Machine Learning approach to POS Tagging in Bengali Language

Heritage Institute of Technology

Presented by:

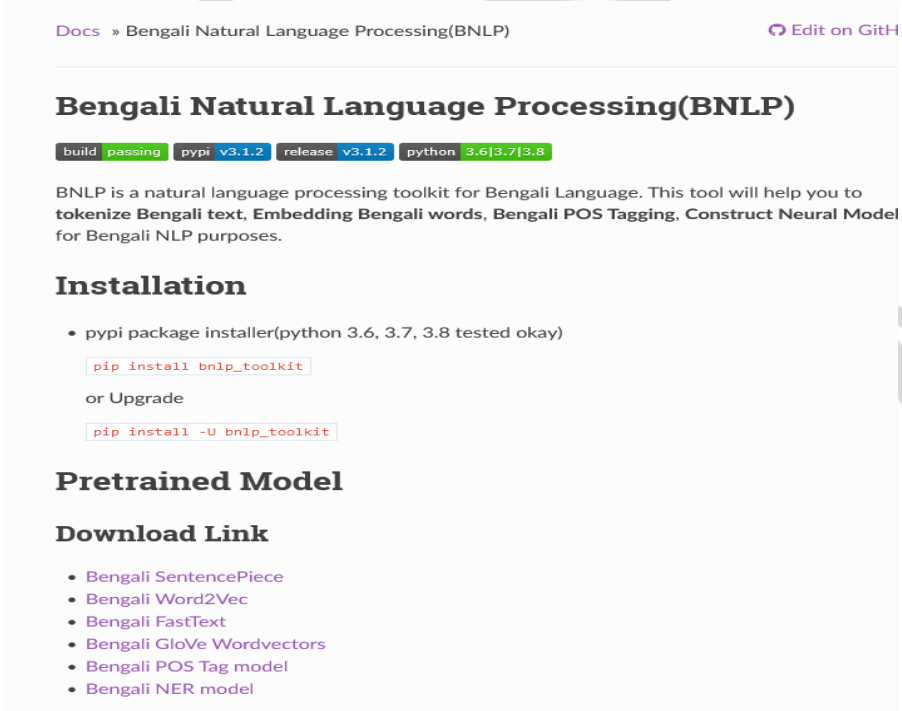
- Rajdeep Das (12619010037)
- Arghyadeep Banerjee (12619010011)
- Soham Chakraborty (12619010051)
- Tanmay Guchhait (12619010056)
- Debabrata Maity (12619010016)
- Alik Sarkar (12619010004)
- Sanju Manna (12619010043)

Mentor:

Prof. Sandipan Ganguly

Data & Model Source

https://bnlp.readthedocs.io/en/latest/



Docs » Bengali Natural Language Processing(BNLP) Edit on GitH

Bengali Natural Language Processing(BNLP)

build passing | pypi v3.1.2 | release v3.1.2 | python 3.6|3.7|3.8

BNLP is a natural language processing toolkit for Bengali Language. This tool will help you to **tokenize Bengali text, Embedding Bengali words, Bengali POS Tagging, Construct Neural Model** for Bengali NLP purposes.

Installation

- pypi package installer(python 3.6, 3.7, 3.8 tested okay)

```
pip install bnlp_toolkit
```

or Upgrade

```
pip install -U bnlp_toolkit
```

Pretrained Model

Download Link

- Bengali SentencePiece
- Bengali Word2Vec
- Bengali FastText
- Bengali GloVe Wordvectors
- Bengali POS Tag model
- Bengali NER model

Introduction

- Through this project we've explored the idea of Parts of Speech Tagging in Natural Language Processing.
- Tokenizing & classifying Parts of Speech tags on different Bengali Corpus.
- Utilizing Conditional Random Field based Approach using BNLP Model on Bengali raw text.
- Understanding Rule based & Stochastic based approach.
- Exploring Confusion Matrices & its uses.

Natural Language Processing & its role in POS Tagging

Natural language processing is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language.

Now Natural language processing (NLP) tools have sparked a great deal of interest due to rapid improvements in information and communications technologies.

