HINDI LEMMATIZER

Pranjal Kumar Srivastava   Mayank Devnani   Pradeep Chalotra
22111046, 22111042, 22111045
{pranjalks22, mayankdev22, pchalotra22}@iitk.ac.in
Indian Institute of Technology Kanpur (IIT Kanpur)

Abstract

Stemming is the process of clipping off the prefixes and suffixes from the input word to obtain the respective root word, but it is not necessary that stemming provide us the genuine and meaningful root word. To extract the meaningful root word from the given composition word, we use the concept of 'Lemmatizing'. It is the process by which we crave out the lemma from the given word. In our project, we have designed the lemmatizer to work on the Hindi language written in Devanagari script.

Keywords: Hindi, Lemma, Stems, Lemmatization, Stemming, Hybrid, Inflected

1 Introduction

In linguistics, the objective of a lemmatizer is to group together the different inflected forms of a word, such that these inflected words are analyzed as a common term. When it comes to major Indic languages, the very first hurdle is the presence of various morphological variants of root words in the text. Existence of morphological richness in these languages creates problems for developing several text processing tasks like Natural Language Processing (NLP), Information Extraction (IE) etc.

Lemmatization stands as a mandatory pre-processing module where semantic processing is needed. The role of a lemmatizer is to map a surface word in a context to its root form. In this project, we have created a lemmatizer for the Hindi language written in Devanagari script. To implement this lemmatizer, we needed a dataset containing the mappings between the lemma and corresponding derived words in a context. Hence, we have created a dataset using 2 methods, from the Universal Dependency Tree Bank dataset for Hindi and using the Fast Text word embeddings.

From the literature survey we carried out, there can be multiple methods to tackle this problem of Lemmatization. The most intuitive approach is to extract the suffix from the derived word to get back to the source lemma. This was our approach, known as rule based approach. To implement this rule based approach, we manually defined the rules after analysing the dataset.

In this report, we start with discussing related work and then propose our method for both, dataset generation and lemmatization. Then, we describe these methods in detail in the next section. Finally, we conclude with results and scope for future work.

2 Related Work

A lot of research is going into Natural Language Processing (NLP). Lemmatization is one of the major challenges in NLP. A rule based approach is one of the most accepted lemmatizing algorithms. There have been work done in various Indian languages for lemmatization.

There is only one dataset available in public domain with mapping between words in Hindi and their corresponding Lemma. That is the Universal Dependency (UD) treebank for Hindi created at IIIT

Hyderabad, India [1, 2]. This dataset contains a set of sentences and the mapping of each individual word in the sentence to its corresponding lemma. We have used this dataset for developing and testing our lemmatizer. Furthermore, because of the shortage of publicly available dataset for lemmatization task, we have created a dataset of 9559 word-lemma mappings with context inspired by the method used in [3]. We plan to make this dataset open-source.

In [3], lemmatization data with context is generated for Bengali language. All the roots having length greater than or equal to 6, are mined from the Bengali digital dictionary available with Chicago university [4]. Next, for every root word, its 10 nearest neighbors in the vector space are considered and among them, only those are selected for which the overlapping prefix length with the root is at least 6. To generate the context, the words are searched in the FIRE Bengali news corpus and when an instance is found, the surrounding context is extracted.

For lemmatization, a method has been proposed in which a clustering based approach is used for discovering the equivalent classes of root words. There has also been work done using composite deep neural network architecture [5]. Significant work in different languages has been done using a rule based approach [6, 7]. The rule based methods are simple to implement and understand. Also, they serve as a good baseline before trying out more complex techniques. Hence, we have implemented a rule based method for carrying out lemmatization here.

3 Proposed Idea

There were 2 major tasks in this project. The first was the development of a rule based lemmatizer and second was the generation of dataset. The lemmatizer has been developed and tested on the UD treebank dataset. For a given word, the lemmatizer searches in its database to figure out if the given word is a lemma. If it is a lemma, lemmatizer outputs the word, otherwise it applies suffix stripping and addition rules to output the lemma for the given word. The database for the lemmatizer has been made from the UD treebank dataset.

Parallelly, we have also created a dataset of word-lemma mappings with context. To do this, we have taken a root word and found the top-5 nearest words to it using the FastText vector embeddings. The valid words among these top-5 are then taken as derived word for the given root word (lemma). The context for these valid words is generated from [8]. For the given word, its neighboring words form its context.

