AJSIHD
GLOBAL RESEARCH NETWORK

AMERICAN Journal of Science on Integration and Human Development
Volume 2, Issue 6, 2024 ISSN (E): 2993-2750

Natural Language Processing for Text in Python

Karshiyev Abduvali Berkinovich
Professor of Samarkand branch of Tashkent University of information technologies named after Muhammad al-Khwarizmi, Samarkand, Uzbekistan

Mamaraimov Mirjalol Shakarboyivich
A Master of Samarkand branch of Tashkent University of information technologies named after Muhammad al-Khwarizmi, Samarkand, Uzbekistan

Abstract: This article covers the basic principles of natural language processing (NLP) technologies and how they can be used in the Python programming language. Processes such as tokenization, stemming, and lemmatization are consistently described. Accordingly, methods of text tokenization using spaCy library tools and using lemma, POS, tag, stop attributes created through the pipeline process are provided.

Keywords: pipeline, NLP, spaCy, Python, part-of-speech, lemmatization, token, parsing, "stop" words.

I. INTRODUCTION

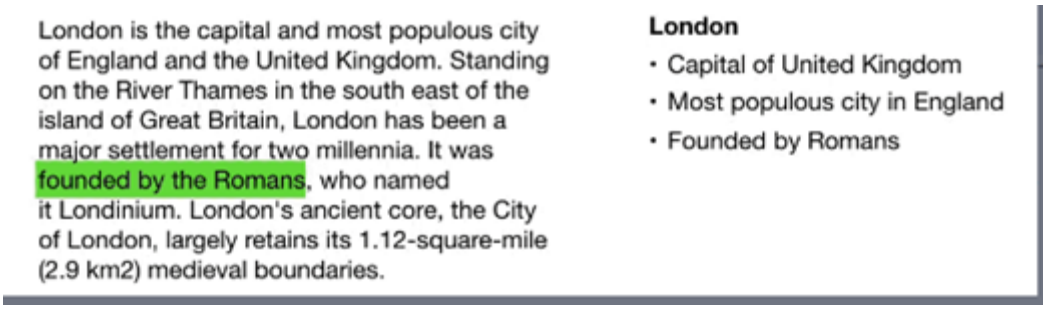
"Concept of development of the Uzbek language and improvement of the language policy in 2020-2030" approved by the decree of the President of the Republic of Uzbekistan dated October 20, 2020 "On measures to further develop the Uzbek language and improve the language policy in our country" ” to create an electronic national corpus of the Uzbek language that includes all scientific, theoretical and practical information about the Uzbek language in the priority direction of ensuring the active integration of the state language into modern information technologies and communications, and The task of popularizing the Uzbek language in the world information network, and ensuring that it occupies a worthy place in it, has given great responsibility to specialists in the field [1]. Natural language processing (NLP) is one of the most important and rapidly developing areas of modern computer science. This technology enables computers and software to understand, analyze and process human language. We make significant progress in many areas by solving natural language problems, understanding text, and extracting information from it. For example, practices such as automatic text translation, data analysis, customer service with the help of artificial intelligence are implemented on the basis of NLP technologies.

II. MAIN BODY

First, we will consider the principles of Python's NLP (Natural language processing) system, and then we will try to teach a computer to understand human language. We will create a pipeline for text analysis and support it with the spaCy library.

Computers typically work with structured data, such as tables in a database. However, people communicate not with tables, but with words. This is too complicated for computers.

Most of the information in life is not structured. Examples of this are texts in natural languages. Is it possible to solve the problem of extracting important information from them with the help of a computer? Or, in other words, the question arises whether it is possible to teach a computer to extract important information from texts. A special branch of artificial intelligence known as "natural language processing (NLP)" deals with this problem. In this article, we will learn how it works. As an example, we will create a program to extract information from unstructured text.



Can a computer understand language?

Since the dawn of the computer age, manufacturers have been trying to teach computers simple languages, such as English. Humanity has accumulated a huge amount of written information over thousands of years. It is clear to everyone that it would be a great result if a computer could read and analyze these data [3].

Although computers are not yet able to fully understand human language, many problems in this field have been solved in Python through the NLP system.

Extract meaning:

The process of reading and understanding English text is considered complex. Also, in many cases, people do not pay attention to the logic and consistency of the story.