As a result, many different NLP tools are being produced. One such tool is part of speech (POS) tagging.

What is POS Tagging

Part-of-speech (POS) tagging is a popular Natural Language Processing technique which refers to categorizing words in a text (corpus) in correspondence with a particular part of speech, depending on the definition of the word and its context.

It consists of assigning to each word of a text the proper morphosyntactic tag in its context of appearance.

It is very useful for a number of NLP applications: as a pre-processing step to syntactic parsing, in information extraction and retrieval (e.g. document classification in internet searchers), text to speech systems, corpus linguistics, etc.

An example of Preprocessing POS Tagging using NLTK

In [6]:
```
import nltk
from nltk.tokenize import word_tokenize
text = word_tokenize("This is a sample POS Tagging Testing on English Text.")
nltk.pos_tag(text)
```

Out[6]:
```
[('This', 'DT'),
 ('is', 'VBZ'),
 ('a', 'DT'),
 ('sample', 'JJ'),
 ('POS', 'NNP'),
 ('Tagging', 'NNP'),
 ('Testing', 'NNP'),
 ('on', 'IN'),
 ('English', 'NNP'),
 ('Text', 'NNP'),
 ('.', '.')]
```

Applications of POS Tagging

- Text-to-Speech Conversion
- Named Entity Recognition
- Co-reference Resolution
- Speech Recognition
- Sentiment Analysis
- Word Sense Disambiguation

Why Bengali POS Tagging?

- **Developing an efficient POS tagger is a challenging task for resource-scarce languages like Bengali.**

- **In Bengali we can also tokenize & classify words into their corresponding Parts of Speech tags. Such classifiers help computers in understanding Bengali language, used in different regional real life applications.**

Scopes in Bengali POS Tagging

POS tagging in Bengali has several real life applications as follows:

- **Bengali Handwriting recognition**
- **Bengali Literature, Poem or Song Sentiment Analysis.**
- **Creating Chatbot in Bengali language.**
- **For medical purpose.**
- **Bengali Manuscript's sentence/word analysis using Continuous Bag of Model or Skip-gram model.**

Challenges faced in Bengali POS Tagging

- **The main challenge of POS tagging comes due to the ambiguity in language. In Bengali, a large percentage of the words in a corpus are ambiguous.**

- **Bengali is a morphologically rich language. It is a highly agglutinative language, because of which the vocabulary size grows of a high rate with increase in the size of the corpus.**

Per Word Tags	No. of Words
1 tag	41,719
2 tags	3,149
3 tags	630
4 tags	504
5 tags	256
6 tags	33

Challenges in doing Bengali POS Tagging

Bengali is a relatively free word order language in comparison with European Languages.

As an example, we can consider the English Sentence:

I eat rice (PRP VB NN)
We have the following possible Bengali equivalents of the sentence:

- **Ami vat khai (I rice eat) (PPR NC VM)**
- **Ami khai vat (I eat rice) (PPR VM NC)**
- **Vat ami khai (rice I eat) (NC PPR VM)**
- **Vat khai ami (rice eat I) (NC VM PPR)**
- **Khai ami vat (Eat I rice) (VM PPR NC)**
- **Khai vat ami (Eat rice I) (VM NC PPR)**

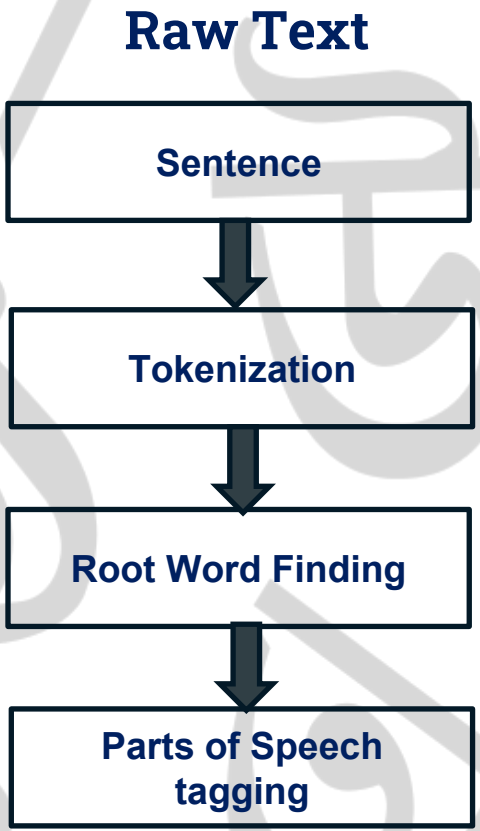
- **Hence, Part of Speech tagging using Linguistic rules is a difficult problem for such kinds of languages.**

