ISSN:0975-9646

Natural Language Processing using NLTK and WordNet

Alabhya Farkiya, Prashant Saini, Shubham Sinha
Computer Department
MIT College of Engineering (MITCOE)
Pune (India)

Sharmishta Desai
Assistant Professor
Computer Department
MIT College of Engineering (MITCOE)
Pune (India).

***Abstract*-- Natural Language Processing is a theoretically motivated range of computational techniques for analysing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications [1]. To perform natural language processing a variety of tools and platform have been developed, in our case we will discuss about NLTK for Python.The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for the Python programming language[2]. It provides easy-to-use interfaces to many corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning. In this paper we discuss different approaches for natural language processing using NLTK.**

Keywords: NLP, NLTK, semantic reasoning.

1. INTRODUCTION

1.1 NLP

Natural language processing (NLP) is a field of computer science, artificial intelligence, and computational linguistics concerned with the interactions between computers and human (natural) languages [3].

The goal of natural language processing is to allow that kind of interaction so that non-programmers can obtain useful information from computing systems. Natural language processing also includes the ability to draw insights from data contained in emails, videos, and other unstructured material. The various aspects of NLP include Parsing, Machine Translation, Language Modelling, Machine Learning, Semantic Analysis etc. In this paper we only focus on semantic analysis aspect of NLP using NLTK.

1.2 NLTK

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, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.NLTK includes graphical demonstrations and sample data. NLTK is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, cognitive science, artificial intelligence, information retrieval, and machine learning.We discuss how we can perform semantic analysis in NLP using NLTK as a platform for different corpora. Adequate representation of natural language semantics requires access to vast amounts of common sense and domain-specific world knowledge. We focus our efforts on using WordNet as a preferred corpora for using NLTK.

1.3 WordNet

[5]Because meaningful sentences are composed of meaningful words, any system that hopes to process natural languages as people do must have information about words and their meanings. This information is traditionally provided through dictionaries, and machine-readable dictionaries are now widely available. But dictionary entries evolved for the convenience of human readers, not for machines. WordNet provides a more effective combination of traditional lexicographic information and modern computing. WordNet is an online lexical database designed for use under program control. English nouns, verbs, adjectives, and adverbs are organized into sets of synonyms, each representing a lexicalized concept. Semantic relations link the synonym sets. Using NLTK and WordNet, we can form semantic relations and perform semantic analysis on texts, strings and documents.

1.4 Python

Python is a dynamic object-oriented programming language. It offers strong support for integrating with other technologies, higher programmer productivity throughout the development life cycle, and is particularly well suited for large or complex projects with changing requirements.

Python has a very shallow learning curve and an excellent online learning resource with support of innumerable libraries.

2. NATURAL LANGUAGE PROCESSING

Natural language processing refers to the use and capability of systems to process sentences in a natural language such as English, rather than in a specialized artificial computer language such as Java.

[12]Some basic terminologies can aid in better understanding of natural language processing.

- Token: Tokens are linguistic units such as words, punctuation, numbers or alphanumeric.
- Sentence: An ordered sequence of tokens.
- Tokenization: The process of breaking a sentence into its constituent tokens. For segmented languages such as English, the presence of white space makes tokenization relatively easy.
- Corpus: A body of text, usually containing multiple number of sentences with semantic structures within it.
- Part-of-speech (POS) Tag: A word can be categorized into one or more of a set of lexical or part-of-speech categories such as Nouns, Verbs, Adjectives and Articles, to name a few. A POS tag is a symbol representing such a lexical category.
- Parse Tree: A tree defined over a given sentence that showcase the syntactic structure of the sentence as stated by a formal grammar.

Broadly construed, natural language processing (with respect to the interpretation side) is considered to involve the following subtopics:

- Syntactic analysis
- Semantic analysis
- Pragmatics

Syntactic analysis includes consideration of morphological and syntactic knowledge on the part of the natural language processor, semantic analysis includes consideration of semantic knowledge, and pragmatics includes consideration of pragmatic, discourse, and world knowledge [6].
We perform syntactic analysis and semantic analysis by benefitting from the structures in the WordNet library and comparing them using NLTK. But, Pragmatics still remains out of the domain of this approach.