4 Methodology

4.1 Development of Lemmatizer

The lemmatizer that we built was using rule based approach

4.1.1 Approach Used

1) We have used rule based approach for extracting the suffixes.

2) In rule based approach we have created many different rules for striping/appending of suffixes.

4.1.2 Suffix Generation

For suffix generation we went through the training dataset and examined them. We developed list of suffixes to be used from this knowledge. This led to the making of rules. For example, if we take the word " then we find the word is derived by adding suffix' ' . Similarly there are many others words with same suffix.

4.1.3 Rule Generation

1)After the generation of suffix we have developed rules.

2)We developed rule in such a way that we removed the suffix from the word and if required addition of ‘maatra’ or character take place example: - + =

4.1.4 Algorithm steps

1) Check the word in database.

2) If present then display it.

3) Otherwise apply the rules.

4) Strip the suffix and if required then add suffix.

4.2 Dataset Generation

1) Extract the list of lemma given in the UD treebank.

2) Find the FastText vector embeddings for these lemmas.

3) Use these embeddings to find the 5 nearest words to these lemmas.

4) From these 5 nearest words, pick only those as valid words for whom the complete lemma exists in their prefix. This gives the word-lemma mapping.

5) To find the context, we have created trigram of words from [8]. Each valid word is searched in this trigram to find its context.

6) The context for a word is its immediate neighbor words present in trigram.

7) Thus, using this method we have created a dataset of 9,345 words, along with their mappings to lemma and context.

Furthermore, similarly we have parsed the UD treebank dataset to generate word-lemma mappings with immediate neighboring words as context. This way we have generated 17,451 word lemmatization dataset. Thus overall, we have generated total 27,010 word-lemma mappings with context.

5 Results

The rule based Lemmatizer was tested on the UD tree bank data. This gave an accuracy of 58.99 %.

As mentioned earlier in the report, we have generated a total of 27,010 word-lemma mappings with context using UD treebank dataset along with FastText vector embeddings.

6 Discussion and Future Work

1) If a bigger list of Hindi root words is present, more word-lemma mappings with context can be generated to create a bigger dataset using similar technique as mentioned in the report.

2) The rule based lemmatizer developed here can act as a baseline using which more complex Deep Learning based techniques can be explored.

7 Project Readme

7.1 Hindi Lemmatizer

1) Code Files
a)**lemmatizer.ipynb** : Rule based lemmatizer for Hindi language in Devanagari script.

7.2 Dataset Generation

1) Code Files
a)**trigrams-of-hindi.ipynb** : Generate dataset from FastText embeddings.
b)**trigrams-of-hindi-for-ud-treebank** : Generate dataset from UD treebank.

2) Dataset
a) **Context**$_C$*leanedData.csv* : *GenerateddatasetfromFastTextembeddings.*
b)**ContextUD.csv** : *GenerateddatasetfromUDtreebank.*

References

[1] R. A. Bhat, R. Bhatt, A. Farudi, P. Klassen, B. Narasimhan, M. Palmer, O. Rambow, D. M. Sharma, A. Vaidya, S. R. Vishnu, *et al.*, "The hindi/urdu treebank project," in *Handbook of Linguistic Annotation*, Springer Press.

[2] M. Palmer, R. Bhatt, B. Narasimhan, O. Rambow, D. M. Sharma, and F. Xia, "Hindi syntax: Annotating dependency, lexical predicate-argument structure, and phrase structure," in *The 7th International Conference on Natural Language Processing*, pp. 14–17, 2009.

[3] A. Chakrabarty, A. Chaturvedi, and U. Garain, "A neural lemmatizer for bengali," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 2558–2561, 2016.

[4] "Chicago university digital dictionary.," Available at http://dsal.uchicago.edu/dictionaries/list.html.

[5] A. Chakrabarty, O. A. Pandit, and U. Garain, "Context sensitive lemmatization using two successive bidirectional gated recurrent networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1481–1491, 2017.

[6] R. Rodrigues, H. Gonçalo Oliveira, and P. Gomes, "Lemport: a high-accuracy cross-platform lemmatizer for portuguese," in *3rd Symposium on Languages, Applications and Technologies*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

[7] R. Prathibha and M. Padma, "Design of rule based lemmatizer for kannada inflectional words," in *2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*, pp. 264–269, IEEE, 2015.

[8] "IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages,"