An Affix Removal Stemmer for Natural Language Text in Nepali

Abhijit Paul
Research Scholar
Department of Computer Science
Assam University Silchar, India

Arindam Dey
Research Scholar
Department of Computer Science
Assam University Silchar, India

Bipul Syam Purkayastha
Professor
Department of Computer Science
Assam University Silchar, India

ABSTRACT
Stemming is the prerequisite step in Text Mining, Spelling Checker applications as well as a basic requirement for Natural Language Processing (NLP) tasks. Also it is very important in most of the Information Retrieval (IR) systems. This paper describes an affix stripping technique for finding out the stems from context free text in Nepali Language using lexical lookup based and rule based approach. It starts by introducing different types of lexicon, the basic unit of Nepali stemmer and few rules to identify the word in the lexicon. These rules and lexicons are applied in the design and implementation of an extensible architecture of a stemmer system for Nepali text. Finally designed stemmer performance is evaluated over different domains of 1,800 words. These domains include news on Economics, Health & Political in Nepali language, which are based on Devanagari Script. The overall accuracy of the designed system is 90.48%. Due to the absence of extensive linguistic resources, this technique shows improvement in the performance over simple rule based system.

Keywords
Stemmer; Lexicon; NLP; Text Mining; Spelling Checker; IR

1. INTRODUCTION
Natural Language Processing is a field of Computer Science, Artificial Intelligent and in linguistic concerned; it is an interaction between computer and human (natural) languages. Stemming is one of the important applications of Natural Language Processing. It is a process by which a word is split into its stem i.e. root and affixes [1] without doing complete morphological analysis. Word stemming is useful for indexing and search systems. Indexing and searching are key concept of Text Mining applications and IR systems. It is also used to improve the performance of spelling checkers where morphological analysis would be computationally expensive. A stemmer can also reduce the size of a dictionary which is the main feature to use a stemmer in spelling checker applications for mobile and other handheld device.

Nepali **(नेपाली)** is the national language of Nepal. Nepali is spoken in Nepal and some parts in India. It is a language which takes its root from Sanskrit, the classical language of India [2]. Nepali is a morphologically rich language [3], also inflected and one has to consider many features to build a stemmer for such language. And in this paper context free words are taken into consideration for stemming. Context free words are those which are not depend on the context of the sentence. Context based technique like Lemmatizer is there which deals with the complex process of first understanding the context, then determining the POS of a word in a sentence and then finally finding the 'lemma' (root word) depending on its POS category. This task can be considered as a future work. Following are few words along with their prefix, root and suffix part.

Table I: Stemming Example

Word	Prefix	Root	Suffix
विदेशी	वि	देश	ई
अनुभवले	------	अनुभव	ले
उपनगरपालिकाहरू	उप	नगर	पालिका, हरू
सहायतामा	------	सहायता	मा
मामा	------	मामा	------

2. LITERATURE SURVEY
Lovins Stemmer was the first ever published stemmer, which was written by Julie Beth Lovins in 1968 [4]. This paper was remarkable for its early date and has greate influence on later work in the area of NLP. Then in July 1980 at the University of Cambridge [5], Martin Porter developed the "Porter Stemmer", which is a rule based stemmer with five steps using a set of rules [6]. This stemmer was widely used and became the standard algorithm used for English language. Later, few other stemmers were developed by Paice & Husk [7], Dawson [8] and Krovetz [9]. Most of these stemmers were rule based stemmers which followed the Suffix Stripping approach. Among these, the Porter Stemmer has proved to be an invaluable resource to researchers who work on stemmers also it has been applied to languages other than English.

Since the inception of the concept of stemming in 1968, a lot of work had been done in English and other European languages but a little work had been done in Nepali language. Natural Language Processing in Nepali started in the year 2005 with the release of the first Spell Checker for Nepali and the "Dobhase" English to Nepali machine translation project, respectively developed by Madan Puraskar Pustakalay [10] with the collaboration of Kathmandu University, in Nepal. Corpus building and annotation for Nepali, Text-To-Speech

System for Nepali, digitized Nepali dictionary also got started in the same year under the NeLRaLEC (Nepali Language Resources and Localization for Education and Communication) Project, also known as the Bhasha Sanchar Project [11] in Madan Puraskar Pustakalay. These works motivated researcher to do further research and implement of Natural Language Processing functions in Nepali language. Also in one paper, a hybrid based algorithm was used for stemming on Nepali Text [12]. It shows an accuracy of 68.43%. This paper also compared the results of rule based and hybrid based stemmer using precision and recall metrics.

