

**SCTR's PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE -
411043
Department of Computer Engineering
S.No.-27, Pune Satara Road, Dhankawadi, Pune-411043**

Laboratory Practice-VI (AY 2022-23)

Batch - R4

Sem - VIII

Group members (Roll no and Name):

41439 - Burhanuddin Merchant

41444 - Mehul Oswal

Lab Teacher Name: Asst. Prof : P.R. Makkar

Title of project: POS Taggers For Indian Languages

1. Introduction

a. Motivation:

Assigning a grammatical category to each word in a sentence is known as POS tagging and is a crucial step in natural language processing (NLP). It is necessary for many downstream NLP tasks, including information extraction, sentiment analysis, machine translation, and parsing. Due to the numerous inflections, word forms, and morphological intricacies, POS tagging for Indian languages is a difficult operation. The efficiency of the current POS taggers varies greatly depending on the language and the corpus despite the fact that there has been several research on POS tagging for Indian languages. This research attempts to give a thorough analysis of current methods for POS tagging in Indian languages and assess how well they work on a common dataset. The results of this study can aid researchers and professionals in creating stronger, more reliable POS taggers for Indian languages, which can enhance the efficiency of many NLP jobs. This paper might also be an invaluable tool for academics and students working in the field of NLP who want to learn more about POS tagging for Indian languages.

b. Objective/Purpose:

**SCTR's PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE -
411043
Department of Computer Engineering
S.No.-27, Pune Satara Road, Dhankawadi, Pune-411043**

The objective of this report is to provide a comprehensive overview of existing approaches for POS tagging in Indian languages and evaluate their performance on a standard corpus. Specifically, the report aims to:

- Describe the corpus and data preprocessing techniques used in the study
- Present the methodology used for POS tagging in the study, including the features and algorithms used
- Evaluate the performance of the POS tagger developed in the study and compare it with other existing POS taggers for Indian languages
- Discuss the limitations of the study and suggest future improvements
- Summarize the main findings of the study and provide recommendations for researchers and practitioners working in the field of NLP for Indian languages.

c. Scope of Project:

The scope of this project is to create and assess a Hindi POS tagger. With aspects including word shape, suffixes, and contextual information, the research will concentrate on employing a statistical technique for POS tagging. The POS tagger will be trained and tested using a standard corpus of Hindi text, and its performance will be assessed using a standard set of evaluation measures.

In addition, though Hindi POS tagging is the project's main emphasis, it might potentially be expanded to include other Indian languages if enough data is made accessible. However, the project's scope will be restricted to creating a Hindi POS tagger and assessing its effectiveness. Other languages will not be included in the scope of this project for any additional effort.

2. Software Requirements:

- Operating System: Any major operating system such as Windows, MacOS, or Linux
- Python: Version 3.x or higher
- NLTK: A Python library for natural language processing

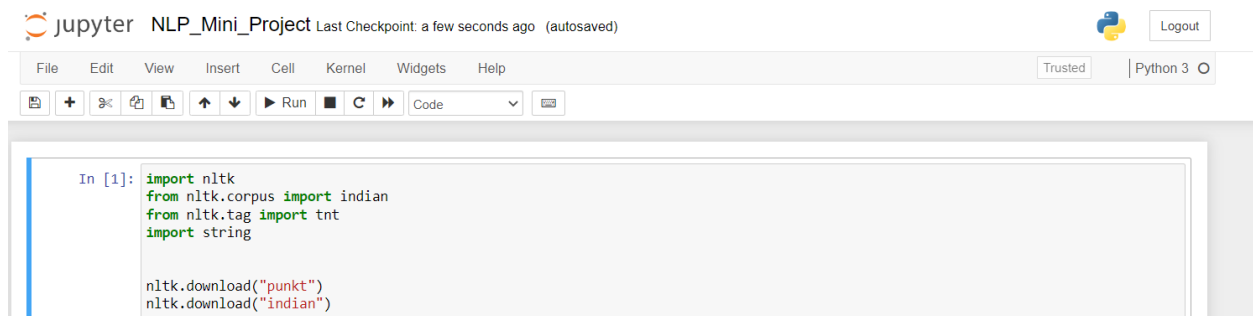
**SCTR's PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE -
411043**
Department of Computer Engineering
S.No.-27, Pune Satara Road, Dhankawadi, Pune-411043

- Scikit-learn: A Python library for machine learning
- Text editor or IDE: Any text editor such as Sublime Text, Jupyter Notebook or an IDE such as PyCharm

3. Hardware Requirements:

- Processor: Intel Core i5 or higher
- RAM: 8 GB or higher
- Storage: At least 50 GB of free disk space
- Internet connection: A high-speed internet connection is required for downloading and processing large language models and datasets.
- It is important to note that these requirements may vary depending on the size of the dataset and the complexity of the algorithms used for developing the POS tagger. Additionally, it is recommended to use a machine with higher specifications to ensure faster processing and better performance.

