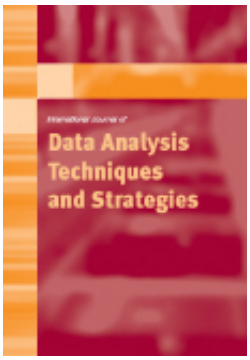


International Journal of Data Analysis Techniques and Strategies

ISSN online: 1755-8069 - ISSN print: 1755-8050
https://www.inderscience.com/ijdats

Study of Python libraries for NLP

Anand Khandare, Nipun Agarwal, Amruta Bodhankar, Ankur Kulkarni, Ishaan Mane

DOI: 10.1504/IJDATS.2023.10057032

Article History:

Received:	16 August 2022
Last revised:	25 November 2022
Accepted:	08 January 2023
Published online:	28 July 2023

Study of Python libraries for NLP

Anand Khandare, Nipun Agarwal*, Amruta Bodhankar, Ankur Kulkarni and Ishaan Mane

Computer Department,
TCET,
Mumbai, India
Email: anand.khandare@thakureducation.org
Email: nipunagarwal2001@gmail.com
Email: bodhankar.amruta@gmail.com
Email: kulankur5@gmail.com
Email: 20ishaanmane@gmail.com
*Corresponding author

Abstract: Without words, any language is incomplete, and grammar is responsible for driving those words. Interaction with computers is similar to human conversation in that it requires words and signs to communicate with one another. The traditional method of interacting with a computer by clicking and tapping on options is gradually being replaced by a more seamless approach that includes conversations. This modern mode of communication includes speaking to the computer in a more natural manner, similar to how we speak to other people. Natural language processing is the science behind how humans can interact with computers more intuitively. The goal of this domain is to figure out how to make computers understand and make sense of commonly spoken human language in addition to the usual and rationally defined set of instructions.

Keywords: artificial intelligence; machine learning; natural language processing; NLP; Python; Python libraries.

Reference to this paper should be made as follows: Khandare, A., Agarwal, N., Bodhankar, A., Kulkarni, A. and Mane, I. (2023) 'Study of Python libraries for NLP', *Int. J. Data Analysis Techniques and Strategies*, Vol. 15, Nos. 1/2, pp.116–128.

Biographical notes: Anand Khandare is an Associate Professor and Dy. HOD, Computer Engineering, Thakur College of Engineering and Technology, Mumbai with 17 years of teaching experience. He completed his PhD in Computer Science and Engineering in the domain of data clustering in machine learning from Sant Gadge Baba Amravati University. He has 50+ publications in national and international conferences and journals. He has one copyright and two patents. He guided various research and funded projects. He is also a reviewer in various journals and conferences.

Nipun Agarwal is an upcoming student of Masters of Science in Computer Science at Columbia University. He is currently pursuing his Bachelor in Computer Engineering from Thakur College of Engineering and Technology, Mumbai. He has worked in many internships as a data science intern and also has developed his skills in machine learning, natural language processing, and deep learning.

Amruta Bodhankar is currently pursuing her Bachelor of Computer Engineering from Thakur College of Engineering and Technology, Mumbai. She has knowledge about data science, machine learning, statistics and mathematics, Python. She has also done capstone projects in this field. Also, she has skills related to web development. She is well aware of HTML, CSS, JavaScript, ReactJS, PHP, and MySQL. She is an enthusiast, hard-working, inquisitive youngster with keen interests in the field of, data analytics and web development to enhance her knowledge and skills to achieve excellence.

Ankur Kulkarni is a student of Thakur College of Engineering and Technology and is an experienced professional in the fields of web development and natural language processing (NLP). He began his career as a Web Developer at PathfoundIT, where he worked on enhancing and customising the company's website, as well as managing their social media accounts. He later served as a Web Developer at TCET-CSI and TCET-ISTE, where he led teams of developers to design scalable and visually appealing websites. He has acquired expertise in NLP as a consultant on a project, leveraging his web development skills to create NLP applications that can be deployed on websites.