3. TYPES OF STEMMING

There are several types of stemming algorithms [13] which are mentioned below; each of these groups has a typical way of finding the stems of the word variants.

Brute Force Algorithms:
In Brute force stemmers, a lexicon which contains a relation between root form and inflected form. To stem a word, the lexicon is queried to find a matching inflection. If a matching inflection is found, the associated root form is returned.

Suffix Stripping Algorithms:
Suffix stripping algorithms based on a typically small list of "rules" stored in an array or a lexicon. This list provides a path for the algorithm to find out the root form of a given input word.

Lemmatization Algorithms*:*
This process involves first determining the part of speech category (e.g.: noun, verb, adjective, adverb etc) of the input word, and applying different normalization rules for each part of speech category. The part of speech is first detected and finally the stemming rules change depending on word's part of speech category.

Stochastic Algorithms:
In this method machine first trains the inflected word along with root word and produces some probabilistic internal rule set. Based on these internal rules machine finds out the most probable root word from a given input word and most of the cases it removes the affixes from the word.

Hybrid Approaches:
As the name clearly suggests that these methods are combination of two or more of the approaches described above. A simple example is a suffix tree algorithm which first consults a lookup table using brute force.

Affix Stemmers:
In linguistics concern the term affix refers to either prefix or suffix. In addition to dealing with only suffixes, several approaches are there which discussed in literature survey. If many of the approaches mentioned above can strip prefix as well as suffix then they are called affix stemmer.

Matching Algorithms:
Such algorithms use a stem database (for example a set of documents that contain stem words). These stems are not necessarily valid words themselves (but rather common sub-strings). In order to stem a word the algorithm tries to match it with stems from the database, applying various constraints, such as on the relative length of the candidate stem within the word.

4. SYSTEM DESCRIPTION

For explaining the proposed system three lexicons are taken into consideration: prefix, suffix, root lexicon and a set of hand written rules for prefix and suffix stripping, especially for inflected and derived word. Lexicon means dictionary, where various entries of words of any language are included. Words in the lexicons are manually created.

4.1 Suffix Lexicon

Suffix means those words that come after the root or stem. Approximately 120 suffixes are taken into consideration.

4.2 Prefix Lexicon

Prefix means those words which are added to the front of the word. Around 25 words are taken to build prefix lexicon.

4.3 Root Lexicon

Root lexicon contains over 1000 words.

4.4 Proposed Algorithm for Stemmer

Since domains are taken from Nepali corpus so obviously these texts will have sufficient number of Nepali punctuations, Nepali and English digits and Single-letter-words. But while developing a Stemmer we need not take these unnecessary characters into consideration .So before doing actual stemming it will be efficient to remove these characters step by step and this process is called cleaning. This will be the pre-processing steps of a good Stemmer.

4.4.1 Tokenization

Tokenization is the process of breaking the sentences as well as the text file into word delimited by white space or tab or new line etc. Outcome of this tokenization phase is a set of word delimited by new line.

4.4.2 Punctuation Removal

A document may contain lots of Nepali punctuations in the text. These characters have no importance for stemming.

4.4.3 Digit Removal

In general Nepali text file may contain Nepali as well as English digits. But as meaningful Nepali words do not contain digits.

4.4.4 Single Letter-Word Removal

There exist a lot of words having a single letter. Most of these Single-Letter-Words are Stop-Words (those words which have extremely high term frequency in a corpus are known as Stop Words). As a step of stemming the stop-words need to be removed before further processing. So the Single-Letter-Words are removed in this phase

For above all pre-processing steps list of digits both English and Nepali, list of stop words and list punctuations are maintained in a text file.

