Natural Language Processing

Chapter 1

Introduction to NLP

Natural Language Processing (NLP) is a subfield of artificial intelligence (AI) that focuses on the interaction between computers and human language.

It enables machines to understand, interpret, and generate human language in a way that is both meaningful and useful.

NLP combines computational linguistics with machine learning and deep learning to process and analyze large amounts of natural language data.

Key components of NLP

Tokenization: Breaking text into individual words or phrases (tokens).

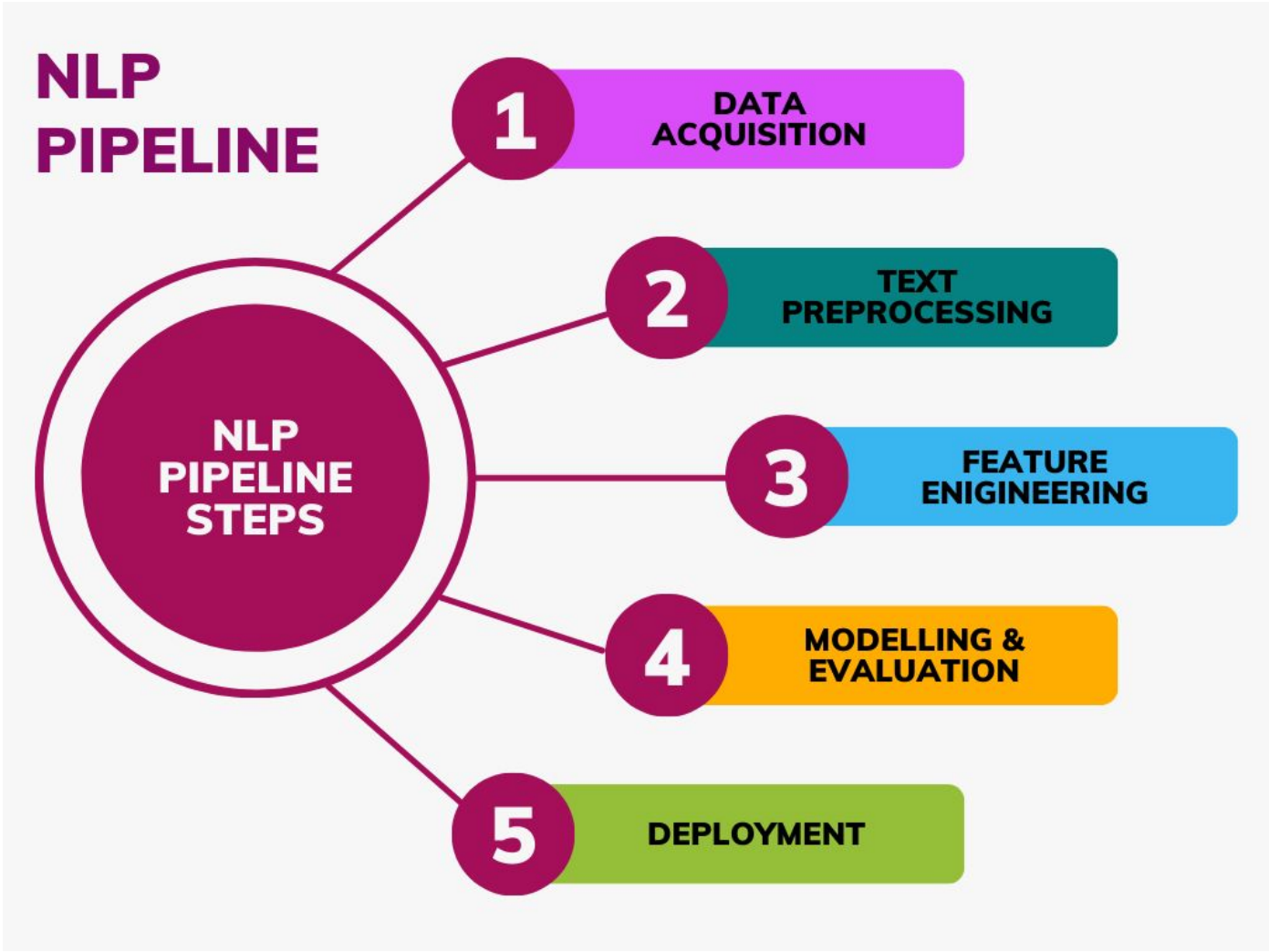
Part-of-Speech Tagging: Identifying the grammatical parts of speech (e.g., nouns, verbs, adjectives).

