# Implementation of Conversion from Natural Language Texts into Structures Texts Using Natural Language Tool Kit with Improved Version

**Jessia Esther Leena J[1]**

**Department of Computer Science, St Joseph's University, Bangalore, India, jessiabritto@gmail.com**

**B.G.Prasanthi[2]**

**Department of Computer Science, St Joseph's University, Bangalore, India, nitai2009@gmail.com**

**Abstract- Natural Language Processing is a powerful tool that has applications in various fields such as machine learning, businesses, data analysis, language translations and so on. Natural language processing is a task that humans are able to perform effortlessly. Yet, providing computers with this ability remains to be a daunting venture. There are numerous algorithms that help in breaking down language understood by humans into language understood by computers. These algorithms are available in libraries of programming languages such as Python. One such library is called Natural Language Toolkit (NLTK). In this paper, we learn different ways to turn natural language texts into structured texts, understandable by computers. We also see the drawbacks in some of these algorithms, comparisons of time complexities and explore different ways to increase the accuracy of these algorithms.**

***Keywords--- Lemmatisation, Natural Language Processing, Natural Language Understanding, Part of Speech tagging, Stemming, Tokenisation***

## I. LITERATURE SURVEY

NLP is a branch of computer science that gives computers the ability to process human languages. Due to the various ambiguities of human languages, developing softwares that are able to perform natural language understanding is a hard task. NLP tasks help in breaking down text into format easily processable by computers[1].

There are various NLP algorithms like part of speech tagging, parsing, named entity recognition etc. Parsing is the task of diving a sentence into its grammatical components. Named entity recognition deals with identifying named entities like people and places[2].

Multimedia Tools and Applications - *Natural language processing: State of the art, current trends and challenges.*

"At lexical level, semantic representation can also be replaced by assigning the correct POS tag which improves the understanding of the intended meaning of a sentence. It is used for cleaning and feature extraction using various techniques such as removal of stop words, stemming, lemmatization etc"[3].

Natural Language Processing Journal - *A survey on Named Entity Recognition—Datasets, tools, and methodologies.*

"The well-known platform for making applications in Python that use human linguistic data is called NLTK (Bird et al., 2009). In addition to a compilation of text processing libraries for parsing, categorization, tagging, stemming, tokenization, and semantic reasoning, named entity recognition and an active discussion forum, it is frequently employed when performing research and training students"[4].

## II. INTRODUCTION

Natural Language Processing (NLP) is a field of Artificial Intelligence that gives computers the ability to understand and generate natural language, the way humans do. This field combines linguistics with machine learning algorithms, giving computers the ability to convert natural language used by humans into structured language used by computers, and vice versa. There are two subsets of Natural Language Processing: Natural Language Understanding (NLU) and Natural Language Generation (NLG). NLU is used to convert unstructured data into structured ones, easily understandable by computers. For instance, computers use NLU for speech recognition. They receive input from the 'speaker' which is the unstructured data. It is then converted into structured data and in this way the computers are able to understand information from speech. NLG, on the other hand, is used to convert structured data into human language text[5]. Text summarisation, for example, uses NLG to produce a summary that maintains the integrity of the original data while being comprehensible by humans.

## III. NATURAL LANGUAGE UNDERSTANDING

The first step towards NLU is getting the input, either as a written text or spoken text, which is then converted into written text. Once we have our input text, we perform various algorithms on it to "break it down" into structured data.

 DOI : 10.35291/2454-9150.2024.0004 

These algorithms include Tokenisation, Stemming, Lemmatisation, Part of Speech Tagging, Named Entity Recognition, Sentence Parsing and so on.

Tokenisation:

In this process, we take a sentence and break it down into its constituent words or 'tokens.' This process of fractionating data makes working with them a much simpler task, than if we were to work with them as a whole. Tokenisation is also a great way to make sure each and every 'tokens' are taken into consideration during language processing.

Stemming:

Once we have our tokens, we further categorise them based on their 'stem words.' A stem word is the form of a word before any prefixes or suffixes are added to it. Stemming is the process of removing these affixes, as well as normalising the tense, further simplifying the data.

Lemmatisation:

Another way of categorising tokens is by determining the lemma of the tokens. This process is called Lemmatisation. In the process, the lemma of a word is identified by analysing the intended meaning of the word. Unlike stemming, lemmatisation also considers the context in which the word is used.

Part of Speech tagging:

Part of speech tagging deals with identifying the intended grammatical use for a particular word in the given context. Same words can be used as different word forms, like noun and verb, and it helps to identify what form the particular word is used as, for the computer to be able to recognise the intended meaning.

Named Entity Recognition:

Identifying various entities from a text, that may be a single word or a series of words, whose value will represent that same item, and classifying these entities based on predefined categories is called Named Entity Recognition. This process helps in identifying the key information from the given text, such as location, person, events etc, and hence speeds up the process of understanding the input.

