NLTK for Information Extract

Net Id: KG24

What is NLTK?

NLTK is a leading platform for building Python programs to work with human language data. It also 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, semantic reasoning and wrappers for industrial-strength NLP libraries. NLTK has been described as a wonderful tool for teaching, and working in, computational linguistics using Python.

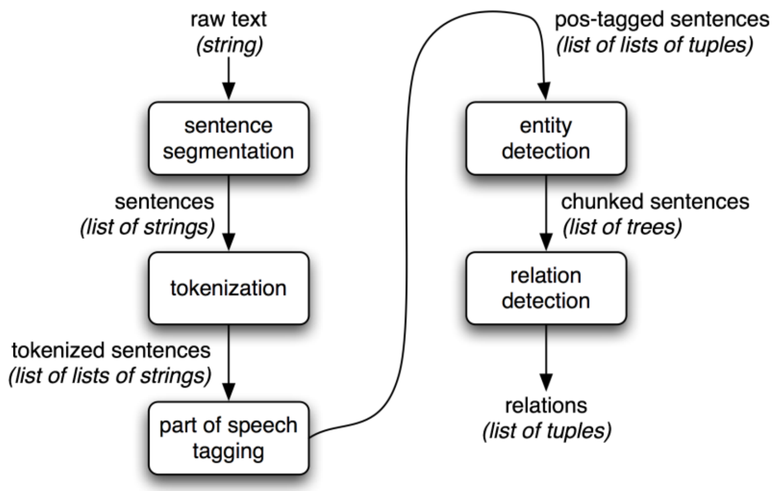
This paper describes about extracting information from Text using NLTK tool kit.

Information Extraction

The amount of natural language text that is available in electronic form is truly staggering, and is increasing every day. However, the complexity of natural language can make it very difficult to access the information in that text. In case of structured data, extracting entities and its relationships is easier compared to the unstructured data. Information Extraction deals with extracting information from the unstructured data by converting it into structured data first. Information extraction has many benefits, such as business intelligence, resume harvesting, media analysis, sentiment detection, patent search, and email scanning.

Information Extraction Architecture

Information Extraction begins by processing a document using several of the procedures like tokenization, categorizing and tagging of words: first, the raw text of the document is split into sentences using a sentence segmenter, and each sentence is further subdivided into words using a tokenizer. Next, each sentence is tagged with part-of-speech tags, which will prove very helpful in the next step, named entity detection. In this step, we search for mentions of potentially interesting entities in each sentence. Finally, we use relation detection to search for likely relations between different entities in the text. The below diagrams depicts the architecture of Information Extraction.



NLTK for information Extraction

NLTK provides modules for Information Extraction and thus, making it more appropriate tool for Information Extraction. The below table shows the language processing tasks and corresponding NLTK modules with examples of functionality

Language processing task	NLTK modules	Functionality
Accessing corpora	corpus	standardized interfaces to corpora and lexicons
String processing	tokenize, stem	tokenizers, sentence tokenizers, stemmers
Collocation discovery	collocations	t-test, chi-squared, point-wise mutual information
Part-of-speech tagging	tag	n-gram, backoff, Brill, HMM, TnT
Machine learning	classify, cluster, tbl	decision tree, maximum entropy, naive Bayes, EM, k-means
Chunking	chunk	regular expression, n-gram, named-entity
Parsing	parse, ccg	chart, feature-based, unification, probabilistic, dependency
Semantic interpretation	sem, inference	lambda calculus, first-order logic, model checking
Evaluation metrics	metrics	precision, recall, agreement coefficients

Language processing task	NLTK modules	Functionality
Probability and estimation	probability	frequency distributions, smoothed probability distributions
Applications	app, chat	graphical concordancer, parsers, WordNet browser, chatbots
Linguistic fieldwork	toolbox	manipulate data in SIL Toolbox format

Information extraction systems search large bodies of unrestricted text for specific types of entities and relations, and use them to populate well-organized databases. These databases can then be used to find answers for specific questions.

The typical architecture for an information extraction system begins by segmenting, tokenizing, and part-of-speech tagging the text. The resulting data is then searched for specific types of entity. Finally, the information extraction system looks at entities that are mentioned near one another in the text, and tries to determine whether specific relationships hold between those entities.

Entity recognition is often performed using chunkers, which segment multi-token sequences, and label them with the appropriate entity type. Common entity types include ORGANIZATION, PERSON, LOCATION, DATE, TIME, MONEY, and GPE (geo-political entity).

Chunkers can be constructed using rule-based systems, such as the RegexpParser class provided by NLTK; or using machine learning techniques, such as the ConsecutiveNPChunker presented in this chapter. In either case, part-of-speech tags are often a very important feature when searching for chunks.

Although chunkers are specialized to create relatively flat data structures, where no two chunks are allowed to overlap, they can be cascaded together to build nested structures.

Relation extraction can be performed using either rule-based systems which typically look for specific patterns in the text that connect entities and the intervening words; or using machine-learning systems which typically attempt to learn such patterns automatically from a training corpus.

References

https://www.nltk.org/

https://www.nltk.org/book/ch07.html