Parts of Speech Tagging Bengali Texts

In [4]:
```
from bnlp import POS
bn_pos = POS()
model_path = "bn_pos.pkl"
text = "আমি ভাত খাই।"
res = bn_pos.tag(model_path, text)
print(res)
```

```
[('আমি', 'PPR'), ('ভাত', 'NC'), ('খাই', 'VM'), ('।', 'PU')]
```

Pre-processing steps



Pre-processing Testing sample

Raw text = “ আমি বাংলায় গান গাই।"

Tokenization

```
In [2]: from bnlp import BasicTokenizer
        basic_t = BasicTokenizer()
        raw_text = "আমি বাংলায় গান গাই।"
        tokens = basic_t.tokenize(raw_text)
        print(tokens)
```

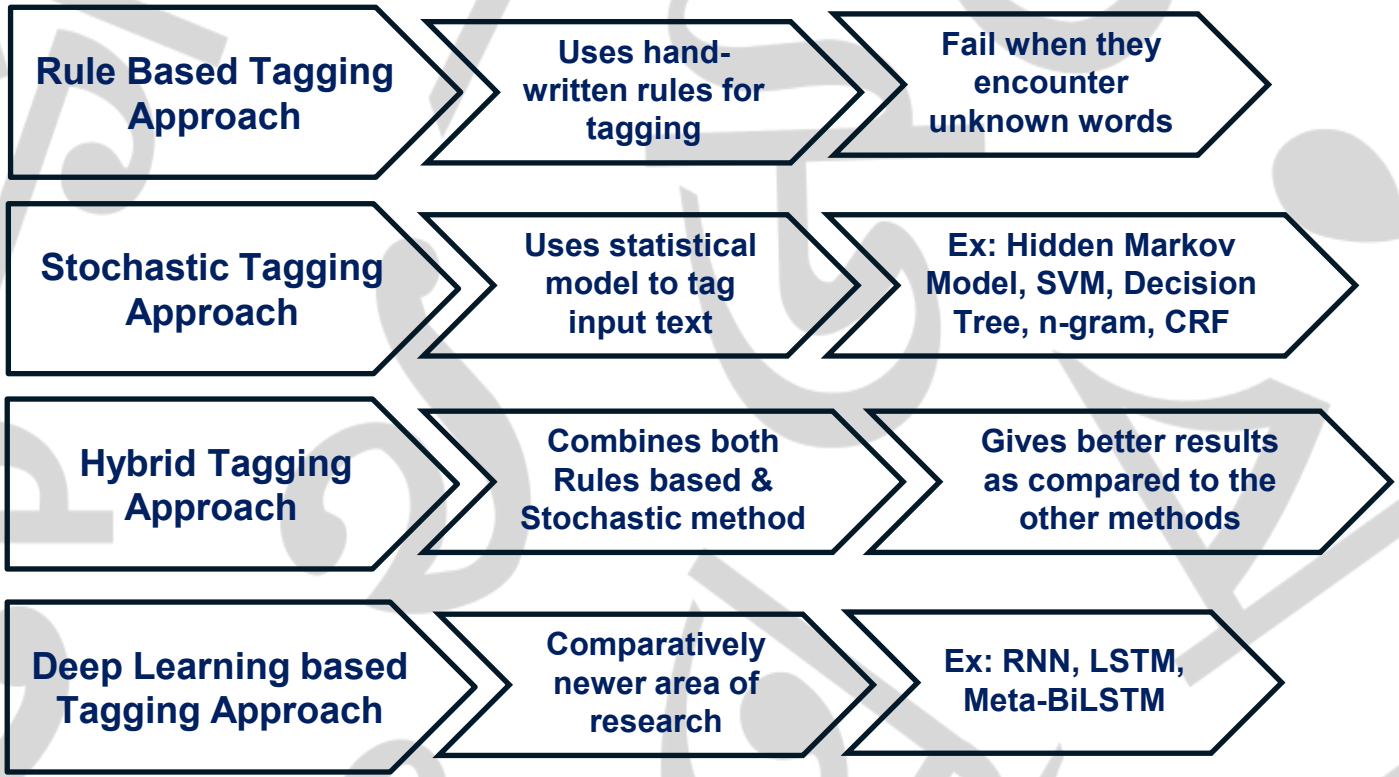
```
['আমি', 'বাংলায়', 'গান', 'গাই', '।']
```

