NLTK Server Documentation

Release 1

Preetham MS

January 31, 2017

Contents

NLTK Server enables you to access the features provided by NLTK Library over a REST interface. It can be easily installed on your Application VM or a seperate VM. NLTK Server makes it easy to use NLTK with all other Languages which can make REST calls and parse JSON.

CHAPTER 1

Documentation

1.1 Installation

1.1.1 Prerequisites

To install NLTK Server you will need the following:

- Python 2.7.x
- Pip

Read more on installing PIP here. https://pip.pypa.io/en/latest/installing.html

1.1.2 How to Install

1. Clone the NLTK server repository.

```
git clone https://github.com/preems/nltk-server
```

Or download latest Zip from Github:

2. Install dependencies from pip.

```
cd nltk-server
pip install -r requirements.txt
```

Some linux distros might require to use sudo.

3. Install JRE (only required for Stanford NER).

Installation of JRE is specific to Operating system your running on. For Ubuntu, it can be installed by running following commands

```
$ sudo apt-get update
$ sudo apt-get install default-jre
```

1.1.3 Run NLTK Server

```
python wsgi.py
```

1.2 API Documentation

1.2.1 Sentence Tokenizer

`POST /sent_tokenize`

Takes a document and return a array of sentences. Uses nltk.sent_tokentize.

Example request:

```
POST /sent_tokenize HTTP/1.1
Host: example.com
Accept: application/json

Lorem Ipsum is simply dummy text of the printing. Lorem Ipsum has been the industry's standard dummy
```

Example response:

```
HTTP/1.1 200 OK
Vary: Accept
Content-Type: application/json

{
  "result": [
    "Lorem Ipsum is simply dummy text of the printing.",
    "Lorem Ipsum has been the industry's standard dummy text, when an unknown printer took a galley
    "It has survived not only five centuries, but also the leap into electronic typesetting, remainin
  ],
  "status": "success"
}
```

1.2.2 Word Tokenizer

`POST /word_tokenize`

Takes a sentence and tokenizes into words. Uses nltk.word_tokenize.

Example request:

```
POST /word_tokenize HTTP/1.1
Host: example.com
Accept: application/json

Lorem Ipsum is simply dummy text of the printing.
```

Example response:

```
HTTP/1.1 200 OK
Content-Length: 164
Content-Type: application/json
Date: Wed, 24 Dec 2014 01:58:15 GMT
Server: Werkzeug/0.9.6 Python/2.7.6

{
  "result": [
    "Lorem",
    "Ipsum",
    "is",
```

```
    "simply",
    "dummy",
    "text",
    "of",
    "the",
    "printing"
  ],
  "status": "success"
}
```

1.2.3 Part of Speech Tagging

POST /pos_tag

Takes a array of words tokenized by the word tokenizer.

Example request:

```
POST /pos_tag HTTP/1.1
Host: example.com
Accept: application/json

[
  "Lorem",
  "Ipsum",
  "is",
  "simply",
  "dummy",
  "text",
  "of",
  "the",
  "printing"
]
```

Example response:

```
HTTP/1.1 200 OK
Content-Length: 164
Content-Type: application/json
Date: Wed, 24 Dec 2014 02:12:15 GMT
Server: Werkzeug/0.9.6 Python/2.7.6

{
"result": [
  [
    "Lorem",
    "NNP"
  ],
  [
    "Ipsum",
    "NNP"
  ],
  [
    "is",
    "VBZ"
  ],
  [
    "simply",
```

```
    "RB"
  ],
  [
    "dummy",
    "JJ"
  ],
  [
    "text",
    "NN"
  ],
  [
    "of",
    "IN"
  ],
  [
    "the",
    "DT"
  ],
  [
    "printing",
    "NN"
  ]
],
"status": "success"
}
```

1.2.4 Stemming

POST /stem/ (**string:** *algorithm*)

Takes an array of words and return the stem of words. The valid algorithms are 'porter', 'lancaster' and 'snowball'.

Example request:

```
POST /stem/porter HTTP/1.1
Host: example.com
Accept: application/json

[
  "the",
  "buses",
  "are",
  "crowded"
]
```

Example response:

```
HTTP/1.1 200 OK
Content-Length: 212
Content-Type: application/json
Date: Wed, 24 Dec 2014 06:45:29 GMT

{
"result": [
  [
    "the",
    "the"
  ],
```

```
  [
    "buses",
    "buse"
  ],
  [
    "are",
    "are"
  ],
  [
    "crowded",
    "crowd"
  ]
],
"status": "success"
}
```

1.2.5 Lemmatizing

POST /lemmatize/wordnet

Takes an array of words or words with corressponing POS Tag. POS Tag is optional and by default every word is considered noun. Both Wordnet and Penn style Tags are supported. Example for both of them are below.

Example request without POS Tag:

```
POST /lemmatize/wordnet HTTP/1.1
HOST: example.com
Accept: application/json

[
  "the",
  "buses",
  "are",
  "crowded"
]
```

Example response:

```
HTTP/1.1 200 OK
Content-Length: 213
Content-Type: application/json
Date: Sat, 27 Dec 2014 21:19:54 GMT

{
"result": [
  [
    "the",
    "the"
  ],
  [
    "buses",
    "bus"
  ],
  [
    "are",
    "are"
  ],
  [
```

```
    "crowded",
    "crowded"
  ]
],
"status": "success"
}
```

Example request with POS Tag:

```
POST /lemmatize/wordnet HTTP/1.1
HOST: example.com
Accept: application/json

[
  [
    "the",
    "DT"
  ],
  [
    "buses",
    "NNS"
  ],
  [
    "are",
    "VBP"
  ],
  [
    "crowded",
    "VBN"
  ]
]
```

Example Response:

```
HTTP/1.1 200 OK
Content-Length: 210
Content-Type: application/json
Date: Sat, 27 Dec 2014 21:44:28 GMT

{
  "result": [
    [
      "the",
      "the"
    ],
    [
      "buses",
      "bus"
    ],
    [
      "are",
      "be"
    ],
    [
      "crowded",
      "crowd"
    ]
  ],
  "status": "success"
```

```
}
```

1.3 Named Entity Recognition

POST /stanfordNER

This API uses the Stanford NER Library. You can read more details about this project on http://nlp.stanford.edu/software/CRF-NER.shtml.

The API requires JRE to be installed.

Example request:

```
POST /stanfordNER HTTP/1.1
Host: example.com
Accept: application/json

Rami Eid is studying at Stony Brook University in NY.
```

Example response:

```
HTTP/1.1 200 OK
Content-Length: 479
Content-Type: application/json
Date: Tue, 30 Dec 2014 19:23:14 GMT
Server: Werkzeug/0.9.6 Python/2.7.6

{
"result": [
  [
    "Rami",
    "PERSON"
  ],
  [
    "Eid",
    "PERSON"
  ],
  [
    "is",
    "O"
  ],
  [
    "studying",
    "O"
  ],
  [
    "at",
    "O"
  ],
  [
    "Stony",
    "ORGANIZATION"
  ],
  [
    "Brook",
    "ORGANIZATION"
  ],
```

```
    [
      "University",
      "ORGANIZATION"
    ],
    [
      "in",
      "O"
    ],
    [
      "NY",
      "O"
    ]
],
"status": "success"
}
```

/lemmatize

`POST /lemmatize/wordnet`, 7

/pos_tag

`POST /pos_tag`, 5

/sent_tokenize

`POST /sent_tokenize`, 4

/stanfordNER

`POST /stanfordNER`, 9

/stem

`POST /stem/(string:algorithm)`, 6

/word_tokenize

`POST /word_tokenize`, 4