## IV. IMPLEMENTATION USING NLTK

The Natural Language Tool Kit (NLTK) is a Python library for natural language processing on the English language[6][7]. The various algorithms of NLU, such as tokenisation, classification, stemming, lemmatisation, named entity recognition etc, can be executed using NLTK. Although it is a widely used library for natural language processing in research and teaching fields, there are quite some drawbacks in its execution. A few such errors are discussed below.

Stemming:

Proper nouns are a type of noun that always refer to a specific entity. Since stemming is about removing any prefixes or suffixes to normalize words, it should not affect proper nouns, whose root words are not necessary for comprehending the meaning of the text. When stemming is implemented using NLTK, proper nouns are incorrectly stemmed down to another word. For instance, the name "Bunny" is stemmed to "bunni." Such incorrect stemming of proper nouns will result in inaccurate results.

Stemmed words are sometimes misleading. In some cases, removing of prefixes or suffixes might lead to incorrect root words. In the example of the word "several", which means various or multiple, the given word is stemmed down to the word "sever", the meaning of which is to detach or cut off.

Two completely different words are stemmed down to the same stem word. The words "universal" and "university" are both stemmed down to "univers" even though the vast difference in their respective meanings is axiomatic[8].

Two similar words are stemmed down to an incorrect stem word. In the case of the words "paternal" and "paternity", both of whose stem word is "pater" (the Latin word for 'father'), incorrect stemming takes place where the stem word becomes "patern."

The stemming of words of superlative degree to their stem word is absent in some cases. The word "biggest" is not stemmed down to "big" but rather remains as "biggest."

In all these instances, the Snowball stemmer is used for the stemming process. Similar results are seen with Porter stemmer.

**FIGURE 1: INCORRECT RESULTS IN STEMMED WORDS**

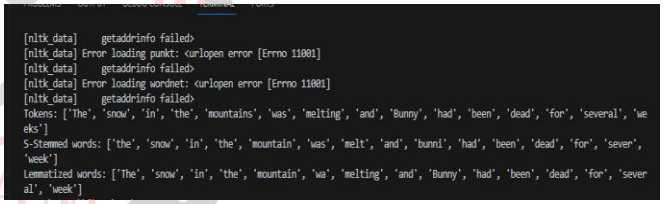


Figure 1: In this example, the first line of Donna Tartt's novel, The Secret History is used as the input text. The word "melting" is stemmed down to "melt" after removing the suffix. The name "Bunny" is stemmed down incorrectly, along with the word "several." Note that the proper noun is not changed in case of lemmatisation.

**FIGURE 2: INCORRECT RESULT IN STEMMED WORDS**

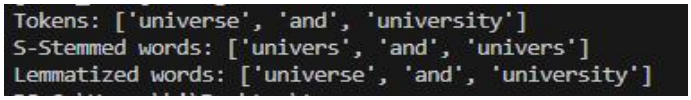


Figure 2: Input words "universe" and "university" both stem down to "universe." In this example, we could see how to different could be incorrectly stemmed down to the same word. Note that these words are not broken down to the same word in case of lemmatisation.

**FIGURE 3: INCORRECT RESULT IN STEMMED WORDS**

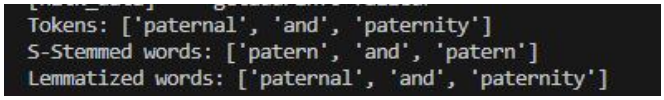


Figure 3: Similar words "paternal" and "paternity" are both affected by faulty stemming. Note that this is not case when lemmatisation is used.

 DOI : 10.35291/2454-9150.2024.0004 

Lemmatisation:

Erroneous lemmatisation is seen when auxiliary verbs are given as the input. The word “was” is lemmatised to “wa” and the word “has” is lemmatised to “ha.”

Lemmatisation is missing for words of different verb forms. For example, the words “walking” and “walked” are not lemmatised to “walk”, which the dictionary word for both these words.

Lemmatisation results are quite similar to the results obtained after removing ‘stopwords’ from the input. Stopwords are commonly used words that are omitted from the text, as they make little to no difference in giving the text its meaning. There are 179 such words in NLTK.

**FIGURE 4: INCORRECT RESULT IN LEMMATIZED WORDS**

```
Tokens: ['is', 'are', 'was', 'were', 'have', 'has']
S-Stemmed words: ['is', 'are', 'was', 'were', 'have', 'has']
Lemmatized words: ['is', 'are', 'wa', 'were', 'have', 'ha']
```

Figure 4: Auxiliary verbs are lemmatised incorrectly. These words are not required to be broken down for processing, other than normalising their tense. Note that these words remain unchanged when they undergo stemming.