3. Using NLTK

3.1 Introduction

The Natural Language Toolkit is a collection of program modules, data sets, tutorials and exercises, covering symbolic and statistical natural language processing. NLTK is written in Python and distributed under the GPL open source license.
NLTK is implemented as a large collection of minimally interdependent modules, organized into a shallow hierarchy [7]. A set of core modules defines basic data types that are used throughout the toolkit. The remaining modules are task modules, each devoted to an individual natural language processing task. For example, the nltk.parser module encompasses to the task of parsing, or deriving the syntactic structure of a sentence; and the nltk.tokenizer module is devoted to the task of tokenizing, or dividing a text into its constituent parts.

3.2 NLTK corpora

NLTK incorporates several useful text corpora that are used widely for NLP. Some of them are as follows:
Brown Corpus: The Brown Corpus of Standard American English is the first general English corpus that could be used in computational linguistic processing tasks. This corpus consists of one million words of American English texts printed in 1961. For the corpus to represent as general a sample of the English language as possible, 15 different genres were sampled such as Fiction, News and Religious text. Subsequently, a POS-tagged version of the corpus was also created with substantial manual effort.
Gutenberg Corpus: The Gutenberg Corpus is a collection of 14 texts chosen from Project Gutenberg - the largest online collection of free e-books. The corpus contains a total of 1.7 million words.
Stopwords Corpus: Apart from regular content words, there is another class of words called stop words that perform important grammatical functions but are unlikely to be interesting by themselves, such as prepositions, complementizers and determiners. NLTK comes bundled with the Stopwords Corpus - a list of 2400 stop words across 11 different languages (including English).
Apart from these corpora which are shipped with NLTK we can use intelligent sources of data like WordNet or Wikipedia.

3.3 Modules of NLTK

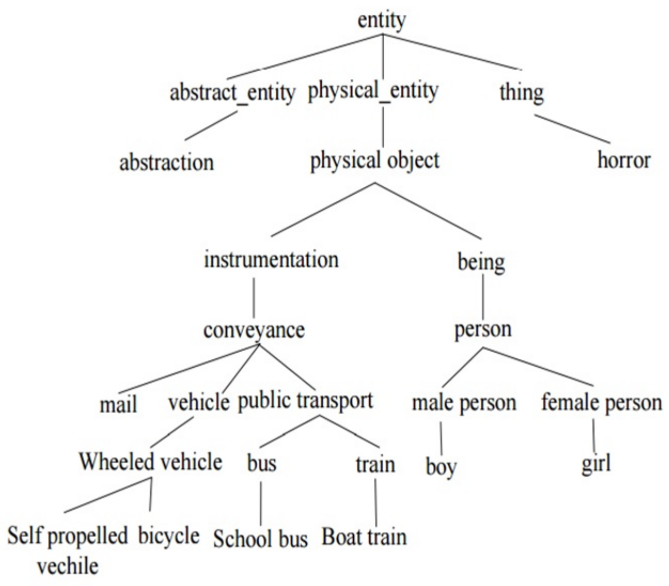
3.3.1 Parsing Modules
The parser module defines a high-level interface for creating trees that represent the structures of texts [7]. The chunkparser module defines a sub-interface for parsers that identify non overlapping linguistic groups (such as base noun phrases) in unrestricted text.Four modules provide implementations for these abstract interfaces.
The srparser module implements a simple shift-reduce parser. The chartparser module defines a flexible parser that uses a chart to record hypotheses about syntactic constituents. The pcfgparser module provides a variety of different parsers for probabilistic grammars. And the rechunkparser module defines a transformational regular-expression based implementation of the chunk parser interface.
3.3.2 Tagging Modules
The tagger module defines a standard interface for extending each token of a text with additive information, such as its part of speech or its WordNet synset tag. It also provides several different implementations for this interface.
3.3.3 Finite State Automata
The fsa module provides an interface for creating automata from regular expressions.
3.3.4 Type Checking
Debugging time is an important factor in the toolkit’s ease of use. To reduce the amount of time students must spend debugging their code, a type checking module is provided, which can be used to ensure that functions are given valid arguments. However, when efficiency is an issue, type checking can be disabled which causes no performance penalty.

3.3.5 Visualization
Visualization modules define graphical interfaces for viewing and manipulating data structures. It also defines graphical tools for experimenting with NLP tasks. The visualization modules provide interfaces for interaction and experimentation i.e. they do not directly implement NLP data structures or tasks. A few visualization modules include draw.tree, draw.tree_edit, draw.plot_graph, draw.fsa and draw.chart.
3.3.6 Text Classification
The classifier module defines a standard interface for classifying texts into categories. This interface is presently being implemented by two modules. The classifier.naivebayes module defines a text classifier based on the Naive Bayes assumption. The classifier.maxent module defines the maximum entropy model for text classification, and implements two algorithms for training the model: Generalized Iterative Scaling and Improved Iterative Scaling.