Parts of Speech tagging

```
In [5]: from bnlp import POS
        bn_pos = POS()
        model_path = "bn_pos.pkl"
        text = "আমি বাংলায় গান গাই।"
        res = bn_pos.tag(model_path, text)
        print(res)
```

```
[('আমি', 'PPR'), ('বাংলায়', 'NP'), ('গান', 'NC'), ('গাই', 'NC'), ('।', 'PU')]
```

Methodologies



Rule based approach in tokenization

Rule-based taggers use dictionary or lexicon for getting possible tags for tagging each word. If a word has more than one possible tag, then rule-based taggers use hand-written rules to identify the correct tag.

In this project, we have used 3 types of Tokenization under Rule based approached. These are Basic Tokenizer, Punctuation splitting Tokenizer & NLTK Tokenizer.

Basic Tokenzier used in tokenizing Bengali words from a sentence.
NLTK Tokenizer used in tokenizing Bengali sentences from group of sentences.
Punctuation Splitting Tokenization helps in splitting punctuation from sentences.

Rule based approach on English Text POS Tagging

In [4]:
```
import nltk
from nltk.tokenize import word_tokenize
text = word_tokenize(" Rabindranath Tagore, Bengali Rabīndranāth Ṭhākur, (born May 7, 1861, Calcutta [now Kolkata], India—died
nltk.pos_tag(text)
```

Out[4]:
```
[('Rabindranath', 'NNP'),
 ('Tagore', 'NNP'),
 (',', ','),
 ('Bengali', 'NNP'),
 ('Rabīndranāth', 'NNP'),
 ('Ṭhākur', 'NNP'),
 (',', ','),
 ('(', '('),
 ('born', 'VBN'),
 ('May', 'NNP'),
 ('7', 'CD'),
 (',', ','),
 ('1861', 'CD'),
 (',', ','),
 ('Calcutta', 'NNP'),
 ('[', 'NNP'),
 ('now', 'RB'),
 ('Kolkata', 'NNP'),
 (']', 'NNP'),
```

Rule based approach in Tokenization Bengali Texts

Basic Tokenization on Bengali Words

In [2]:
```
from bnlp import BasicTokenizer
basic_t = BasicTokenizer()
raw_text = "আমি বাংলায় গান গাই।"
tokens = basic_t.tokenize(raw_text)
print(tokens)
```

```
['আমি', 'বাংলায়', 'গান', 'গাই', '।']
```

NLTK Tokenization on i) Bengali Word ii) Bengali Sentence

In [3]:
```
from bnlp import NLTKTokenizer

bnltk = NLTKTokenizer()

text = "আমি ভাত খাই। সে বাজারে যায়। তিনি কি সত্যিই ভালো মানুষ?"

word_tokens = bnltk.word_tokenize(text)
sentence_tokens = bnltk.sentence_tokenize(text)
print(word_tokens)
print(sentence_tokens)
```

```
['আমি', 'ভাত', 'খাই', '।', 'সে', 'বাজারে', 'যায়', '।', 'তিনি', 'কি', 'সত্যিই', 'ভালো', 'মানুষ', '?']
['আমি ভাত খাই।', 'সে বাজারে যায়।', 'তিনি কি সত্যিই ভালো মানুষ?']
```

Conditional Random Field based approach

A CRF is a sequence modeling algorithm which is used to identify entities or patterns in text, such as POS tags. This model not only assumes that features are dependent on each other, but also considers future observations while learning a pattern. In terms of performance, it is considered to be the best method for entity recognition.

