Faculty of Computers and AI
Cairo University
NLP - 2024

Lab2: Introduction to NLTK

Lab Objectives

- Introduce NLTK package.
- Learn how to use NLTK to perform different types of processing for text.

NLTK: Natural Language Toolkit

- NLTK is a leading platform for building Python programs to work with human language data.
- It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet,
- It provides a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.

Task1: Tokenize text

Tokenizing is the task of dividing a string into substrings by splitting on the specified string. For example **Word tokenizing** is the task of dividing a string into words; **Sentence tokenizing** is the task of dividing a string into sentences. Check the provided code to see the different outputs for each task below:

1. **Word tokenizing using regex**
   You should be able to use the first lab to write a regular expression that combines most common English word delimiters and use the split function of re library.
2. **Word tokenizing using NLTK**
   Check the differences in tokens in the regex output and NLTK output
3. **Sentence tokenizing using NLTK**

Task2: Stemming

Stemming is the task of removing morphological affixes from words, leaving only the word stem. NLTK provides multiple types of stemmers but **Porter Stemmer** is the most popular one. It can correctly handle words like lying (which is mapped to lie, not ly). Check the code provided for task 2 to see and compare different NLTK stemmers.

Task3: Lemmatizing

Stemming of a word may result in words that are not valid words in the language. Lemmatization, unlike Stemming, reduces the inflected words properly ensuring that the root word belongs to the language. In Lemmatization root word is called **Lemma or dictionary form**. For example, runs, running, ran are all forms of the word run, therefore run is the lemma.

- NLTK provides **WordNet Lemmatizer** that uses the WordNet Database to lookup lemmas of words.
- **Note: Download the WordNet corpora from NLTK downloader before using the WordNet Lemmatizer**

Check the code provided for task 3 to see an example of NLTK lemmatizer.

Task4: Part-Of-Speech (POS) Tagging

POS Tagging is the process of classifying words into their **parts of speech** and labeling them accordingly. Parts of speech are also known as **word classes** or **lexical categories**. NLKT provide a **POS-tagger** that processes a sequence of words, and attaches a POS tag to each word.

Check the code provided for task 4 to see an example of NLTK POS Tagging.

- To get list of POS tags used by NLTK use this statement:

```
nltk.help.upenn_tagset()
```

Task5: Using Wordnet

- WordNet is a large lexical database of English (semantically oriented dictionary).
- Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept.
- Synsets are interlinked by means of conceptual-semantic and lexical relations.
- The result is a network of meaningfully related words and concepts. You can test it online from this link:

http://wordnetweb.princeton.edu/perl/webwn

The Wordnet hierarchy:

Synsets (synonyms sets) form relations with other synsets to form a hierarchy of concepts, ranging from very general ("entity", "state") to moderately abstract ("animal") to very specific ("plankton").

For a given synset,

(1) “is-a” relations:

- **hypernyms** are the synsets that are more general
- **hyponyms** are the synsets that are more specific. Note that Hyponyms have an "is-a" relationship to their hypernyms

(2) "is-made-of" and "comprises" relationships.

- **holonyms** are things that the item is contained in
- **meronyms** are components or substances that make up the item

Semantic similarity

The similarity between two synsets can be measure using different methods. One method is **Wu-Palmer Similarity. Wu-Palmer Similarity** Return a score denoting how similar two word senses are, based on the depth of the two senses in the taxonomy and that of their Least Common Subsumer (most specific ancestor node).

Hint: compare it to other methods, for example path similarity

Using NLTK Wordnet corpus:

Check the code provided for task 5 to see an example.

Student task:

Use the previous functions you learned, and given the provided paragraph of multiple sentences, write 2 functions, one that returns the processed version of the paragraph based on stemming, and the second returns processed version of the paragraph based on lemmatizing.

The task is as follow:

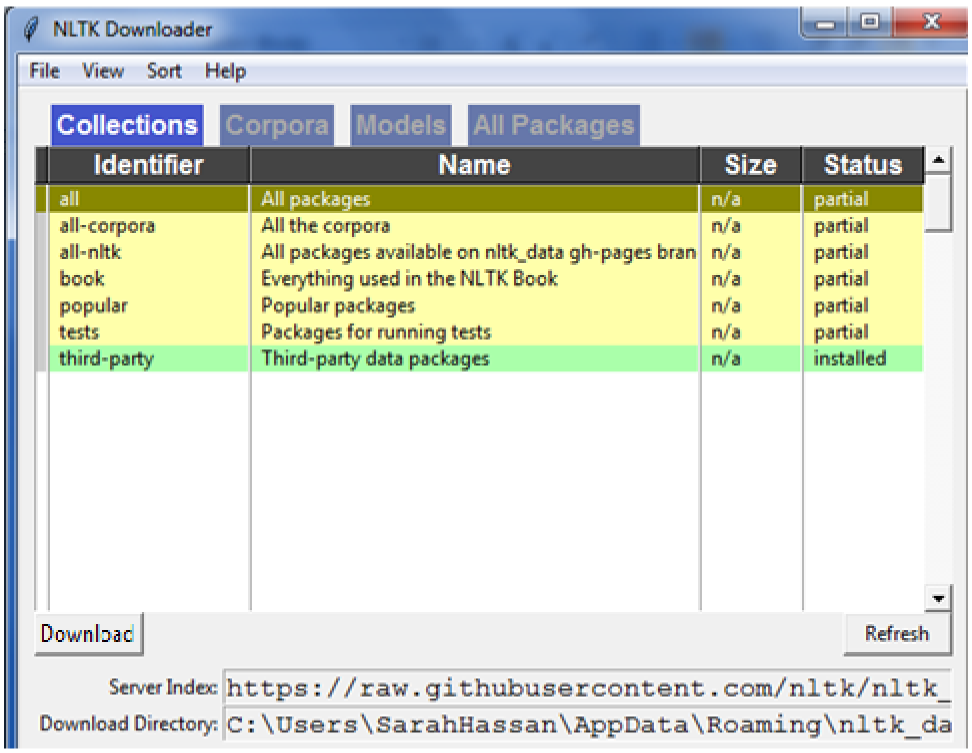
(1) Separate the paragraph into sentences
(2) Separate each sentence into tokens
(3) Remove un-necessary tokens (e.g. punctuations)
(4) Stem/lemmatize each token
(5) Generate each sentence as a list of processed tokens
(6) Generate the processed paragraph as a list of processed sentences

Hint: use multiple functions as needed for code reusability.

Download NLTK packages/corpuses

In the python command prompt

1) Import NLTK
2) Type: nltk.download()
3) From the dialogue box that will open choose “all” row and click download:



Identifier	Name	Size	Status
all	All packages	n/a	partial
all-corpora	All the corpora	n/a	partial
all-nltk	All packages available on nltk_data gh-pages bran	n/a	partial
book	Everything used in the NLTK Book	n/a	partial
popular	Popular packages	n/a	partial
tests	Packages for running tests	n/a	partial
third-party	Third-party data packages	n/a	installed

Server Index: https://raw.githubusercontent.com/nltk/nltk_

Download Directory: C:\Users\SarahHassan\AppData\Roaming\nltk_da

Useful Links

https://pythonprogramming.net/wordnet-nltk-tutorial/

http://www.nltk.org/howto/wordnet.html