Named Entity Recognition (NER): Detecting and classifying entities like names, dates, and locations.

Syntax and Parsing: Analyzing the grammatical structure of sentences.

Semantic Analysis: Understanding the meaning of words and sentences.

Machine Learning Models: Using algorithms to learn patterns from text data.


NLP
PIPELINE
NLP
PIPELINE
STEPS
1
DATA
ACQUISITION
2
TEXT
PREPROCESSING
3
FEATURE
ENIGINEERING
4
MODELLING &
EVALUATION
5
DEPLOYMENT

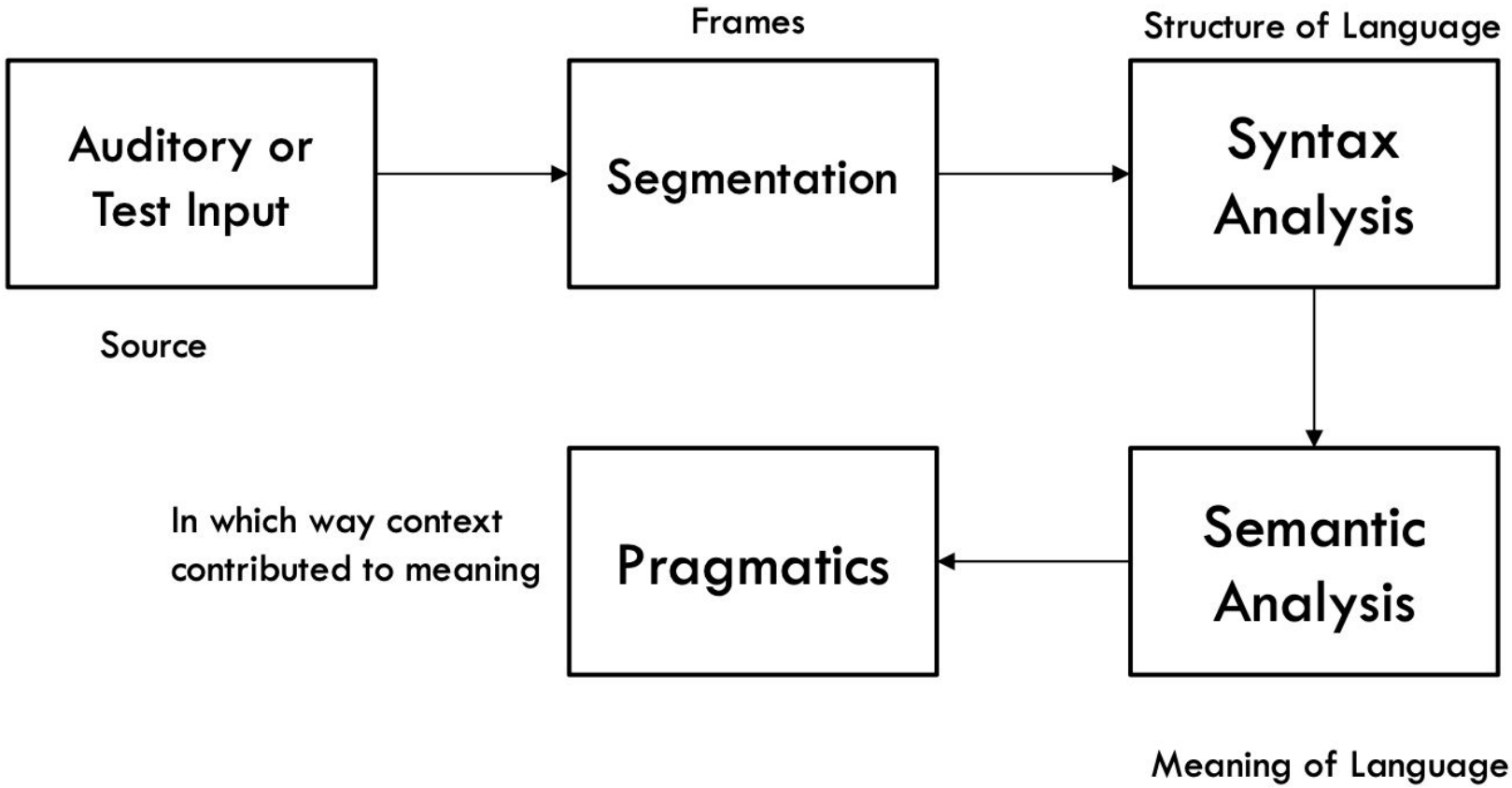
Natural language processing

NLP is one of the field of AI that processes or analyses written or spoken language

NLP=NLG+NLU, where NLG is about generation and NLU is about understanding the natural language

Understanding language requires a lot of knowledge

NLP Processes



NLP Components

Natural Language Understanding

Understanding involves the following tasks

Mapping the given input in natural language into useful representations.

Analyzing different aspects of the language

NLP Component

Natural Language Generation