- **CRF is a discriminative probabilistic classifier**
- **In CRF based models, the input is:**
  - **A set of feature derived from the input sequence.**
  - **Weights associated with the features.**
  - **previous label**
- **Our task is to predict the correct label**
- **The feature functions express certain characteristics of the sequence. Example: the tag sequence noun-> verb -> adjective**

Bengali POS Tagging in CRF Approach

In [2]:
```
from bnlp import POS
bn_pos = POS()
model_path = "bn_pos.pkl"
text = "বাংলা সাহিত্যের নবজাগরণের পথিকৃৎ-কর্মী বঙ্কিমচন্দ্র চট্টোপাধ্যায়। বঙ্কিমচন্দ্রের জন্ম ১৮৩৮ সালের ২৬ শে জুন, অধুনা চব্বিশ পরগণা জেলার অন্তর্গ
res = bn_pos.tag(model_path, text)
print(res)
```

```
[('বাংলা', 'NP'), ('সাহিত্যের', 'NC'), ('নবজাগরণের', 'NC'), ('পথিকৃৎ', 'NC'), ('-', 'PU'), ('কর্মী', 'NP'), ('বঙ্কিমচন্দ্র', 'NP'),
('চট্টোপাধ্যায়', 'NP'), ('।', 'NP'), ('বঙ্কিমচন্দ্রের', 'NP'), ('জন্ম', 'NC'), ('১৮৩৮', 'RDF'), ('সালের', 'NC'), ('২৬', 'RDF'), ('
শে', 'CX'), ('জুন', 'NP'), (',', 'PU'), ('অধুনা', 'NC'), ('চব্বিশ', 'JQ'), ('পরগণা', 'NC'), ('জেলার', 'NC'), ('অন্তর্গত', 'JJ
'), ('নৈহাটির', 'NC'), ('কাঁঠালপাড়া', 'NC'), ('গ্রামে', 'NC'), ('।', 'NC'), ('বাবা', 'NC'), ('যাদবচন্দ্র', 'NP'), ('চট্টোপাধ্যায়', 'NP
'), ('ছিলেন', 'VM'), ('মেদিনীপুরের', 'NP'), ('কলেক্টর', 'NP'), ('।', 'NP'), ('নিয়মমাফিক', 'JJ'), ('পড়াশোনা', 'NC'), ('শুরু', '
NC'), ('বাবার', 'NC'), ('কর্মস্থল', 'VM'), ('মেদিনীপুর', 'NP'), ('জেলার', 'NC'), ('এক', 'JQ'), ('ইংরেজি', 'NC'), ('স্কুলে', 'NC
'), ('।', 'NC'), ('পরে', 'NST'), ('কাঁঠালপাড়ায়', 'NC'), ('ফিরে', 'VM'), ('হুগলি', 'NP'), ('কলেজে', 'NP'), ('।', 'NP'), ('১৮৫
৬', 'RDF'), ('সালে', 'NC'), ('বঙ্কিমচন্দ্র', 'NP'), ('আইন', 'NC'), ('পড়বার', 'NV'), ('জন্য', 'PP'), ('প্রেসিডেন্সি', 'NC'), ('কলে
জে', 'NC'), ('ভর্তি', 'NC'), ('হন', 'VM'), ('এবং', 'CCD'), ('১৮৫৭তে', 'NC'), ('সেখান', 'NC'), ('থেকে', 'PP'), ('প্রথম', 'JQ
'), ('বিভাগে', 'NC'), ('এন্ট্রান্স', 'NP'), ('পরীক্ষা', 'NC'), ('পাশ', 'NC'), ('করেন', 'VM'), ('।', 'NC'), ('১৮৫৮', 'RDF'), ('সা
লে', 'NC'), ('সদ্য', 'NC'), ('প্রতিষ্ঠিত', 'JJ'), ('কোলকাতা', 'NC'), ('বিশ্ববিদ্যালয়ের', 'NC'), ('প্রথম', 'JQ'), ('বি', 'NC'), ('.',
'NC'), ('এ', 'DAB'), ('.', 'NC'), ('পরীক্ষায়', 'NC'), ('বঙ্কিমচন্দ্র', 'NP'), ('দ্বিতীয়', 'JQ'), ('বিভাগে', 'NC'), ('প্রথম', 'JQ'),
('স্থান', 'NC'), ('অধিকার', 'NC'), ('করেন', 'VM'), ('।', 'NC'), ('আইন', 'NC'), ('পড়া', 'NV'), ('শেষ', 'NST'), ('হওয়ার', 'N
V'), ('আগেই', 'NST'), ('যশোরের', 'NP'), ('ডেপুটি', 'NC'), ('ম্যাজিস্ট্রেট', 'NC'), ('ও', 'CCD'), ('ডেপুটি', 'NC'), ('কলেক্টরের',
'NC'), ('চাকরি', 'NC'), ('পান', 'VM'), ('।', 'PU')]
```