A pipeline is a method developed for the sequential execution of operations or functions that process data.

The goal of this approach is to solve the problem in parts.

A turn-by-turn NLP pipeline:

We will analyze the following text in turn.

London is the capital and most populous city of England and the United Kingdom. Standing on the River Thames in the southeast of the island of Great Britain, London has been a major settlement for two millennia. It was founded by the Romans, who named it Londinium.

This paragraph contains some useful facts. We intend for the computer to understand the following facts: London is a city, it is located in England, it was founded by the Romans, etc. But first we need to teach him the most basic concepts of the written language.

Step 1: Split the sentences

he first step in this process is to divide the text into separate sentences. As a result, the following is obtained:

1. London is the capital and most populous city of England and the United Kingdom.
2. Standing on the River Thames in the southeast of the island of Great Britain, London has been a major settlement for two millennia.
3. It was founded by the Romans, who named it Londinium.

Each isolated sentence can be considered as an independent thought or idea. It's easier to train a program to understand a single sentence than an entire paragraph. Using certain punctuation

marks, text can be broken up into chunks. Modern NLP pipelines have more complex methods that allow working with unformatted fragments.

Step 2. Tokenization (split words):

Now we can review the received offers one by one. Let's start with the first one:

London is the capital and most populous city of England and the United Kingdom.

The next step in the pipeline is the separation of individual words or tokens - tokenization. At this stage, the result will look like this:

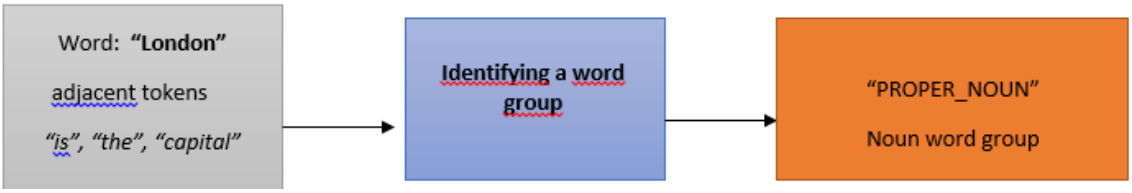
"London", "is", "the", "capital", "and", "most", "populous", "city", "of", "England", "and", "the", "United " ", "Kingdom", "."

It will not be difficult to carry out this process in English. Whenever a space character encounters a space character, the preceding fragment is truncated. Punctuation marks are also tokens because they can have important meanings.

Step 3. Identify word groups.

Now we will see the process of determining which word class (noun, verb, adjective or other) each token belongs to. To understand the general meaning of a word, it is necessary to know its role in the sentence [2, p. 177].

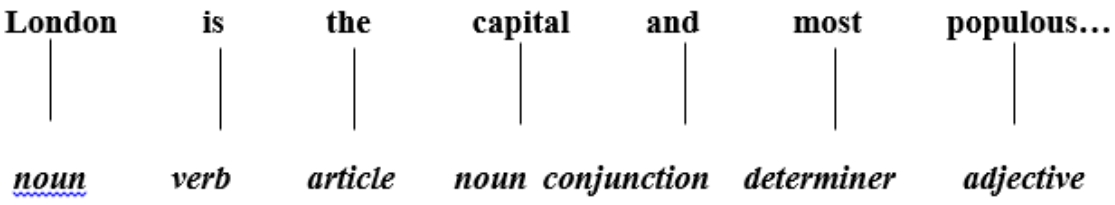
At this stage, we analyze each word together with its neighboring words using a previously prepared classification model:



The classification model was trained on the basis of a collection of texts consisting of one million English sentences, in which each word is associated with information indicating which word group it belongs to. This trained model allows you to determine which word group the words in the input text belong to.

It should be remembered that this analysis is based on statistical data, and the model does not understand the true meaning of the words. It simply knows how to identify vocabulary information based on similar sentence structures and previously learned tokens.

After the process is completed, the following result will be generated:



Based on this information, the analysis process can be started to determine the meaning of the words. For example, if we analyze the words "London" and "capital" in the noun group, it can be assumed that the sentence has a meaning about London.