It is the process of producing meaningful phrases and sentences in the form of natural language from some internal representation. It involves :-

Text planning − It includes retrieving the relevant content from knowledge base.

Sentence planning − It includes choosing required words, forming meaningful phrases, setting tone of the sentence.

Text Realization − It is mapping sentence plan into sentence structure. The NLU is harder than NLG.

NLP Component

Lexical ambiguity − It is at very primitive level such as word-level. For example, treating the word “board” as noun or verb?

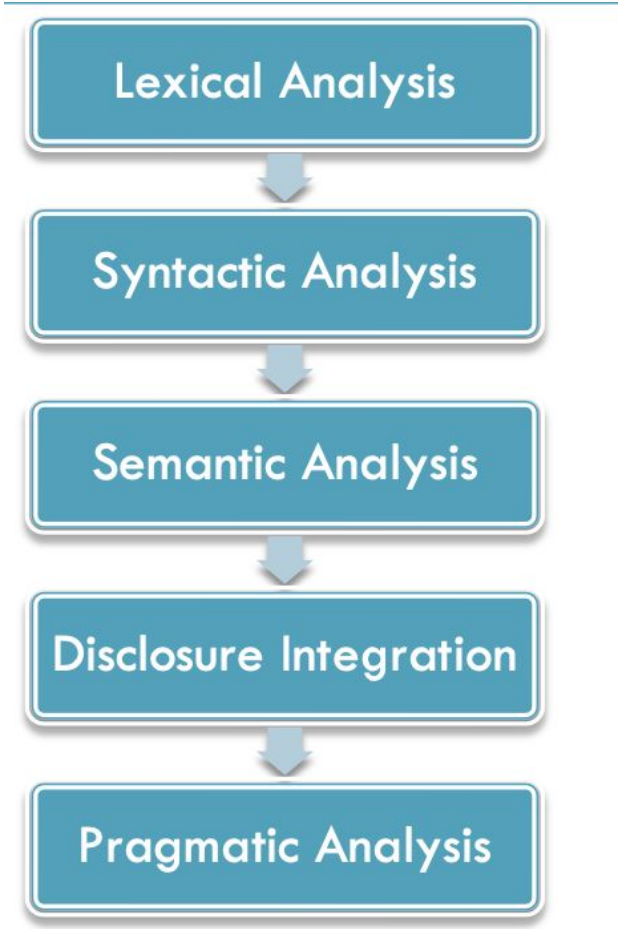
Syntax Level ambiguity − A sentence can be parsed in different ways. For example, “He lifted the beetle with red cap.” − Did he use cap to lift the beetle or he lifted a beetle that had red cap?

Referential ambiguity − Referring to something using pronouns. For example, Rima went to Gauri. She said, “I am tired.” − Exactly who is tired?

One input can mean different meanings.