Ishaan Mane is a student of Thakur College of Engineering and Technology, has developed his skills in web development, framework building in Python, machine learning, natural language processing, data science, and digital marketing. He gained some experience working as a Full Stack Developer at Kansastek where he used modern technologies like AngularJS, NodeJS, ExpressJS, and Mongodb to develop applications and websites. During his time there, he also explored data science problems and learned through relevant courses. Additionally, he has some experience in digital marketing from his work with TCET-ISTE (a professional student body) where he helped them attract members through social media.

1 Introduction

Natural language processing (NLP) is a wide subject matter that falls under artificial intelligence (AI) technology. As a, end result of NLP, computer systems are capable to interpret textual content and spoken words in a similar manner to how humans do it. To get the intended outcome, NLP must be able to interpret not just letters, words but also phrases, sentences and paragraphs in their context-based entirely on syntax and semantics, grammar, etc. NLP algorithms distinguish the human language into machine-understandable chunks that can be utilised to build NLP-based software. Today, NLP is finding applications throughout the different industrial landscape, thanks to the introduction of helpful NLP libraries in python. In fact, NLP is now a crucial element of deep learning development.

The boundaries of language understanding and creation have been expanded by the introduction of transfer learning and pretrained language models in NLP. The main trend of the most recent research advances is the application of transformers and transfer learning to various downstream NLP tasks.

The most recent advancements in NLP language models appear to be driven not only by the enormous increases in computing power but also by the identification of creative strategies for model lightening while maintaining high performance. When used, NLP models like bidirectional encoder representations from transformers (BERT), GPT2,

XLNet and PaLM have very strong empirical foundations and are conceptually simple to understand. These models are made with the intention of achieving the most recent state-of-the-art outcomes on key NLP tasks, such as answering questions, recognising named entities, understanding language, reasoning, and creating new languages.

Among other NLP applications, extracting applicable information from text is fundamental for building chatbots, digital assistants because to educate the NLP algorithms a large amount of dataset is required for better overall performance however our Google Assistant and Alexa are becoming extra herbal day-by-day.

The Python libraries are a powerful way to invent AI-based and NLP-based systems in a very efficient plus pragmatic manner. This paper intends to focus on and analyse the features of the most famous Python libraries, as well as their potential for natural language processing. Throughout the paper, each of these techniques will be deeply evaluated, with examples of their use in diverse domains. These libraries are the most commonly used and respected resources for solving real-world issues and developing high-tech systems.

2 Tools in Python aiding to natural language processing

Python is a powerful interpreted language with a solid core foundation and a robust modular component that extends the language with external modules that provide new features. As a result, we now have an extensible language with tools for doing a particular operation as efficiently as feasible. Packages are frequently used to arrange modules. A package is a logical grouping of modules that all serve the same function.

2.1 Gensim

Gensim is an open-source framework for unsupervised topic modelling and natural language processing written in Python (Srinivasa-Desikan, 2018). It is a tool for extracting semantic concepts from documents that is capable of handling large text collections. As a result, it differs from other machine learning software packages that concentrate on memory processing. To improve processing speed, Gensim also provides efficient multicore implementations for several algorithms. It has more text processing capabilities than other packages such as Scikit-learn, R, and so on.

It performs a variety of complex tasks using best models and modern statistical machine learning, such as creating word or document vectors, topic identification, comparing and contrasting papers, and detecting semantic structure in plain-text materials (Ebeid and Arango, 2016).

Aside from performing complex tasks, Gensim, which is written in Python and Cython, is intended to handle large text collections via data streaming and incremental online algorithms. This distinguishes it from machine learning software packages that are only intended for in-memory processing.

2.1.1 Installation

If you use pip to install your Python libraries, you can download the Gensim library with the following command:

- $ pip install gensim.

If you use the Anaconda Python distribution, you can install the Gensim library by running the following command:

- $ conda install -c anaconda gensim

2.1.2 Key features

- Create a corpus from a given dataset.
- Create a Doc2Vec model using Gensim.
- Create a TFIDF matrix in Gensim.
- Create topic model with LDA.
- Summarise text documents.
- Create bigrams and trigrams with Gensim.
- Create a Word2Vec model using Gensim.
- Compute similarity matrices.
- Create topic model with LSI.