Confusion Matrix

In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one.

By definition a confusion matrix C is such that C(i, j) is equal to the number of observations known to be in group i, and predicted to be in group j.
Thus in binary classification, the count of

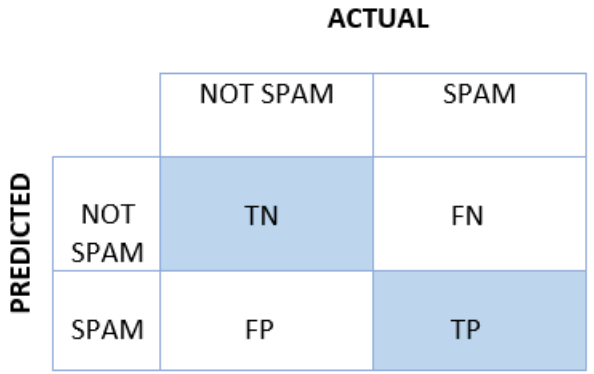
- **True negatives is C(0,0) - TN**
- **False negatives is C(1,0) - FN**
- **True positives is C(1,1) - TP**
- **False positives is C(0,1) - FP**

Confusion Matrix

In this project, while using pre-trained model of Bengali POS Tagger, we have retrieved our values from Precision and Recall from following calculations:

Precision= 81.74: TP/(TP+ FP)

Recall=79.78: TP/(TP+FN)



Confusion Matrix for Spam detection

```
In [5]: from bnlp import POS
        bn_pos = POS()
        model_path = "bn_pos.pkl"
        text = "আমি বাংলায় গান গাই।"
        res = bn_pos.tag(model_path, text)
        print(res)
```

```
[('আমি', 'PPR'), ('বাংলায়', 'NP'), ('গান', 'NC'), ('গাই', 'NC'), ('।', 'PU')]
```

In the above image, the term “গাই” is expected to be tagged as VM (Verb) but is wrongly tagged as NC (Noun). That means, it can be considered as FALSE POSITIVE.

```
In [2]: from bnlp import POS
        bn_pos = POS()
        model_path = "bn_pos.pkl"
        text = "আমি ভাত খাই।"
        res = bn_pos.tag(model_path, text)
        print(res)
```

```
[('আমি', 'PPR'), ('ভাত', 'NC'), ('খাই', 'VM'), ('।', 'PU')]
```

Unlike the previous one, here the term “খাই” is correctly tagged as VM (Verb). So it is TRUE POSITIVE.

Bengali Natural Language Processing Toolkit

• BNLP is an open source language processing toolkit for Bengali language consisting with tokenization, word embedding, POS tagging, NER tagging facilities.
• BNLP provides pre-trained model with high accuracy to do model based tokenization, embedding, POS tagging, NER tagging task for Bengali language.
• BNLP pre-trained model achieves significant results in Bengali text tokenization, word embedding, POS tagging and NER tagging task.
• BNLP is using widely in the Bengali research communities.