Step 4. Lemmatization

In English and other languages, the same word can have different forms. Let's consider the following examples:

I had a **pony** I had two **ponies.**

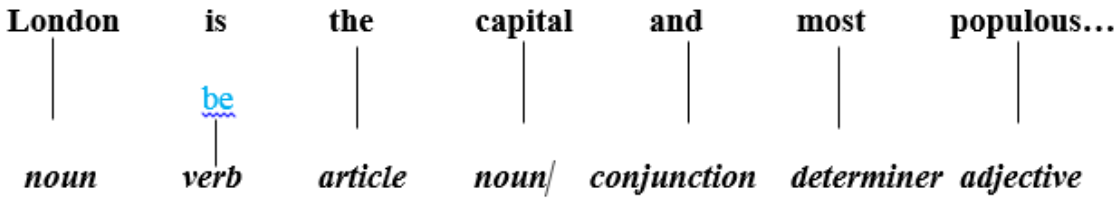
Both sentences contain the noun "pony" and differ in the additions at the end: "pony" and "ponies". When text is processed by a computer, the computer needs to know the base form for both of these word forms. Otherwise, **"pony"** and **"ponies"** would be considered completely different words.

In NLP, this process is called lemmatization, and it consists in finding the main form (lemma) of each word in the sentence.

The same situation applies to the verb group. Words belonging to the verb group can be put into the indefinite form. As a result, the sentence **"I had two ponies"** becomes the sentence **"I [have] two [pony]".**

Lemmatization is usually performed by searching word forms from the table [2, pp. 31-32]. Also, some custom rules for parsing words can be added by the user.

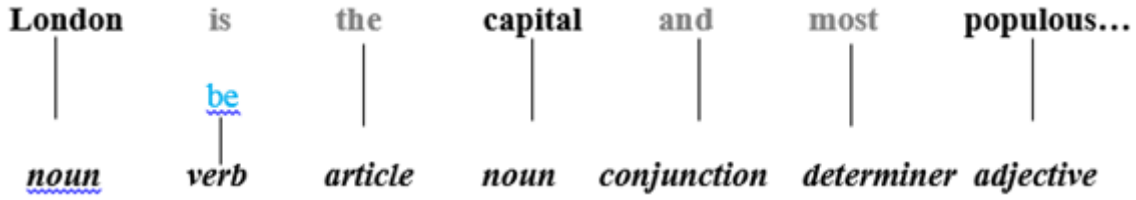
After the lemmatization process, the sentence in question becomes:



The only change is the change of "is" to "be".

Step 5. Identifying the words "stop".

The importance level of each word in the sentence is determined. There are many auxiliary words in English, for example: «and», «the», «a». When analyzing the text statistically, these tokens create a lot of inconvenience, because they appear more often in the text than other tokens. Some NLP pipelines mark them as "stop" words and they are not considered in the quantitative analysis process. When the stop words are identified (the text is colored), the sentence in question looks like this:

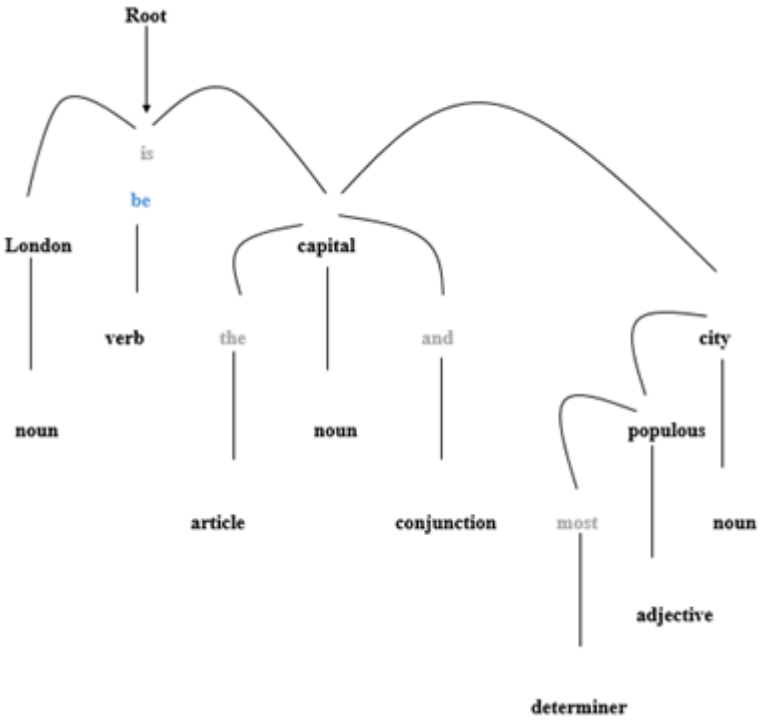


To determine stop words, usually ready-made tables are used. However, there is no single standard list that fits every situation. The list of "stop" words can be different in different NLP problems. For example, when developing a search engine for rock bands, it is best not to use the article "the" as a stop word [5], as it appears in the names of many bands, and even the popular 80's group called "The The!".

