* (pp. 482-490). USA: ACL.

[20] Ramanathan, A., Rao, D. (2003). A lightweight stemmer for Hindi, *In:* Proc. of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL), on Computational Linguistics for South Asian Languages Workshop.