BNLP Toolkit has been developed by Mr. Sagor Sarker (AI Engineer, Researcher)

Link: https://github.com/sagorbrur/bnlp/blob/master/docs/index.rst

Installation:
pypi package installer(python 3.6, 3.7, 3.8 tested okay)

>pip install bnlp toolkit
or Upgrade
>pip install -U bnlp toolkit

Utilizing BNLP Toolkit

BNLP has used here Conditional Random Field (CRF) approach to tag Bengali words & punctuations using training dataset of NLTR with 80% accuracy & has Bengali CRF NER Tagging which was trained with this data with 90% accuracy. Our POS Tagging is CRF approach based. We divided data into 75% train and 25% test. Our evaluation result for POS tagging model is 80.75 F1

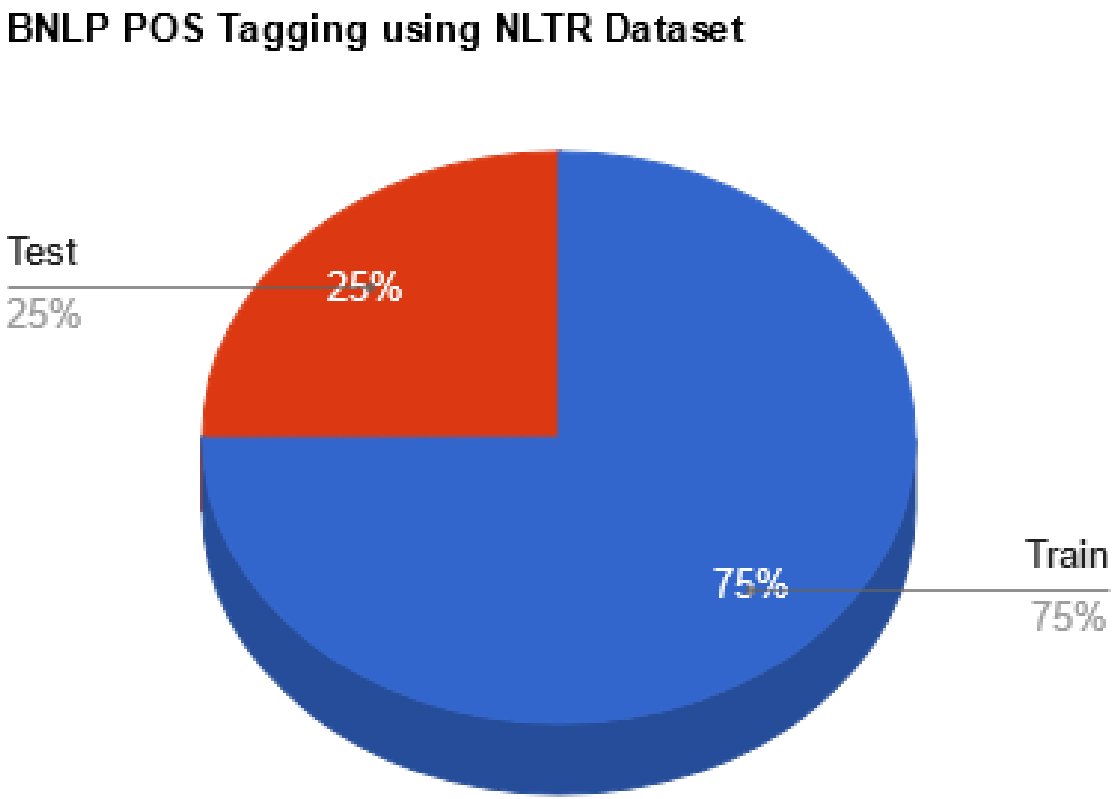
	Sentences	Train	Test
POS	2997	2247	750

Statistics of POS Dataset from NLTR

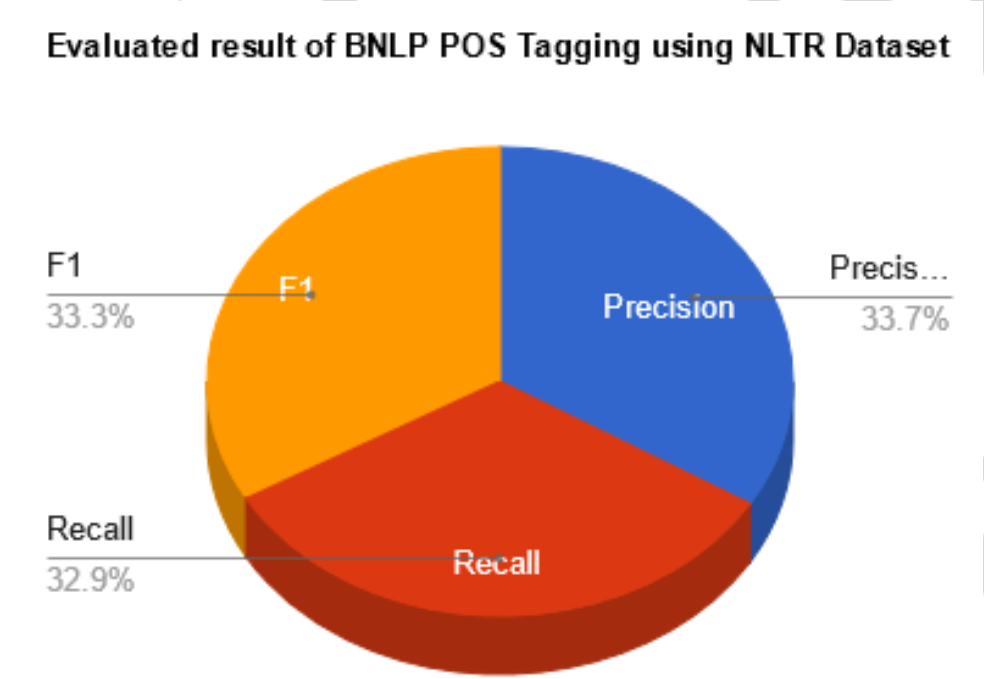
	Precision	Recall	F1
POS	81.74	79.78	80.75

Evaluation result of POS

Cardinality of Corpus we used



Cardinality of Corpus we used



Accuracy

Tagger					Score						Accuracy Score
Unigram Tagger	0.647873865	0.645268664	0.63467189	0.661375661	0.643145161	0.666666667	0.654210028	0.625187406	0.657087189	0.658976931	0.649446346
Bigram Tagger	0.637048193	0.652387986	0.67203219	0.663613232	0.662399241	0.653611394	0.654751131	0.660615229	0.647086915	0.654237288	0.65577828