These are a few of the Gensim library's capabilities. This is especially useful when working on language processing.

Figure 1 GenSim implementation (see online version for colours)

```
import gensim
from gensim import corpora
from pprint import pprint
doc = [
   "Gensim is an open-source framework for unsupervised topic m
   "It performs a variety of complex tasks using best models an
   "Aside from performing complex tasks, Gensim, which is writt
]
texttokens = [[text for text in doc.split()] for doc in doc]
dictionary = corpora.Dictionary(texttokens)
pprint(dictionary.token2id)
```

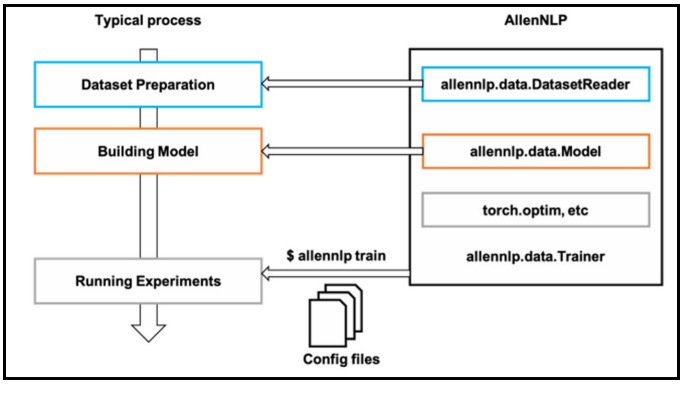
```
{'Aside': 30,
 'Cython,': 31,
 'Gensim': 0,
 'Gensim,': 32,
 'It': 16,
 'Python': 1,
 'a': 17,
 'algorithms': 33,
 'an': 2,
 'and': 3,
```

2.2 AllenNLP

AllenNLP is a PyTorch-based library developed by the Allen Institute for Artificial Intelligence. It is an open-source deep-learning library for NLP and is used for the chatbot development and analysis of text data. The ability to focus on research development is a feature of AllenNLP. Building a model with AllenNLP is significantly simpler than starting from scratch with PyTorch. Not only does it make development simple, but it also aids in experiment administration and evaluation following development. The library is available for initiatives focused on the industry as well as for study and publication. High-level parameters are frequently buried in implementation details in research codebases, which are also difficult to execute, debug, and extend, making them more likely to be rewritten (Gardner et al., 2018). AllenNLP is used for applying deep learning methods to NLP research that addresses these issues with easy-to-use command-line tools, declarative configuration-driven experiments, and modular NLP abstractions.

In order to perform various popular NLP techniques, like transformer experiments, multi-task training, vision+language tasks, fairness, and interpretability, AllenNLP provides a high-level configuration language. This enables testing on a wide range of activities solely through configuration, enabling you to focus on the crucial research topics (Wallace et al., 2019).

Figure 2 Comparison between the typical process and the process with AllenNLP (see online version for colours)



According to our own research project, we only need to implement DatasetReader and model, and then run the various experiments with config files. Basically, we need to understand the three features below to start our project with AllenNLP:

1 define our DatasetReader

2 define our model

3 setup our config files.

Briefly put, AllenNLP is:

- a command-line tool for training PyTorch models

- a library containing carefully considered abstractions encapsulating the typical data and model operations carried out in NLP research
- a set of models that have already been trained and can be used to do predictions
- a set of understandable reference implementations of popular and current NLP models
- a framework for experiments in reproducible science
- open source and community-driven.

2.3 Polyglot

Polyglot is a python NLP pipeline, developed by Rami Al-Rfou, which supports various multilingual applications and offers a wide range of analysis and broad language coverage. It consists of lots of features such as:

1 language detection (196 languages)

2 tokenisation (165 languages)

3 named entity recognition (40 languages)

4 part of speech tagging (16 languages)

5 sentiment analysis (136 languages)

6 word embeddings (137 languages)

7 morphological analysis (135 languages)

8 transliteration (69 languages).

