Practical 1. Exploration of different frameworks, libraries and SDKs for Natural Language Processing.

Name: **Aayush Shah** Natural Language Processing **2CSDE70**
Roll Number: **19BCE245** 25 February, 2022

1 Natural Language ToolKit

The NLTK Library (Natural Language Toolkit) is a collection of libraries and tools for statistical language processing. It is one of the most powerful NLP libraries, containing packages for making robots understand human language and respond appropriately.

Natural Language Processing with Python is a hands-on introduction to language processing programming. Written by the NLTK founders, it walks the reader through the principles of Python programming, working with corpora, categorising text, evaluating linguistic structure, and more. The book's online edition has been updated for Python 3 and NLTK 3.

Learning Natural Language Toolkit will allow us to gain a new skill while also expanding your knowledge of NLP. Learning the NLTK library can help professionals advance their careers in AI and Natural Language Processing with Python.

NLTK includes methods for tokenize words, POS, Tokenization, Stemming, Lemmatization, Punctuation, Character count, word count, WordNet, Word Embedding, seq2seq model, and other useful algorithms for natural language processing. We can use NLTK to analyze a text to see whether the sentiments expressed in it are positive or negative.

2 Gensim

Gensim, which stands for "Generate Similar," is a well-known open source natural language processing (NLP) framework for unsupervised topic modelling. It employs top academic models and advanced statistical machine learning to execute a variety of complex tasks such as document or word vector construction. For performance, Gensim is written in Python and Cython.

Using its incremental online training algorithms, Gensim can easily process massive and web-scale corpora. It is scalable since there is no need for the entire input corpus to be fully stored in Random Access Memory (RAM) at any given time. In other words, regardless of the size of the corpus, all of its methods are memory-independent.

As we all know, Python is an extremely versatile language, and Gensim, being pure Python, operates on any platforms that support Python and Numpy (such as Windows, Mac OS, and Linux).

Features

- Other packages, such as 'scikit-learn' and 'R,' give topic modelling and word embedding capabilities, but Gensim's capabilities for generating topic models and word embedding are unsurpassed. It also gives more convenient text processing features

- Another big advantage of Gensim is that it allows us to handle enormous text files without having to load the entire file into memory.

- Because it employs unsupervised models, Gensim does not necessitate costly annotations or manual labelling of texts.

3 PyNLP

PyNLPl, pronounced as ‘pineapple’, is a Python library for Natural Language Processing. It contains various modules useful for common, and less common, NLP tasks. PyNLPl can be used for basic tasks such as the extraction of n-grams and frequency lists, and to build simple language model. There are also more complex data types and algorithms. Moreover, there are parsers for file formats common in NLP (e.g. FoLiA/Giza/Moses/ARPA/Timbl/CQL). There are also clients to interface with various NLP specific servers. PyNLPl most notably features a very extensive library for working with FoLiA XML (Format for Linguistic Annotatation).

The library is a divided into several packages and modules. It works on Python 2.7, as well as Python 3.

The following modules are available:

- pynlpl.datatypes - Extra datatypes (priority queues, patterns, tries)
- pynlpl.evaluation - Evaluation experiment classes (parameter search, wrapped progressive sampling, class evaluation (precision/recall/f-score/auc), sampler, confusion matrix, multithreaded experiment pool)
- pynlpl.formats.cgn - Module for parsing CGN (Corpus Gesproken Nederlands) part-of-speech tags
- pynlpl.formats.folia - Extensive library for reading and manipulating the documents in FoLiA format (Format for Linguistic Annotation).
- pynlpl.formats.fql - Extensive library for the FoLiA Query Language (FQL), built on top of pynlpl.formats.folia. FQL is currently documented here.
- pynlpl.formats.cql - Parser for the Corpus Query Language (CQL), as also used by Corpus Workbench and Sketch Engine. Contains a convertor to FQL.
- pynlpl.formats.giza - Module for reading GIZA++ word alignment data
- pynlpl.formats.moses - Module for reading Moses phrase-translation tables.
- pynlpl.formats.sonar - Largely obsolete module for pre-releases of the SoNaR corpus, use pynlpl.formats.folia instead.
- pynlpl.formats.timbl - Module for reading Timbl output (consider using python-timbl instead though)
- pynlpl.lm.lm - Module for simple language model and reader for ARPA language model data as well (used by SRILM).
- pynlpl.search - Various search algorithms (Breadth-first, depth-first, beam-search, hill climbing, A star, various variants of each)
- pynlpl.statistics - Frequency lists, Levenshtein, common statistics and information theory functions
- pynlpl.textprocessors - Simple tokeniser, n-gram extraction

4 Pattern

Pattern is a web mining module for Python.We have a distinct function in the NLTK and spaCy libraries for tokenizing, POS tagging, and finding noun phrases in text documents. In the Pattern library, however, there is an all-in-one parse method that accepts a text string as an input parameter and returns corresponding tokens in the string, as well as the POS tag.The parse technique also indicates whether a token is a noun phrase, verb phrase, subject, or object. By setting the lemmata option to True, you can also retrieve lemmatized tokens.