4.4.5 Lexical Look up Approach

After processing all the above steps, words (tokens) are ready for stemming. The stemming system was developed using lexical look up based approach using three different lexicons suffix, prefix and root. The system mainly works in two steps: Firstly the input word is queried in the root lexicon; if it is found, then it is considered as a root word e.g. '**मामा**'. Secondly if it fails then the system queried into prefix and suffix lexicon for affixes, if it is present then its rule number is retrieved and do affix striping according to rule which were written in the lexicon against retrieving rule number e.g. in the word '**विदेशी**', one prefix '**वि**'and one suffix 'ी'is present. Rule

for '**वि**' prefix is strip off '**वि**' from the input word'**विदेशी**'and keep it same but for 'ी' suffix, strip off 'ी' from the input word'**विदेशी'** replace 'ी' by ' **ई**'. If the root word is found after stripping suffix/ prefix or both then the system will store root word along with its constituents parts i.e. prefix and suffix into output file. If one or more prefix/suffix present and the root word is not found then try combining the suffix/prefix one by one with the remaining part of the word and again search in the root lexicon for the root. This process will be continuing until it finds the root word or its prefix/suffix lists are empty otherwise the system will keep the word as it is and store the word into output file.

Lexicons are in column format, i.e. a word per line in a sentence by sentence fashion along with serial numbers; a separator '|' is used to separate the word and number. Following are samples of prefix, suffix and the root lexicon:

हरू | 1	न | 1	अर्थशास्त्र|1
लाई | 2	उप | 2	शाखा|2
ले | 3	महा | 3	रथी|3
को | 4	अ | 4	अतिरिक्त|4
का | 5	सु | 5	कदाचित्|5

Suffix Lexicon **Prefix Lexicon** **Root Lexicon**

All the root, suffix and prefix lexicons are stored into hash table with two field's key and value. The key of the hash function was generated from the summation of the ASCII values of each character of every suffix, prefix and root. Using this key, probable position was generated to store each suffix, prefix and root. If there was a previously stored suffix, prefix and root in that position, then collision occurred. In case of collision the next free space was searched and the suffix, prefix and root, were stored in that position. We found that the collision is minimum here [14].

The architecture of the core engine of the stemmer is presented below:

Table II: Test Cases

Test No	Domain	No. of words
1	Economics	400
2	Health	600
3	Political	800

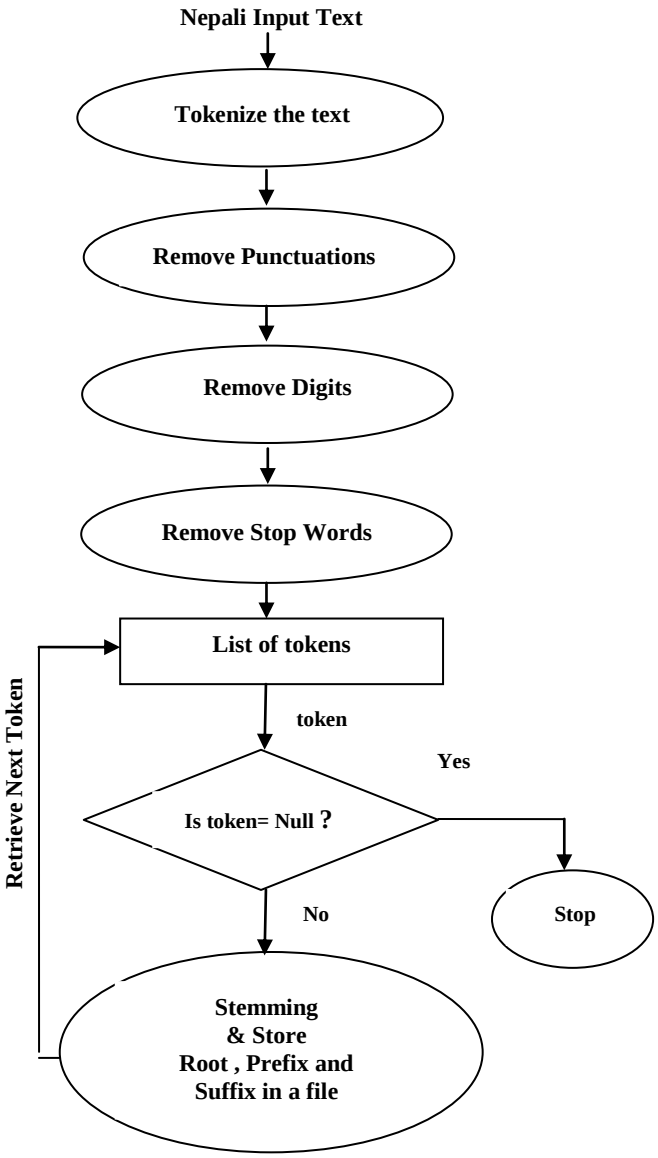

fig: Architecture of Nepali Stemmer

5. EXPERIMENTAL RESULT & DISCUSSION

System performance was evaluated based on different domains of news. These domains include news on Economics, Health & Political. The system was evaluated on 1,800 words. The overall accuracy achieved by the system is 90.48%. Three test data sets were taken from original corpus for testing, which was built by Technology Development for Indian Languages (TDIL) [15]. The following table shows the different test cases for testing:

The evaluation metrics for the data set is precision, recall and F-Measure. These are defined as following:-

Recall (R) = Number of words stemmed by the system / Total number of words.

Precision (P) = Number of words correctly stemmed by the system / Total number of words.

F-Measure = $(\beta^2 + 1)$ PR / $(\beta^2$ R + P)

Where **β** is the weighting between precision and recall and typically **β = 1**.

Table III: Accuracy of System on different Test Cases

Data Set No.	Recall	Precision	F-Measure
Data Set-1	92.35%	89.95%	91.13%
Data Set-2	91.56%	88.80%	89.82%
Data Set-3	91.78%	89.26%	90.50%

5.1 Accuracy measurement of stemmer

During stemming process two text files are generated for each test case: stem.txt and unrecognize.txt. If the system stems the word then it will store its root part into stem.txt otherwise store the word into unrecognize.txt. From these two files we can get total number of stem as well as unknown words. Also in case of getting total number correct root word, one small matcher program is written which sequentially reads stem.txt and root lexicon. The correct root words are those which match in both the files and the remaining words are unknown words. The program will match word by word and increment the count if the word matched in both the files.

The overall accuracy is measured by calculating the mean of three F-measure values. F-measure which combines the precision and recall to give a single score. It is defined to be the harmonic mean of the precision and recall.

6. CONCLUSION AND FUTURE WORK

Though the overall accuracy of the system is over 90% but one can consider following points for future work –

i. It can improve by increasing the size of all three lexicons.

ii. It can be made efficient by considering context based words.

iii. Lemmatizing technique can be used to improve the performance.

iv. It can be compared with many other algorithms.

v. It can implement as a spelling checker.

vi. One can use this for the developing a POS tagger for Nepali Text.

Further analysis and fulfill the above mentioned technique would improve the system performance.

7. ACKNOWLEDGMENTS

Thanks to Mrs. Sunita Sarkar from Assam University Silchar, India for her support and inspiration throughout the duration of the work. Finally thanks go to god and all our well wisher.

8. REFERENCES

[1] T.Siddiqui and U.S. Tiwary, "Natural Language Processing and Information Retrieval", Oxford University Press Publication, 2010.

[2] P.Sinha, B.Sarma and B.Purkayastha, "Kinship Terms in Nepali Language and its Morphology" International Journal of Computer Applications, Vol. 58 pp.9-15, 2012.

[3] B. Prasain, LP. Khatiwada, B.K. Bal, and P. Sheathe, "Part-of-speech Tagset for Nepali", Madan Puraskar Pustakalay, 2008.

[4] J.B.Lovins, "Development of a stemming algorithm", Mechanical Translation and Computational Linguistics 11, 1968, pp. 22-31.

[5] M.F. Porter, "An algorithm for suffix stripping", Program, 14(3) 1980, pp. 130−137.

[6] C.J. van Rijsbergen, S.E. Robertson and M.F. Porter, "New models in probabilistic information retrieval", British Library Research and Development Report, no. 5587,1980.

[7] C.D. Paice, "An evaluation method for stemming algorithms", In the Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, 1990, pp. 42 – 50.

[8] J. Dawson, "Suffix removal and word conflation", LLCbulletin, 2(3), 1974, pp. 33– 46.

[9] R. Krovetz, "Viewing morphology as an inference process", In Proceedings of the 16 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1993, pp. 191-202.

[10] http://www.mpp.org.np

[11] http://www.bhashasanchar.org

[12] C. Sitaula, "A Hybrid Algorithm for Stemming of Nepali Text", Intelligent Information Management, vol. 5, pp. 136-139, 2013.

[13] B. Das & T. Paul, "Development of Bengali Language Stemmer", A Project Report.

[14] Dr. S. Lipschutz, "Data Structures", International Edition 2008.

[15] http://tdil.mit.gov.in