## V. COMPARISON OF THE TIME COMPLEXITY OF NLTK’S TOKENISATION

The execution time of tokenisation using NLTK is compared with that of Regular Expressions. In this experiment, a tweets dataset of varying number of rows are given as the input in two python codes, one written using regular expression and the other using NLTK. To normalise the input, the number of words in each row in one case is equal to the number of words in its respective row of the other case. From this experiment’s observation, we are able to reckon that, for arriving at the same result of tokenising the tweets, the execution using RE is significantly faster than the execution using NLTK. The following table shows comparison between the results of the two python language codes.

**TABLE 1. RESULTS OF RUNTIME USING RE AND NLTK**

| No of rows | Runtime using RE (seconds) | Runtime using NLTK (seconds) |
|---|---|---|
| 2000 | 0.032238 | 0.56304 |
| 4000 | 0.096786 | 0.893805 |
| 6000 | 0.111997 | 1.296149 |
| 8000 | 0.138619 | 1.623223 |
| 10000 | 0.180895 | 2.325591 |
| 12000 | 0.217403 | 2.585193 |
| 14000 | 0.210912 | 2.806966 |
| 16000 | 0.211583 | 3.626791 |
| 18000 | 0.285745 | 4.060769 |
| 20000 | 0.321879 | 4.113108 |
| 22000 | 0.299892 | 4.602840 |
| 24000 | 0.400953 | 5.193449 |
| 26000 | 0.411176 | 5.768627 |
| 28000 | 0.401469 | 5.716132 |
| 30000 | 0.441552 | 6.952494 |
| 32000 | 0.466847 | 6.791107 |

Table 1: The above table compares the runtime results obtained to perform tokenisation using Regular Expressions and NLTK. The number of rows given as input is increased linearly by a value of 2000. For accuracy, the same 2000 rows were incremented so that the runtime difference would not be affected by the number of words in a row.

**FIGURE 5: SAMPLE OF THE TWEETS**


Figure 5: A sample of the input text used is shown. It contains a dataset of raw tweets on footballer Lionel Messi, obtained from Kaggle. For precision, the same number of tweets were used as input for both python codes. Tokenisation time is also affected by the number of words in a sentence. [9]

**FIGURE 6: RESULT USING RE**

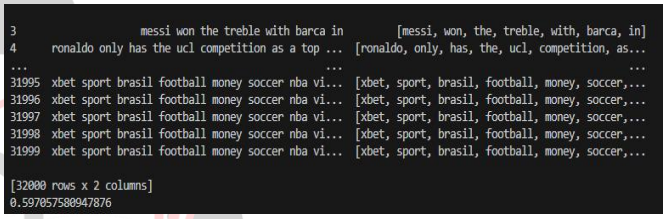


Figure 6: The result obtained after performing tokenisation using Regular Expressions. The above image shows the time taken to tokenize an input of 32000 rows of tweets. The tokens are returned as tuples.

**FIGURE 7: RESULT USING NLTK**


Figure 7: The result obtained after performing tokenisation using NLTK. The above image shows the time taken to tokenize an input of 32000 rows of tweets. The tokens are returned as tuples.

 DOI : 10.35291/2454-9150.2024.0004 

## VI. GRAPHICAL REPRESENTATION

**Figure 8: Scatter Plot Of Runtime Versus Number Of Rows For Re**

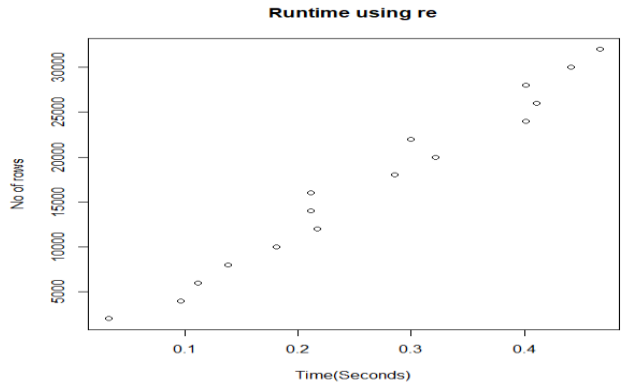


Figure 8: The above image shows the number of rows versus runtime graph for tokenisation using Regular Expressions. The graph shows a positive linear relationship.