It has tools for:

- Data Mining: web services (Google, Twitter, Wikipedia), web crawler, HTML DOM parser
- Natural Language Processing: part-of-speech taggers, n-gram search, sentiment analysis, WordNet
- Machine Learning: vector space model, clustering, classification (KNN, SVM, Perceptron)
- Network Analysis: graph centrality and visualization.

5 Polyglot

Polyglot is a natural language pipeline that supports massive multilingual applications. Polyglot is a Python open-source package that is used to conduct various NLP operations. It is quick because it is built on NumPy. It features a wide range of dedicated commands, which distinguishes it from the competition. It is comparable to spacy and can be used in languages where spacy is not supported.

- Free software: GPLv3 license
- Documentation: http://polyglot.readthedocs.org
- GitHub: https://github.com/aboSamoor/polyglot
- Tokenization (165 Languages)
- Language detection (196 Languages)
- Named Entity Recognition (40 Languages)
- Part of Speech Tagging (16 Languages)
- Sentiment Analysis (136 Languages)
- Word Embeddings (137 Languages)
- Morphological analysis (135 Languages)
- Transliteration (69 Languages)

6 TextBlob

TextBlob is a Python (2 and 3) text processing package. It offers a straightforward API for delving into typical natural language processing (NLP) activities like part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

Features

- Noun phrase extraction
- Part-of-speech tagging
- Sentiment analysis
- Classification (Naive Bayes, Decision Tree)

- Tokenization (splitting text into words and sentences)
- Word and phrase frequencies
- Parsing
- n-grams
- Word inflection (pluralization and singularization) and lemmatization
- Spelling correction
- Add new models or languages through extensions
- WordNet integration

7 Spacy

spaCy is a free, open-source library for NLP in Python. It's written in Cython and is designed to build information extraction or natural language understanding systems. It's built for production use and provides a concise and user-friendly API. If you're working with a lot of text, you'll eventually want to know more about it. For example, what's it about? What do the words mean in context? Who is doing what to whom? spaCy is designed specifically for production use and helps you build applications that process and "understand" large volumes of text. It can be used to build information extraction or natural language understanding systems, or to pre-process text for deep learning. **Features of spaCy :**

- Tokenization Segmenting text into words, punctuations marks etc.
- Part-of-speech (POS) Tagging Assigning word types to tokens, like verb or noun.
- Dependency Parsing Assigning syntactic dependency labels, describing the relations between individual tokens, like subject or object.
- Lemmatization Assigning the base forms of words. For example, the lemma of "was" is "be", and the lemma of "rats" is "rat".
- Sentence Boundary Detection (SBD) Finding and segmenting individual sentences.
- Named Entity Recognition (NER) Labelling named "real-world" objects, like persons, companies or locations.
- Entity Linking (EL) Disambiguating textual entities to unique identifiers in a knowledge base.
- Similarity Comparing words, text spans and documents and how similar they are to each other.
- Text Classification Assigning categories or labels to a whole document, or parts of a document.
- Rule-based Matching Finding sequences of tokens based on their texts and linguistic annotations, similar to regular expressions.
- Training Updating and improving a statistical model's predictions.
- Serialization Saving objects to files or byte strings.

8 CoreNLP

CoreNLP is a framework that allows you to create a fully functional NLP pipeline with just a few lines of code. All of the major NLP processes, including as Part of Speech (POS) tagging, Named Entity Recognition (NER), Dependency Parsing, and Sentiment Analysis, are pre-built methods in the library. It also supports languages other than English, including Arabic, Chinese, German, French, and Spanish. The coreNLP pipeline is the fundamental building piece of coreNLP. The pipeline accepts an input text, processes it, and returns the results in the form of a coreDocument object. A coreNLP pipeline can be customised and tailored to your NLP project's specific requirements. This customisation is possible using the properties objects by adding, removing, or editing annotators. The pipeline itself is composed by 6 annotators.

NLTK is excellent for text pre-processing and tokenization. It also comes with a decent POS tagger. Because Standford NLP demands more resources, using Standford Core NLP for merely tokenizing/POS labelling is a bit overkill. However, one significant distinction is that NLTK does not support parsing syntactic dependencies out of the box. For that, you must provide a Grammar, which can be time-consuming if the text domain is not confined. Standford NLP, on the other hand, provides a probabilistic parser for generic text as a downloadable model that is quite accurate. It also includes NER (Named Entity Recognition) and other features.