Dive Into NLTK, Part II: Sentence Tokenize and Word Tokenize

Posted on April 15, 2014 by TextMiner

This is the second article in the series "Dive Into NLTK", here is an index of all the articles in the series that have been published to date:

Tokenizers is used to divide strings into lists of substrings. For example, Sentence tokenizer can be used to find the list of sentences and Word tokenizer can be used to find the list of words in strings.

Tokenizing text into sentences

Sentence Tokenize also known as Sentence boundary disambiguation, Sentence boundary detection, Sentence segmentation, here is the definition by wikipedia:

> *Sentence boundary disambiguation (SBD), also known as sentence breaking, is the problem in natural language processing of deciding where sentences begin and end. Often natural language processing tools require their input to be divided into sentences for a number of reasons. However sentence boundary identification is challenging because punctuation marks are often ambiguous. For example, a period may denote an abbreviation, decimal point, an ellipsis, or an email address – not the end of a sentence. About 47% of the periods in the Wall Street Journal corpus denote abbreviations. As well, question marks and exclamation marks may appear in embedded quotations, emoticons, computer code, and slang. Languages like Japanese and Chinese have unambiguous sentence-ending markers.*

There are many nlp tools include the sentence tokenize function, such as OpenNLP，NLTK, TextBlob, MBSP and etc. Here we will tell the details sentence segmentation by NLTK.

How to use sentence tokenize in NLTK?

After installing nltk and nltk_data , you can launch python and import sent_tokenize tool from nltk:

```
>>> text = “this’s a sent tokenize test. this is sent two. is this sent three? sent 4 is cool! Now it’s your turn.”
>>> from nltk.tokenize import sent_tokenize
>>> sent_tokenize_list = sent_tokenize(text)
>>> len(sent_tokenize_list)
5
>>> sent_tokenize_list
[“this’s a sent tokenize test.”, ‘this is sent two.’, ‘is this sent three?’, ‘sent 4 is cool!’, “Now it’s your turn.”]
>>>
```

sent_tokenize uses an instance of PunktSentenceTokenizer from the nltk. tokenize.punkt module. This instance has already been trained on and works well for many European languages. So it knows what punctuation and characters mark the end of a sentence and the beginning of a new sentence.

sent_tokenize is one of instances of PunktSentenceTokenizer from the nltk.tokenize.punkt module. Tokenize Punkt module has many pre-trained tokenize model for many european languages, here is the list from the nltk_data/tokenizers/punkt/README file:

> *Pretrained Punkt Models — Jan Strunk (New version trained after issues 313 and 514 had been corrected)*
>
> *Most models were prepared using the test corpora from Kiss and Strunk (2006). Additional models have*
> *been contributed by various people using NLTK for sentence boundary detection.*
>
> *For information about how to use these models, please confer the tokenization HOWTO:*
> *http://nltk.googlecode.com/svn/trunk/doc/howto/tokenize.html*
> *and chapter 3.8 of the NLTK book:*
> *http://nltk.googlecode.com/svn/trunk/doc/book/ch03.html#sec-segmentation*
>
> *There are pretrained tokenizers for the following languages:*
>
> *File Language Source Contents Size of training corpus(in tokens) Model contributed by*
> ==
> *czech.pickle Czech Multilingual Corpus 1 (ECI) Lidove Noviny ~345,000 Jan Strunk / Tibor Kiss*
> *Literarni Noviny*
>
> *danish.pickle Danish Avisdata CD-Rom Ver. 1.1. 1995 Berlingske Tidende ~550,000 Jan Strunk / Tibor Kiss*
> *(Berlingske Avisdata, Copenhagen) Weekend Avisen*
>
> *dutch.pickle Dutch Multilingual Corpus 1 (ECI) De Limburger ~340,000 Jan Strunk / Tibor Kiss*