**Figure 9: Scatter Plot Of Runtime Versus Number Of Rows For Nltk**

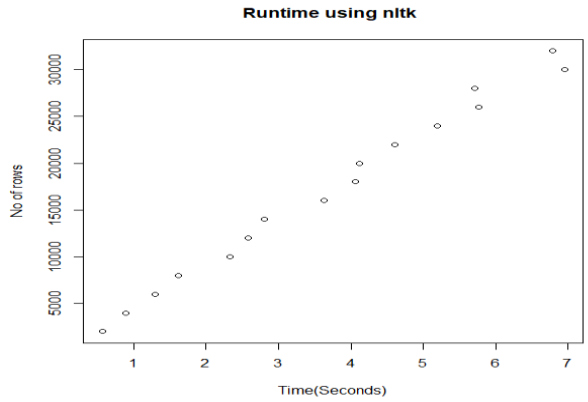


Figure 9: The above image shows the number of rows versus runtime graph for tokenisation using Regular Expressions. The graph shows a positive linear relationship.

From the graphs above, we can see the crucial difference in the runtime of tokenisation using RE and NLTK. It is also worthy to note here that, the variance in the runtime is high when tokenisation is performed using RE, whereas the graph is more linear in the case of tokenisation using NLTK, suggesting a more stable increase in the runtime, as the number of rows increases.

## VII. SUGGESTED WAYS FOR IMPROVEMENT

The inaccuracies in the results of NLTK algorithms like stemming and tokenisation can be rectified by understanding the role a word plays with respect to its context. Part of Speech tagging is a useful algorithm to execute this. Stemming and lemmatisation have their own set of drawbacks but when used in conjunction, they give the best results. Some important particulars to consider in the process of breaking down words using stemming and lemmatisation are:

Proper nouns should be identified in the sentence and Named Entity Recognition should be executed on these nouns to better understand their usage with respect to the context[4]. Once proper nouns are identified, they must be excluded from the processes of stemming and lemmatisation. In this way, we can avoid any incorrect inferences and focus on the words that need to be broken down.

It is important to understand the etymology and linguistics of words before stemming them down. Stems of words should be identified only after considering their semantics with respect to the context. The same words can have different meanings when used under different contexts. Hence, it is important to analyse tokens holistically.

Simpler words like auxiliary verbs do not require to be stemmed or lemmatised. Normalising of the tense can be done if it is required for the processing of the text.

## VIII. CONCLUSION

A better understanding of the algorithms of Natural Language Processing will help in reducing the complexity of the task of providing computers the ability to understand human languages. NLTK is a powerful tool that helps scholars and researchers perform the intricate task of natural language processing. The various NLP tasks available in NLTK helps in understanding the complex endeavour of breaking down human languages. However, these tools do bring with them, certain drawbacks that needs to be rectified. Limitations like producing incorrect outputs and slower runtime, reduce the accuracy of these tools. In comparing NLTK and RE, we were able to find out the advantage and disadvantage of using each of these methods to perform NLU algorithms. Once the accuracies of these individual algorithms are higher, we can expect better results in Natural Language Understanding using NLTK. Though there are drawbacks in NLTK's algorithms, these drawbacks are amendable and once rectified, it could be a more influential tool, helping academics and researchers achieve further accuracy with Natural Language Processing.

## REFERENCES

[1] "What is Natural Language Processing? | IBM." [Online]. Available: https://www.ibm.com/topics/natural-language-processing

[2] Excelsior, "Natural Language Processing- How different NLP Algorithms work," Medium. [Online]. Available: https://medium.com/@get_excelsior/natural-language-processing-how-different-nlp-algorithms-work-daa233b9cee0

[3] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: state of the art, current trends and challenges," *Multimed. Tools Appl.*, vol. 82, no. 3, pp. 3713–3744, Jan. 2023, doi: 10.1007/s11042-022-13428-4.

[4] B. Jehangir, S. Radhakrishnan, and R. Agarwal, "A survey on Named Entity Recognition — datasets, tools, and methodologies," *Nat. Lang. Process. J.*, vol. 3, p. 100017, Jun. 2023, doi: 10.1016/j.nlp.2023.100017.

[5] E. Kavlakoglu, "NLP vs. NLU vs. NLG: the differences between three natural language processing concepts," IBM Blog. [Online]. Available: https://www.ibm.com/blog/nlp-vs-nlu-vs-nlg-the-differences-between-three-natural-language-processing-concepts/

 DOI : 10.35291/2454-9150.2024.0004 

[6] "NLTK :: Natural Language Toolkit." [Online]. Available: https://www.nltk.org/

[7] Bird, Steven, Edward Loper and Ewan Klain (2009), "Natural Language Processing with Python [Book]." O'Reilly Media Inc

[8] *What is NLP (Natural Language Processing)?*, (Aug. 12, 2021). [Online Video]. Available: https://www.youtube.com/watch?v=fLvJ8VdHLA0

[9] "Twitter Sentiment Analysis and Word Embeddings." [Online]. Available: https://www.kaggle.com/datasets/ibrahimserouis99/twitter-sentiment-analysis-and-word-embeddings



 DOI : 10.35291/2454-9150.2024.0004 