4. WORDNET

4.1 Introduction
WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept [8]. Synsets are interlinked by means of conceptual-semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated with the browser. WordNet is also freely and publicly available for download. WordNet's structure makes it a useful tool for computational linguistics and natural language processing.
WordNet superficially resembles a thesaurus, in that it groups words together based on their meanings. However, there are some important distinctions. First, WordNet interlinks not just word forms—strings of letters—but specific senses of words. As a result, words that are found in close proximity to one another in the network are semantically disambiguated. Second, WordNet labels the semantic relations among words, whereas the groupings of words in a thesaurus does not follow any explicit pattern other than meaning similarity.

4.2 Knowledge Structure
WordNet is particularly well suited for similarity measures, since it organizes nouns and verbs into hierarchies of is–a relations [9].For instance, one sense of the word *dog* is found following hypernym hierarchy; the words at the same level represent synset members [10]. Each set of synonyms has a unique index.
dog, domestic dog, Canis familiaris
=> canine, canid
=> carnivore
=> placental, placental mammal, eutherian, eutherian mammal
=> mammal
=> vertebrate, craniate
=> chordate
=> animal, animate being, beast, brute, creature, fauna
=> ...



A Fragment of is-a Relation in WordNet

In version 2.0, there are nine separate noun hierarchies that include 80,000 concepts, and 554 verb hierarchies that are made up of 13,500 concepts.Is–a relations in WordNet do not cross part of speech boundaries, so similarity measures are limited to making judgments between noun pairs (e.g., cat and dog) and verb pairs (e.g., run and walk). While WordNet also includes adjectives and adverbs, these are not organized into is–a hierarchies so similarity measures cannot be applied.
However, concepts can be related in many ways beyond being similar to each other. For example, a wheel is a part of a car, night is the opposite of day, snow is made up of water, a knife is used to cut bread, and so forth. As such WordNet provides relations beyond is–a, including has–part, is–made–of, and is–an–attribute–of. In addition, each concept is defined by a short gloss that may include an example usage. All of this information can be brought to bear in creating measures of relatedness. As a result these measures tend to be more flexible, and allow for relatedness values to be assigned across parts of speech (e.g., the verb murder and the noun gun).
We are using WordNet for realizing the similarity between pairs of words, strings and documents because its ease of use and Gnu Public License.

4.3 Limitations
WordNet does not include information about the etymology or the pronunciation of words and it contains only limited information about usage [10]. WordNet aims to cover most of everyday English and does not include much domain-specific terminology.
WordNet is the most commonly used computational lexicon of English for word sense disambiguation (WSD), a task aimed to assigning the context-appropriate meanings (i.e. synset members) to words in a text. However, it has been argued that WordNet encodes sense distinctions that are too fine-grained. This issue prevents WSD systems from achieving a level of performance comparable to that

of humans, who do not always agree when confronted with the task of selecting a sense from a dictionary that matches a word in a context. The granularity issue has been tackled by proposing clustering methods that automatically group together similar senses of the same word.

4.4 Applications

WordNet has been used for a number of different purposes in information systems, including word sense disambiguation, information retrieval, automatic text classification, automatic text summarization, machine translation and even automatic crossword puzzle generation [10].

A common use of WordNet is to determine the similarity between words. Various algorithms have been proposed, and these include measuring the distance among the words and synsets in WordNet's graph structure, such as by counting the number of edges among synsets. The intuition is that the closer two words or synsets are, the closer their meaning. A number of WordNet-based word similarity algorithms are implemented in a Perl package called WordNet::Similarity, and in a Python package called NLTK. Other more sophisticated WordNet-based similarity techniques include ADW, whose implementation is available in Java. WordNet can also be used to inter-link other vocabularies.

5. Measuring Semantic Similarity In Wordnet

Semantic similarity measure is a central issue in artificial intelligence, psychology and cognitive science for many years [11]. It has been widely used in natural language processing, information retrieval, word sense disambiguation, text segmentation etc.

Many semantic similarity measures have been proposed. On the whole, all the measures can be grouped into four classes: path length based measures, information content based measures, feature based measures, and hybrid measures.