Figure 3 Polyglot implementation (see online version for colours)

```
import polyglot
from pprint import pprint
from polyglot.text import Text
example = """Polyglot is a python NLP pipeline,
 developed by Rami Al-Rfou, which supports
 various multilingual applications and offers
  a wide range of analysis and broad language coverage."""
text = Text(example)
pprint(text.pos_tags)
```

```
('Polyglot', 'PROPN'),
('is', 'VERB'),
('a', 'DET'),
('python', 'NOUN'),
('NLP', 'NOUN'),
('pipeline', 'NOUN'),
(',', 'PUNCT'),
('developed', 'VERB'),
('by', 'ADP'),
('Rami', 'PROPN'),
('Al', 'PROPN'),
('-', 'PUNCT'),
```

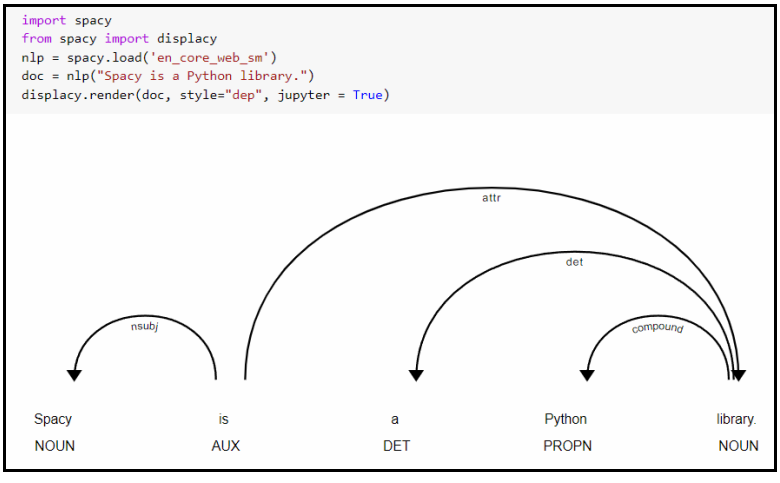
It is quick because it is built on NumPy. It distinguishes itself from the competition owing to its extensive selection of specialised commands. It can be used for languages that do not support Spacy and is analogous to Spacy.

2.4 spaCy

spaCy is a Python library for advanced natural language processing that is free and open source. It will be used to develop text pre-processing for deep learning, natural language understanding systems, and information extraction systems. It supports named entity recognition, dependency parsing, and parts-of-speech tagging in convolutional neural networks using a deep learning workflow (Neumann et al., 2019).

Ines Montani and Matthew Honnibal are the main developers and maintainers of Spacy. Scipy is a Python and Cython program. We have access to more than 60 languages, including English, Hindi, Spanish, German, French, and Dutch, for text processing, concentrate primarily on industrial goals.

Figure 4 Spacy implementation (see online version for colours)



2.4.1 How to setup spaCy

pip, a Python package manager, can be used to install spaCy. To avoid relying on system-wide packages, you can use a virtual environment.

Create a new virtual environment:

- $ python3 -m venv env

Activate this virtual environment and install spaCy:

- $ source ./env/bin/activate
- $ pip install spacy

2.4.2 Key features

1 More than 66 languages are supported.

2 There are 76 trained pipelines for 23 languages.

3 Pretrained transformers such as BERT are used for multi-task learning.

4 Vectors of pre-trained words.

5 Cutting-edge speed.

6 Manufacturing-ready training system.

7 Tokenisation for linguistic reasons.

8 Custom components and attributes are easily extensible.

9 Custom model support in PyTorch, TensorFlow, and other frameworks.

10 Syntax and NER visualisers are built in.

11 Model packaging, deployment, and workflow management are all simplified.

12 Accuracy that has been rigorously tested.

spaCy is an open-source library software for advanced natural language processing (NLP) that is written in Python and Cython and distributed under the MIT license. spaCy is a modern and decisive NLP framework that is the classic source for performing NLP with Python and has excellent features such as speed, accuracy, and extensibility. It quickly became a critical component of the NLP production pipeline.