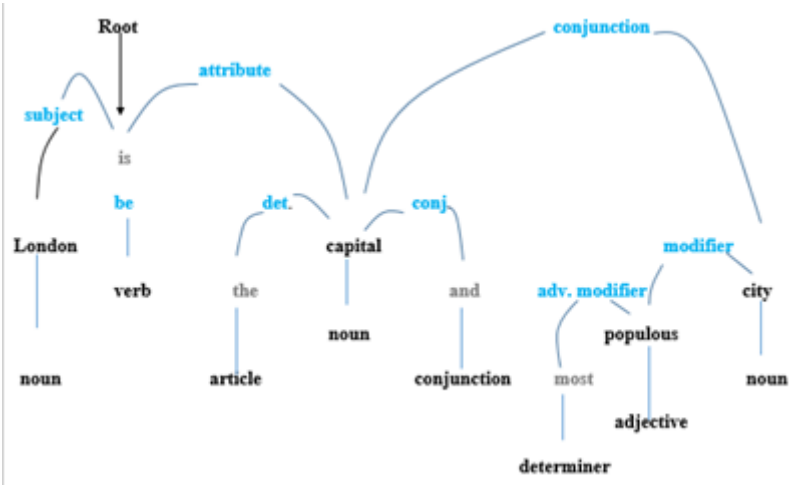
Step 6: Link Parsing.

Now let's look at the process of determining the relationship between the words in the sentence, that is, connections. This process is called link parsing. The result of this step is the construction of a tree of connections, in which each token has a single root [3, pp. 407-412]. As a root, a verb can be the main word of a word group.

In the first approximation, the following scheme is formed:



At the next stage, it is necessary to determine not only the root of the word, but also the type of connection between the two words:



The main subject in this parsing tree is the noun phrase "London". Between u and the word **"capital"** there is a link **"be"**. With such an analysis, it can be determined that London is the capital. If parsing trees are built for the following sentences and the analysis of its branches is continued, it turns out that London is the capital of the United Kingdom.

III. CONCLUSION

This article covers a wide range of NLP tools that can be used in many fields. Because human and machine learning processes are different, computers use several natural language processing methods. Python libraries like SpaCy, NLTK helped to facilitate the workflow. Today, SpaCy is a reliable and popular Python library, widely used in NLP applications due to its speed, ease of use, accuracy, and flexibility. Every minute, mainly textual information is generated in various formats, for example: SMS messages, comments, e-mail messages, etc. The pipeline process was implemented using the SpaCy library through the methods described in this article. Using the pipeline process, text tokenization, parsing, lemmatization, tagging, and stop word attribute values were generated. At this stage, information is obtained from unstructured text, identification of "unnamed objects", analysis of word units in the text is carried out. The use of

NLP technologies significantly improves the modeling of business processes in the development of information systems and, at the same time, increases work efficiency.

REFERENCES

1. https://lex.uz/docs/-5058351
2. Jurafsky, D., & Martin, J. H. (2020). *Speech and Language Processing* (3rd ed.). Prentice Hall.
3. Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media
4. Goldberg, Y. (2017). *Neural Network Methods for Natural Language*
5. *Processing*. Morgan & Claypool Publishers.
6. Pennington, J., Socher, R., & Manning, C. D. (2014). *Glove: Global Vectors for Word Representation*. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1532-1543.
7. McKinney, W. (2010). Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, 51-56.
8. Spacy Documentation (n.d.). *spaCy 101: Everything you need to know*. Retrieved from https://spacy.io/usage/spacy-101
9. Sarkar, D. (2019). *Text Analytics with Python: A Practitioner's Guide to Natural Language Processing*. Apress.
10. Yoo, J. S. (2021). *Machine Learning for Text: NLP in Python with scikit-learn*. Independently published.