5.1. Path-based Measures

The main idea of path-based measures is that the similarity between two concepts is a function of the length of the path linking the concepts and the position of the concepts in the taxonomy.

5.2. Information Content-based Measure

It assumed that each concept includes much information in WordNet. Similarity measures are based on the Information content of each concept. The more common information two concepts share, the more similar the concepts are.

5.3. Feature-based Measure

Different from all the above presented measures, feature-based measure is independent on the taxonomy and the subsumers of the concepts, and attempts to exploit the properties of the ontology to obtain the similarity values. It is based on the assumption that each concept is described by a set of words indicating its properties or features, such as their definitions or “glosses” in WordNet. The more common characteristics two concepts have and the less non-common characteristics they have, the more similar the concepts are.

5.4. Hybrid Measure

The hybrid measures combine the ideas above presented. In practice many measures not only combine the ideas above, but also combine the relations, such as is-a, part-of and so on. A typical method is proposed by Rodriguez. The similarity function includes three parts: synonyms sets, neighbourhoods and features.

6. Calculating Semantic Similarity And Relatedness

Our approach to calculate similarity is as follows:

1. Remove the stopwords from both the sentences using a database for stopwords in WordNet.
2. Tokenize the sentences without stopwords
3. Compare each word of 1st sentence with the database of given words in 2nd sentence from WordNet.
4. Each comparison returns us a score of similarity.
5. Average out the score for the whole sentence.
6. Python code for doing the above is given below:

```
stop = stopwords.words('english')
goodwords= [i for i in sentences.split() if i not in stop]
goodwords1= [i for i in target_sentence.split() if i not in
stop]
m=0
n=0
l=[]
fl=[]
for m,p in enumerate(goodwords):
for n,q in enumerate (goodwords1):
xx = wn.synsets(p)
y = wn.synsets(q)[0]
del l[:]
for x in xx:
if (x.wup_similarity(y))==None:
l.append(0)
else:
l.append(x.wup_similarity(y))
try:
fl.append(max(l))
except:
fl.append(0)
score=sum(fl)/len(fl)
```

7. RELATED WORK

In this section we briefly introduce some related works in WordNet, NLTK and NLP.

Lingling Meng et al. [11] have illustrated different measures of semantic similarity in WordNet i.e path-based measures, Information Content-based Measures, Feature-based Measures, and Hybrid Measures.

George A. Miller [5] has focussed on the basic concepts involved in understanding WordNet as a lexical database.

Elizabeth D. Liddy [1] has presented the basics of Natural Language processing.

Nitin Madnani [12] helps us in getting started on Natural Language Processing with Python i.e. NLTK.

Ted Pedersen et al.[9] have written about WordNet::Similarity, a freely available software package that makes it possible to measure the semantic similarity and relatedness between a pair of concepts (or synsets).

Edward Loper et al. [7] have explained working of NLTK,the Natural Language Toolkit.

8. CONCLUSION

In this paper we have presented and discussed various aspects of natural language processing, NLTK and semantic analysis. Also, we discuss measuring semantic similarity for a pair of words, strings and documents. We have also highlighted the advantages of using a lexical database like WordNet.

REFERENCES

[1] Liddy, E.D. 2001. Natural Language Processing. In Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker, Inc.

[2] http://en.wikipedia.org/wiki/Natural_Language_Toolkit

[3] http://en.wikipedia.org/wiki/Natural_language_processing

[4] nltk.org

[5] George A. Miller. WordNet: A Lexical Database for English in COMMUNICATIONS OF THE ACM November 1995/Vol. 38, No. 11

[6] http://www.mind.ilstu.edu/curriculum/protothinker natural_language_processing.php

[7] Edward Loper and Steven Bird. NLTK: The Natural Language Toolkit at Department of Computer and Information Science University of Pennsylvania, Philadelphia, PA 19104-6389, USA

[8] https://wordnet.princeton.edu

[9] Ted Pedersen,Siddharth Patwardhan,Jason Michelizzi. WordNet::Similarity - Measuring the Relatedness of Concepts

[10] http://en.wikipedia.org/wiki/WordNet

[11] Lingling Meng et al. A Review of Semantic Similarity Measures in WordNet at International Journal of Hybrid Information Technology Vol. 6, No. 1, January, 2013

[12] Nitin Madnani. Getting Started on Natural Language Processing with Python.