> *english.pickle English Penn Treebank (LDC) Wall Street Journal ~469,000 Jan Strunk / Tibor Kiss*
> *(American)*
>
> *estonian.pickle Estonian University of Tartu, Estonia Eesti Ekspress ~359,000 Jan Strunk / Tibor Kiss*
>
> *finnish.pickle Finnish Finnish Parole Corpus, Finnish Books and major national ~364,000 Jan Strunk / Tibor Kiss*
> *Text Bank (Suomen Kielen newspapers*
> *Tekstipankki)*
> *Finnish Center for IT Science*
> *(CSC)*
>
> *french.pickle French Multilingual Corpus 1 (ECI) Le Monde ~370,000 Jan Strunk / Tibor Kiss*
> *(European)*
>
> *german.pickle German Neue Zürcher Zeitung AG Neue Zürcher Zeitung ~847,000 Jan Strunk / Tibor Kiss*
> *(Switzerland) CD-ROM*
> *(Uses “ss”*
> *instead of “ß”)*
>
> *greek.pickle Greek Efstathios Stamatatos To Vima (TO BHMA) ~227,000 Jan Strunk / Tibor Kiss*
>
> *italian.pickle Italian Multilingual Corpus 1 (ECI) La Stampa, Il Mattino ~312,000 Jan Strunk / Tibor Kiss*
>
> *norwegian.pickle Norwegian Centre for Humanities Bergens Tidende ~479,000 Jan Strunk / Tibor Kiss*
> *(Bokmål and Information Technologies,*
> *Nynorsk) Bergen*
>
> *polish.pickle Polish Polish National Corpus Literature, newspapers, etc. ~1,000,000 Krzysztof Langner*
> *(http://www.nkjp.pl/)*
>
> *portuguese.pickle Portuguese CETENFolha Corpus Folha de São Paulo ~321,000 Jan Strunk / Tibor Kiss*
> *(Brazilian) (Linguateca)*

> *slovene.pickle Slovene TRACTOR Delo ~354,000 Jan Strunk / Tibor Kiss*
> *Slovene Academy for Arts*
> *and Sciences*
>
> *spanish.pickle Spanish Multilingual Corpus 1 (ECI) Sur ~353,000 Jan Strunk / Tibor Kiss*
> *(European)*
>
> *swedish.pickle Swedish Multilingual Corpus 1 (ECI) Dagens Nyheter ~339,000 Jan Strunk / Tibor Kiss*
> *(and some other texts)*
>
> *turkish.pickle Turkish METU Turkish Corpus Milliyet ~333,000 Jan Strunk / Tibor Kiss*
> *(Türkçe Derlem Projesi)*
> *University of Ankara*
>
> *The corpora contained about 400,000 tokens on average and mostly consisted of newspaper text converted to*
> *Unicode using the codecs module.*
>
> *Kiss, Tibor and Strunk, Jan (2006): Unsupervised Multilingual Sentence Boundary Detection. Computational Linguistics 32: 485-525.*
>
> *—- Training Code —-*
>
> ```
> # import punkt
> import nltk.tokenize.punkt
>
> # Make a new Tokenizer
> tokenizer = nltk.tokenize.punkt.PunktSentenceTokenizer()
>
> # Read in training corpus (one example: Slovene)
> import codecs
> text = codecs.open(“slovene.plain”,”Ur”,”iso-8859-2").read()
>
> # Train tokenizer
> tokenizer.train(text)
>
> # Dump pickled tokenizer
> import pickle
> out = open(“slovene.pickle”,”wb”)
> pickle.dump(tokenizer, out)
> out.close()
> ```

There are total 17 european languages that NLTK support for sentence tokenize, and you can use them as the following steps:

```
>>> import nltk.data
>>> tokenizer = nltk.data.load(‘tokenizers/punkt/english.pickle’)
>>> tokenizer.tokenize(text)
[“this’s a sent tokenize test.”, ‘this is sent two.’, ‘is this sent three?’, ‘sent 4 is cool!’, “Now it’s your turn.”]
```

Here is a spanish sentence tokenize example:

```
>>> spanish_tokenizer = nltk.data.load(‘tokenizers/punkt/spanish.pickle’)
>>> spanish_tokenizer.tokenize(‘Hola amigo. Estoy bien.’)
[‘Hola amigo.’, ‘Estoy bien.’]
>>>
```

Tokenizing text into words

Tokenizing text into words in NLTK is very simple, just called word_tokenize from nltk.tokenize module:

```
>>> from nltk.tokenize import word_tokenize
>>> word_tokenize(‘Hello World.’)
[‘Hello’, ‘World’, ‘.’]
>>> word_tokenize(“this’s a test”)
[‘this’, “‘s”, ‘a’, ‘test’]
```

Actually, word_tokenize is a wrapper function that calls tokenize by the TreebankWordTokenizer, here is the code in NLTK:

```
# Standard word tokenizer.
_word_tokenize = TreebankWordTokenizer().tokenize
def word_tokenize(text):
    """
    Return a tokenized copy of *text*,
    using NLTK's recommended word tokenizer
    (currently :class:`.TreebankWordTokenizer`).
    This tokenizer is designed to work on a sentence at a time.
    """
    return _word_tokenize(text)
```

Another equivalent call method like the following:

```
>>> from nltk.tokenize import TreebankWordTokenizer
>>> tokenizer = TreebankWordTokenizer()
>>> tokenizer.tokenize(“this’s a test”)
[‘this’, “‘s”, ‘a’, ‘test’]
```

Except the TreebankWordTokenizer, there are other alternative word tokenizers, such as PunktWordTokenizer and WordPunktTokenizer.