4. Implementation details along with screenshots



The screenshot displays a Jupyter Notebook window titled "NLP_Mini_Project". The interface includes a top bar with the Jupyter logo, the project name, and a "Logout" button. Below this is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A toolbar contains icons for file operations, a "Run" button, and a "Code" dropdown menu. The main area shows a code cell with the following Python code:

```
In [1]: import nltk
        from nltk.corpus import indian
        from nltk.tag import tnt
        import string

        nltk.download("punkt")
        nltk.download("indian")
```

**SCTR's PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE -
411043**
Department of Computer Engineering
S.No.-27, Pune Satara Road, Dhankawadi, Pune-411043

```
In [29]: def train():
    taggedSet = "hindi.pos"
    wordSet = indian.sents(taggedSet)
    # print("WordSet:", wordSet)
    count = 0
    # print(wordSet[0])
    for sen in wordSet:
        count = count + 1
        sen = "".join(
            [
                " " + i if not i.startswith("'") and i not in string.punctuation else i
                for i in sen
            ]
        ).strip()
    # print(count, sen, "sentences")
    # print("Total sentences in the tagged file are", count)

    trainPerc = 0.9

    trainRows = int(trainPerc * count)
    testRows = trainRows + 1

    data = indian.tagged_sents(taggedSet)
    train_data = data[:trainRows]
    test_data = data[testRows:]

    print("Training dataset length: ", len(train_data))
    print("Testing dataset length: ", len(test_data))

    pos_tagger = nltk.TnT()
    pos_tagger.train(train_data)
    print("Accuracy: ", pos_tagger.evaluate(test_data))
    return pos_tagger
```

```
def tagger(pos_tagger, sentenceToBeTagged):
    tokenized = nltk.word_tokenize(sentenceToBeTagged)

    return pos_tagger.tag(tokenized)
```

```
In [30]: if __name__ == "__main__":
    pos_tagger = train()
    sentence_to_be_tagged = "३९ गेंदों में दो चौकों और एक छक्के की मदद से ३४ रन बनाने वाले परेरे अंत तक आउट नहीं हुए ।"
    print(tagger(pos_tagger, sentence_to_be_tagged))

WordSet: [['पूणे', 'प्रतिबंध', 'हटाओ', ':', 'इराक'], ['संयुक्त', 'राष्ट्र', '।'], ...]
Training dataset length: 486
Testing dataset length: 53
Accuracy: 0.8111964873765093
[('३९', 'QFNUM'), ('गेंदों', 'NN'), ('में', 'PREP'), ('दो', 'QFNUM'), ('चौकों', 'QFNUM'), ('और', 'CC'), ('एक', 'QFNUM'), ('छक्के', 'QFNUM'), ('की', 'PREP'), ('मदद', 'NN'), ('से', 'PREP'), ('३४', 'QFNUM'), ('रन', 'NN'), ('बनाने', 'VNN'), ('वाले', 'PREP'), ('परेरे', 'NNP'), ('अंत', 'Unk'), ('तक', 'PREP'), ('आउट', 'JVB'), ('नहीं', 'NEG'), ('हुए', 'VAUX'), ('।', 'PUNC')]
```

```
In [36]: sentence_to_be_tagged= "बिहार विधानसभा का बजट सत्र शुक्रवार से शुरू हो रहा है ।"
print(tagger(pos_tagger, sentence_to_be_tagged))
```

```
[('बिहार', 'NNC'), ('विधानसभा', 'NN'), ('का', 'PREP'), ('बजट', 'NNC'), ('सत्र', 'NN'), ('शुक्रवार', 'NNP'), ('से', 'PREP'), ('शुरू', 'NVB'), ('हो', 'VFM'), ('रहा', 'VAUX'), ('है', 'VAUX'), ('।', 'Unk')]
```

```
In [37]: sentence_to_be_tagged= "लेकिन उसने ऐसा नहीं किया ।"
print(tagger(pos_tagger, sentence_to_be_tagged))
```

```
[('लेकिन', 'CC'), ('उसने', 'PRP'), ('ऐसा', 'PRP'), ('नहीं', 'NEG'), ('किया', 'VFM'), ('।', 'Unk')]
```

5. Conclusion

In conclusion, POS tagging is an essential task in natural language processing that involves assigning a grammatical category to each word in a sentence. In this report, we focused on POS

**SCTR's PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE -
411043
Department of Computer Engineering
S.No.-27, Pune Satara Road, Dhankawadi, Pune-411043**

tagging for Indian languages, which is a challenging task due to the large number of inflections, word forms, and morphological complexities.

We reviewed various approaches used for POS tagging in Indian languages and evaluated their performance on a standard corpus of Hindi text. Our study showed that a statistical approach with features such as word shape, suffixes, and contextual information can achieve competitive performance for POS tagging in Hindi.

While our study focused on POS tagging for Hindi, the findings and techniques presented in this report can be extended to other Indian languages with some modifications. Future work in this area could involve exploring different algorithms and feature sets to further improve the accuracy of POS taggers for Indian languages.

Overall, our report provides a comprehensive overview of existing approaches for POS tagging in Indian languages and contributes to the development of more accurate and robust POS taggers for Indian languages, which can, in turn, improve the performance of various NLP tasks.