Search projects

Help Sponsors Log in Register

bnlp-toolkit 3.1.2

Latest version

```
pip install bnlp-toolkit
```

Released: Sep 11, 2021

BNLP is a natural language processing toolkit for Bengali Language

Navigation

Project description

Release history

Download files

Project links

Homepage

Statistics

GitHub statistics:

Stars: 159

Forks: 36

Project description

bnlp Bengali Natural Language Processing(BNLP)

build passing | pypi v3.1.2 | release v3.1.2 | python 3.6|3.7|3.8
docs passing | chat on gitter

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**, **Bengali Name Entity Recognition**, **Construct Neural Model** for Bengali NLP purposes.

Installation

PIP installer(Python: 3.6, 3.7, 3.8 tested okay, OS: linux, windows tested okay)

```
pip install bnlp_toolkit
```

Open issues/PRs: 0

View statistics for this project via Libraries.io, or by using our public dataset on Google BigQuery

Meta

License: MIT License (MIT)

Author: Sagor Sarker

Requires: Python >=3.6

Maintainers

sagor_sarker

Classifiers

License
- OSI Approved :: MIT License

Operating System
- OS Independent

Programming Language
- Python :: 3

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

Training Details

- Sentencepiece, Word2Vec, Fasttext, GloVe model trained with **Bengali Wikipedia Dump Dataset**
  - Bengali Wiki Dump
- SentencePiece Training Vocab Size=50000
- Fasttext trained with total words = 20M, vocab size = 1171011, epoch=50, embedding dimension = 300 and the training loss = 0.318668,
- Word2Vec word embedding dimension = 100, min_count=5, window=5, epochs=10
- To Know Bengali GloVe Wordvector and training process follow this repository
- Bengali CRF POS Tagging was training with nltr dataset with 80% accuracy.
- Bengali CRF NER Tagging was train with this data with 90% accuracy.

Tokenization

- **Basic Tokenizer**

```
from bnlp import BasicTokenizer
basic_tokenizer = BasicTokenizer()
raw_text = "আমি বাংলায় গান গাই।"
tokens = basic_tokenizer.tokenize(raw_text)
print(tokens)

# output: ["আমি", "বাংলায়", "গান", "গাই", "।"]
```

- **NLTK Tokenization**

```
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)

# output
# word_token: ["আমি", "ভাত", "খাই", "।", "সে", "বাজারে
# sentence_token: ["আমি ভাত খাই।", "সে বাজারে যায়।", "তি
```

- **Bengali SentencePiece Tokenization**
  - tokenization using trained model

```
from bnlp import SentencepieceTokenizer

bsp = SentencepieceTokenizer()
model_path = "./model/bn_spm.model"
input_text = "আমি ভাত খাই। সে বাজারে যায়।"
tokens = bsp.tokenize(model_path, input_text)
print(tokens)
text2id = bsp.text2id(model_path, input_text)
print(text2id)
id2text = bsp.id2text(model_path, text2id)
print(id2text)
```

  - Training SentencePiece

```
from bnlp import SentencepieceTokenizer

bsp = SentencepieceTokenizer()
```

```
data = "raw_text.txt"
model_prefix = "test"
vocab_size = 5
bsp.train(data, model_prefix, vocab_size)
```

Word Embedding

- **Bengali Word2Vec**
  - Generate Vector using pretrain model

```
from bnlp import BengaliWord2Vec

bwv = BengaliWord2Vec()
model_path = "bengali_word2vec.model"
word = 'গ্রাম'
vector = bwv.generate_word_vector(model_path, w
print(vector.shape)
print(vector)
```

  - Find Most Similar Word Using Pretrained Model

```
from bnlp import BengaliWord2Vec

bwv = BengaliWord2Vec()
model_path = "bengali_word2vec.model"
word = 'গ্রাম'
similar = bwv.most_similar(model_path, word, to
print(similar)
```

  - Train Bengali Word2Vec with your own data

    Train Bengali word2vec with your custom raw data or tokenized sentences.

    custom tokenized sentence format example:

```
sentences = [['আমি', 'ভাত', 'খাই', '।'], ['সে', '
```

Check gensim word2vec api for details of training parameter

```
from bnlp import BengaliWord2Vec
bwv = BengaliWord2Vec()
data_file = "raw_text.txt" # or you can pass cu
model_name = "test_model.model"
vector_name = "test_vector.vector"
bwv.train(data_file, model_name, vector_name, e
```