Tigram Tagger	0.660217084	0.653692615	0.65635256	0.646687697	0.643867925	0.645937358	0.64370664	0.678863746	0.652109912	0.648975791	0.653041132
Unigram Bigram Trigram Tagger	0.658385093	0.648886827	0.67096135	0.660048426	0.661089866	0.660611855	0.648594378	0.639512195	0.639512195	0.639512195	0.654671868
Affix Based Tagger	0.334864727	0.355238095	0.33253589	0.346898263	0.335504886	0.341513455	0.354997538	0.346314325	0.341048654	0.337493759	0.342639613
Affix Unigram Bigram Tigram Tagger	0.773422562	0.778386454	0.76669716	0.756005652	0.783231084	0.766909976	0.778246601	0.757142857	0.773729626	0.776486989	0.771025897
Brill Based Tagger with AUBT as the Trainer Tagger	0.77410593	0.768367347	0.77040573	0.764204545	0.789970209	0.782851344	0.76412649	0.770356816	0.758321274		0.770185157

Future scope of BNLP

- **BNLP language processing toolkit provides tokenization, embedding, pos tagging, NER tagging, language modeling facilities for Bengali language.**
- **BNLP pre-trained model achieves significant results in Bengali text tokenizing, word**
- **embedding, POS tagging and NER tagging task.**
- **BNLP is using widely in Bengali language research communities and appreciated by the communities.**

Thank you