PunktTokenizer splits on punctuation, but keeps it with the word:

```
>>> from nltk.tokenize import PunktWordTokenizer
>>> punkt_word_tokenizer = PunktWordTokenizer()
>>> punkt_word_tokenizer.tokenize(“this’s a test”)
[‘this’, “‘s”, ‘a’, ‘test’]
```

WordPunctTokenizer splits all punctuations into separate tokens:

```
>>> from nltk.tokenize import WordPunctTokenizer
>>> word_punct_tokenizer = WordPunctTokenizer()
>>> word_punct_tokenizer.tokenize(“This’s a test”)
[‘This’, “‘”, ‘s’, ‘a’, ‘test’]
```

You can choose any word tokenizer in nltk for your using purpose.

Posted by TextMiner

Related Posts:

1. **Dive Into NLTK, Part I: Getting Started with NLTK**
2. **We have launched the Text Analysis API on Mashape**
3. **Text Analysis Online no longer provides NLTK Stanford NLP API Interface**
4. **Dive Into NLTK, Part X: Play with Word2Vec Models based on NLTK Corpus**

Posted in NLP, NLTK, Text Analysis, Text Mining Tagged NLP, NLTK, nltk word tokenize, Sent Tokenize, Sentence Boundary Detection, Sentence Segmentation, sentence tokenizer, Text Analysis, text analysis online, Text Mining, text mining online, Word Tokenize, word tokenizer

permalink [http://textminingonline.com/dive-into-nltk-part-ii-sentence-tokenize-and-word-tokenize]

Comments

***Dive Into NLTK, Part II: Sentence Tokenize and Word Tokenize* — 11 Comments**

Pingback: Dive Into NLTK, Part I: Getting Started with NLTK | Text Mining Online | Text Analysis Online

Pingback: Dive Into NLTK, Part VI: Add Stanford Word Segmenter Interface for Python NLTK | Text Mining Online | Text Analysis Online | Text Processing Online

sourabh kulhare
on November 4, 2014 at 4:57 am said:

I want to process each sentence separately, means take a random text and then work on each sentence of that text to identify that which class is associated to each sentence of that text. So to process on each sentence of the text what function and tool I should use.?
thanks

TextMiner
on November 6, 2014 at 3:31 am said:

Use sent_tokenize(text) to get each sentence

Fred
on July 3, 2015 at 3:51 pm said:

Hello, if I want to parse phrase, not only single word,
using word_tokenize seems not tokenize the phrase?

Desh Raj
on February 5, 2017 at 7:07 pm said:

You can use the MWE (multi-word expression) tokenizer available in NLTK for this purpose.
from nltk.tokenize.mwe import MWETokenizer

Kantajit Shaw
on July 9, 2015 at 2:19 pm said:

I am new to nltk. I was trying some basics.

```
import nltk
nltk.word_tokenize(“Tokenize me”)
```

gives me this following error

```
Traceback (most recent call last):
File “”, line 1, in
nltk.word_tokenize(“hi im no onee”)
File “C:\Python27\lib\site-packages\nltk\tokenize\__init__.py”, line 101, in word_tokenize
return [token for sent in sent_tokenize(text, language)
File “C:\Python27\lib\site-packages\nltk\tokenize\__init__.py”, line 85, in sent_tokenize
tokenizer = load(‘tokenizers/punkt/{0}.pickle’.format(language))
File “C:\Python27\lib\site-packages\nltk\data.py”, line 786, in load
resource_val = pickle.load(opened_resource)
AttributeError: ‘module’ object has no attribute ‘defaultdict’
```

Please help. Please tell me how to fix this error.

wuxinle
on November 22, 2016 at 7:13 am said:

```
import nltk
nltk.download()
```

Pinkesh
on October 1, 2015 at 10:40 am said:

I would like to make tokenizer for my natural language that is Gujarati language. i make .pickle file for it and sentence tokenization is done using tokenizer.tokenize(text) but how to make word tokenization for same. How tokenizer.train(text) works?

Christian
on September 26, 2016 at 12:11 pm said:

Does the sentence_tokenizer only identify punctuation as the tend of a sentence or is it also possible to identify a sentence when punctuation is missing?

Do you have any rules or tools to recommend?

Maha
on April 6, 2017 at 6:04 pm said:

I want to read a text file and segment its sentences. Is it possible? How?