2.5 Scikit

Scikit-learn is one of the most helpful and a key python library that is structured for machine learning and statistical modelling. It is open-source and commercially accessible software. It is successful in performing several statistical, data mining, and data evaluation operations like- Classification, Clustering, and Regression. Scikit-learn is simple in design, efficient and is effortlessly approachable via non-experts. It first emerged from David Cournapeau as a Google summer time season code venture in 2007. Pedregosa et al. (2011), from FIRCA, took this mission to the next feasible diploma and made the major launch in 2010.

The facets of the package Scikit-learn are:

- *Supervised learning algorithms* – nearly all supervised mastering algorithms like, linear regression, decision tree, and support vector machine (SVM) belong to Scikit-learn. These algorithms help to estimate the results for unexpected data.
- *Unsupervised learning algorithms* – this library additionally includes very popular unsupervised studying algorithms of clustering, principal component analysis (PCA), factor analysis which helps in performing more complex processing tasks. It approves the model to work on its own, without any supervision and discover facts and data.
- *Cross validation* – it is used to verify the accuracy of supervised models on unseen statistics and helps in estimating the overall performance of models.

- *Feature extraction* – it is used to extract elements from the data consisting textual content and images in codecs supported via machine-learning. It includes functions like-DictVectorizer(), feature_name, CountVectorizer() and many more.
- *Feature selection* – this module is used to apprehend beneficial attributes to create supervised models. Feature selection strategies are used for simplification of models, improve data compatibility, and making the records simpler for the users to interpret.
- *Dimensionality reduction* – this is used to minimise the attributes in the statistics for characteristic selection, summarisation and visualisation. The converted low-dimensional space retains important elements of the authentic data and is handy to analyse and procedure using computing device learning techniques. This can be finished with the usage of the PCA features like PCA (n-components,svd_solver), pca.fit().

Figure 5 Basic program for linear regression using Scikit learn (see online version for colours)

```
import numpy as np
from sklearn.linear_model import LinearRegression
X = np.array([[1,1],[1,2],[2,2],[2,3]])
y = np.dot(X, np.array([1,2])) + 3
#Creating linear regression object
regr = LinearRegression(
fit_intercept = True, normalize = True, copy_X = True, n_jobs = 2).fit(X,y)
#Using predict() method to predict using this linear model
regr.predict(np.array([[3,5]]))
#To get the coefficient of determination of the prediction we can use Score()
#regr.score(X,y)
#To estimate the coefficients by using attribute named 'coef'
#regr.coef_
```

```
array([16.])
```

```
[2] import numpy as np
from sklearn.linear_model import LinearRegression
X = np.array([[1,1],[1,2],[2,2],[2,3]])
y = np.dot(X, np.array([1,2])) + 3
#Creating linear regression object
regr = LinearRegression(
fit_intercept = True, normalize = True, copy_X = True, n_jobs = 2).fit(X,y)
#Using predict() method to predict using this linear model
#regr.predict(np.array([[3,5]]))
#To get the coefficient of determination of the prediction we can use Score()
#regr.score(X,y)
#To estimate the coefficients by using attribute named 'coef'
regr.coef_
```

```
1.0
```

Scikit-learn is a very integral library for natural language processing. Whenever we work on any NLP-associated problem, we method a lot of textual data. The textual statistics after processing desires to be fed into the model. Since the model would not accept textual facts and only knows numbers, these records need to be vectorised. Vectorisation is a procedure of changing the text records into a machine-readable form. The phrases are represented as vectors. Scikit-learn is very environment-friendly for this process. Scikit-learn makes use of a method of The Bag of Words(BoW) model for changing text into machine readable numbers as we cannot bypass textual content without delay to educate our models in NLP (Studytonight, 2021). The BoW model is very easy as it discards all the facts and order of the textual content and just considers the occurrences of the word, in brief, it transforms a sentence or a paragraph into a collection of meaningless words. It converts the files to a fixed-length vector of numbers. CountVectorizer

tokeniser (tokenisation skill breaking down a sentence or paragraph or any textual content into words) the text along with performing very fundamental preprocessing like doing away with the punctuation marks, converting all the phrases to lowercase, etc. A dictionary of recognised words develops and is later utilised to encode incoming text.