- Pre-train or resume word2vec training with same or new corpus or tokenized sentences

  Check gensim word2vec api for details of training parameter

```
from bnlp import BengaliWord2Vec
bwv = BengaliWord2Vec()

trained_model_path = "mytrained_model.model"
data_file = "raw_text.txt"
model_name = "test_model.model"
vector_name = "test_vector.vector"
bwv.pretrain(trained_model_path, data_file, mod
```

- **Bengali FastText**

  To use `fasttext` you need to install fasttext manually by `pip install fasttext==0.9.2`

  NB: `fasttext` may not be worked in `windows`, it will only work in `linux`

  - Generate Vector Using Pretrained Model

```
from bnlp.embedding.fasttext import BengaliFast

bft = BengaliFasttext()
word = "গ্রাম"
model_path = "bengali_fasttext_wiki.bin"
word_vector = bft.generate_word_vector(model_pa
print(word_vector.shape)
print(word_vector)
```

- Train Bengali FastText Model

  Check fasttext documentation for details of training parameter

```
from bnlp.embedding.fasttext import BengaliFast

bft = BengaliFasttext()
data = "raw_text.txt"
model_name = "saved_model.bin"
epoch = 50
bft.train(data, model_name, epoch)
```

- Generate Vector File from Fasttext Binary Model

```
from bnlp.embedding.fasttext import BengaliFast

bft = BengaliFasttext()

model_path = "mymodel.bin"
out_vector_name = "myvector.txt"
bft.bin2vec(model_path, out_vector_name)
```

- **Bengali GloVe Word Vectors**

  We trained glove model with bengali data(wiki+news articles) and published bengali glove word vectors
  You can download and use it on your different machine learning purposes.

```
from bnlp import BengaliGlove
glove_path = "bn_glove.39M.100d.txt"
word = "গ্রাম"
bng = BengaliGlove()
res = bng.closest_word(glove_path, word)
print(res)
vec = bng.word2vec(glove_path, word)
print(vec)
```

Bengali POS Tagging

- **Bengali CRF POS Tagging**
  - Find Pos Tag Using Pretrained Model

```
from bnlp import POS
bn_pos = POS()
model_path = "model/bn_pos.pkl"
text = "আমি ভাত খাই।" # or you can pass ['আমি',
res = bn_pos.tag(model_path, text)
print(res)
# [('আমি', 'PPR'), ('ভাত', 'NC'), ('খাই', 'VM'),
```

  - Train POS Tag Model

```
from bnlp import POS
bn_pos = POS()
model_name = "pos_model.pkl"
train_data = [[('রপ্তানি', 'JJ'), ('দ্রব্য', 'NC'), ('

test_data = [[('রপ্তানি', 'JJ'), ('দ্রব্য', 'NC'), ('-

bn_pos.train(model_name, train_data, test_data)
```

Bengali NER

- **Bengali CRF NER**
  - Find NER Tag Using Pretrained Model

```
from bnlp import NER
bn_ner = NER()
model_path = "model/bn_ner.pkl"
text = "সে ঢাকায় থাকে।" # or you can pass ['সে', 'ঢ
result = bn_ner.tag(model_path, text)
print(result)
# [('সে', 'O'), ('ঢাকায়', 'S-LOC'), ('থাকে', 'O')]
```

  - Train NER Tag Model

```
from bnlp import NER
bn_ner = NER()
model_name = "ner_model.pkl"
train_data = [[('ত্রাণ', 'O'),('ও', 'O'),('সমাজকল্য

test_data = [[('ত্রাণ', 'O'),('ও', 'O'),('সমাজকল্যাণ

bn_ner.train(model_name, train_data, test_data)
```

Bengali Corpus Class

- Stopwords and Punctuations

```
from bnlp.corpus import stopwords, punctuations, le

print(stopwords)
print(punctuations)
print(letters)
print(digits)
```

- Remove stopwords from Text

```
from bnlp.corpus import stopwords
from bnlp.corpus.util import remove_stopwords

raw_text = 'আমি ভাত খাই।'
result = remove_stopwords(raw_text, stopwords)
print(result)
# ['ভাত', 'খাই', '।']
```

Contributor Guide

Check CONTRIBUTING.md page for details.

Thanks To

- Semantics Lab

Extra Contributor

- Mehadi Hasan Menon
- Kazal Chandra Barman