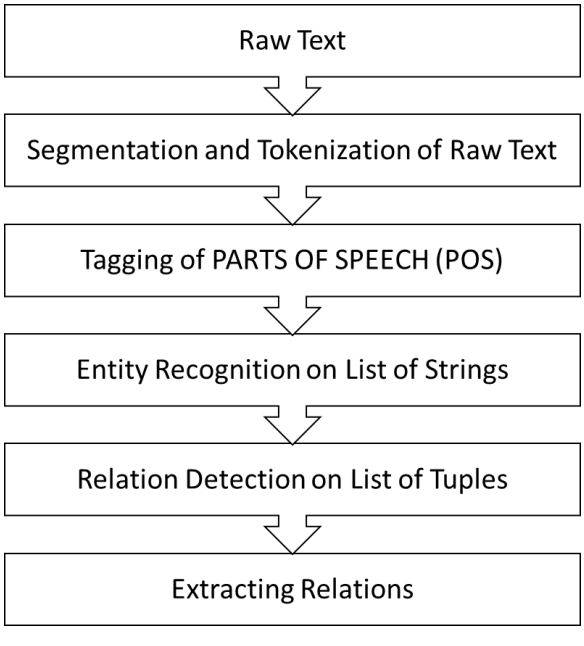
Scikit-learn uses a widespread and big variety of computing devices to gain knowledge of algorithms. It helps natural language processing in python and approves the construction of a variety of computing devices studying fashions for predicting and interpreting abstract, unorganised and sudden data. It is a useful tool to procedure giant to small scale data. Both supervised and unsupervised learning methods can be adopted by using nicely suitable and task-based interfaces. This permits the assessment of methods and strategies for a given application.

2.6 CoreNLP

CoreNLP is a toolkit that allows you to create a fully functional NLP pipeline with just a few lines of code. All of the major NLP procedures, such as part of speech (POS) tagging, named entity recognition (NER), dependency parsing, and sentiment analysis, are pre-built methods in the library (Manning et al., 2014).

Using Stanford's CoreNLP to analyse text data makes text data analysis simple and efficient. CoreNLP can extract all kinds of text properties, such as named-entity recognition or part-of-speech tagging, with just a few lines of code (Kaur and Agrawal, 2018).

Figure 6 Information extraction using CoreNLP



CoreNLP is written in Java and requires Java to be installed on your device; however, it provides programming interfaces for a number of popular programming languages, including Python.

It also supports languages other than English, including Arabic, Chinese, German, French and Spanish.

2.7 NLTK

Processing and understanding human language data are crucial for any interactive AI to function properly, provide more value and solve problems. NLP is a domain that focuses on understanding, processing and implementing human language data effectively to solve real world problems by ensuring that the computer-human interaction takes place smoothly (qblocks.cloud, 2021).

The Natural Language Toolkit (NLTK) was created in 2001 at the University of Pennsylvania in connection with a computational linguistics course. Assignments, demonstrations, and projects were the three pedagogical uses in mind when it was created.

NLTK is a Python package which is predominantly used for NLP. NLTK preprocesses unstructured data containing human language references using computational linguistics, NLP data types and animated algorithms. NLTK also provides problem sets and tutorials to make the user familiar with this python library. NLTK is very beneficial for the students or programmers who are learning NLP or conducting research on the same topic.

Run the following instructions in your terminal to install NLTK:

- sudo pip install nltk

Then, on your terminal, type python to launch the Python shell and run the following instructions:

- import nltk
- nltk.download('all')

Since NLTK is completely written in python, it has the following features:

- easier and convenient to learn
- exceptional at string-handling
- well-defined syntax
- data encapsulation is possible and data can be reused multiple times.

NLTK implementation:

- chatbots
- machine translation
- speech recognition
- text summarisation
- recommendation engine
- sentiment analysis for customer reviews.

NLTK has been effectively utilised as a teaching tool, as a tool for individual study, and as a platform for prototyping and developing research systems. NLTK offers a simple, versatile, and consistent framework for assignments, projects, and class presentations. It's well-documented, easy to understand and utilise. The most popular tool for teaching NLP is NLTK. It's also commonly used as a prototype and research tool (Singh et al., 2019).

2.8 TextBlob

TextBlob is a Python open-source library with a simple API for accessing its methods and performing basic NLP tasks. It offers API for tagging parts of speech, noun phrase extraction, sentiment analysis, translation, and classification. TextBlob strings are similar to Python strings, which is a major bonus. The fact that the library is compatible with Python string makes it much easier to implement a wide range of Python applications. The TextBlob is already easier to use. Textblob is popular among data scientists for prototyping because of its lightweight nature.

Some challenges are also faced while using TextBlob by people while performing sentiment analysis. TextBlob only describes the polarity and subjectivity. Subjective sentences usually refer to personal opinion, judgement, or emotion, whereas objective refers to factual information. Subjectivity is also a float that lies in the range of [0, 1]. Polarity is a float that lies in the range of [–1, 1] where 1 means a positive statement and –1 means a negative statement (Zahidi et al., 2021). TextBlob may not give an accurate analysis of the emojis. With the wide usage of different languages, sentimental analysis using TextBlob may become difficult to analyse emotions (Gujjar and Kumar, 2021).

3 Conclusions

NLP ideas are now very simple to implement thanks to Python's libraries, modules, and frameworks. Python NLP libraries have become the most popular language for developing NLP algorithms. Understanding Python is critical for developing conceptual knowledge and specialising in NLP. Python libraries are essential in applications such as tokenisation, normalisation, data visualisation, image and data processing, and others. This paper adequately discussed and emphasised the critical and required Python programming libraries involved in researching the vast topic of NLP.

References

Ebeid, I. and Arango, J. (2016) *Mallet vs GenSim: Topic Modeling Evaluation Report*, DOI: 10.13140/RG.2.2.19179.39205/1.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N.F., Peters, M., Schmitz, M. and Zettlemoyer, L. (2018) 'AllenNLP: a deep semantic natural language processing platform', *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, Association for Computational Linguistics, Melbourne, Australia, pp.1–6.

Gujjar, J.P. and Kumar, H.P. (2021) 'Sentiment analysis: textblob for decision making', *Int. J. Sci. Res. Eng. Trends*, Vol. 7, No. 2, pp.1097–1099.

Kaur, S. and Agrawal, R. (2018) 'A detailed analysis of core NLP for information extraction', *International Journal of Machine Learning and Networked Collaborative Engineering*, Vol. 1, No. 1, pp.33–47.

Manning, C.D. et al. (2014) 'The Stanford CoreNLP natural language processing toolkit', *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Neumann, M., King, D., Beltagy, I. and Ammar, W. (2019) 'ScispaCy: fast and robust models for biomedical natural language processing', *Proceedings of the 18th BioNLP Workshop and Shared Task*, Association for Computational Linguistics, Florence, Italy, pp.319–327.

Pedregosa, F. et al. (2011) 'Scikit-learn: machine learning in Python', *The Journal of Machine Learning Research*, Vol. 12, pp.2825–2830.

qblocks.cloud (2021) *Best Python Libraries for NLP in 2021 and their Use Cases*.

Singh, A., Ramasubramanian, K. and Shivam, S. (2019) *Building an Enterprise Chatbot: Work with Protected Enterprise Data Using Open Source Frameworks*, DOI: 10.1007/978-1-4842-5034-1.

Srinivasa-Desikan, B. (2018) *Natural Language Processing and Computational Linguistics*, June, Packt Publishing Ltd., Birmingham, UK ISBN: 9781788838535.

Studytonight (2021) *Scikit-learn CountVectorizer in NLP*.

Wallace, E., Tuyls, J., Wang, J., Subramanian, S., Gardner, M. and Singh, S. (2019) *AllenNLP Interpret: A Framework for Explaining Predictions of NLP Models*, arXiv preprint arXiv:1909.09251.

Zahidi, Y., El Younoussi, Y. and Al-Amrani, Y. (2021) 'Different valuable tools for Arabic sentiment analysis: a comparative evaluation', *International Journal of Electrical & Computer Engineering*, Vol. 11, No. 1, p.755, ISSN: 2088-8708.