Many inputs can mean the same thing

NLP Steps


Lexical Analysis
Syntactic Analysis
Semantic Analysis
Disclosure Integration
Pragmatic Analysis

NLP Steps

Lexical Analysis − It involves identifying and analyzing the structure of words. Lexicon of a language means the collection of words and phrases in a language. Lexical analysis is dividing the whole chunk of text into paragraphs, sentences, and words.

Syntactic Analysis Parsing − It involves analysis of words in the sentence for grammar and arranging words in a manner that shows the relationship among the words. The sentence such as “The school goes to boy” is rejected by English syntactic analyzer.

NLP Steps

Semantic Analysis − It draws the exact meaning or the dictionary meaning from the text. The text is checked for meaningfulness. It is done by mapping syntactic structures and objects in the task domain. The semantic analyzer disregards sentence such as “hot icecream”.

Discourse Integration − The meaning of any sentence depends upon the meaning of the sentence just before it. In addition, it also brings about the meaning of immediately succeeding sentence.

Pragmatic Analysis − During this, what was said is re-interpreted on what it actually meant. It involves deriving those aspects of language which require real world knowledge.

Morphological and Lexical Analysis

- The lexicon of a language is its vocabulary that includes its words and expressions
- Morphology depicts analyzing, identifying and description of structure of words
- Lexical analysis involves dividing a text into paragraphs, words and the sentences

Syntactic Analysis

- Syntax concerns the proper ordering of words and its affect on meaning
- This involves analysis of the words in a sentence to depict the grammatical structure of the sentence
- The words are transformed into structure that shows how the words are related to each other
- Eg. “the girl the go to the school”. This would definitely be rejected by the English syntactic analyzer

Semantic Analysis

- Semantics concerns the (literal) meaning of words, phrases, and sentences
- This abstracts the dictionary meaning or the exact meaning from context
- The structures which are created by the syntactic analyzer are assigned meaning
- E.g.. “colorless blue idea” .This would be rejected by the analyzer as colorless blue do not make any sense together

Discourse Integration

- Sense of the context
- The meaning of any single sentence depends upon the sentences that precedes it and also invokes the meaning of the sentences that follow it
- E.g. the word “it” in the sentence “she wanted it” depends upon the prior discourse context

Pragmatic Analysis

- Pragmatics concerns the overall communicative and social context and its effect on interpretation
- It means abstracting or deriving the purposeful use of the language in situations
- Importantly those aspects of language which require world knowledge
- The main focus is on what was said is reinterpreted on what it actually means
- E.g. “close the window?” should have been interpreted as a request rather than an order

Parsing

- The first task for any NLP-based system is to *read (or to parse)* the text
- Parsing depends on three components of a language-

1. Lexicon
2. Categorization
3. Grammar Rules

Lexicon

stench | breeze | glitter | nothing | wumpus | pit | pits | gold | east | ..

is | see | smell | shoot | feel | stinks | go | grab | carry | kill | turn | …

right | left | east | south | back | smelly | …

here | there | nearby | ahead | right | left | east | south | back | …

me | you | I | it | S=HE | Y'ALL …

John | Mary | Boston | UCB | PAJC | …

the | a | an | …

to | in | on | near | …

and | or | but | …

0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Categorization

Noun > stench | breeze | glitter | nothing | wumpus | pit | pits | gold | east | ..

Verb > is | see | smell | shoot | feel | stinks | go | grab | carry | kill | turn | …

Adjective > right | left | east | south | back | smelly | …

Adverb > here | there | nearby | ahead | right | left | east | south | back | …

Pronoun > me | you | I | it | S=HE | Y'ALL …

Name > John | Mary | Boston | UCB | PAJC | …

Article > the | a | an | …

Preposition > to | in | on | near | …

Conjunction > and | or | but | …

Digit > 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Grammar Rules

- “The large cat”
- This phrase can be parsed by an NLP-system if it has a grammar like

  Noun Phrase -> Determiner + Adjective + Noun
- If your system finds a phrase or sentence that has a pattern not mentioned in its set of Grammar Rules it won’t be able to parse them.

Therefore…

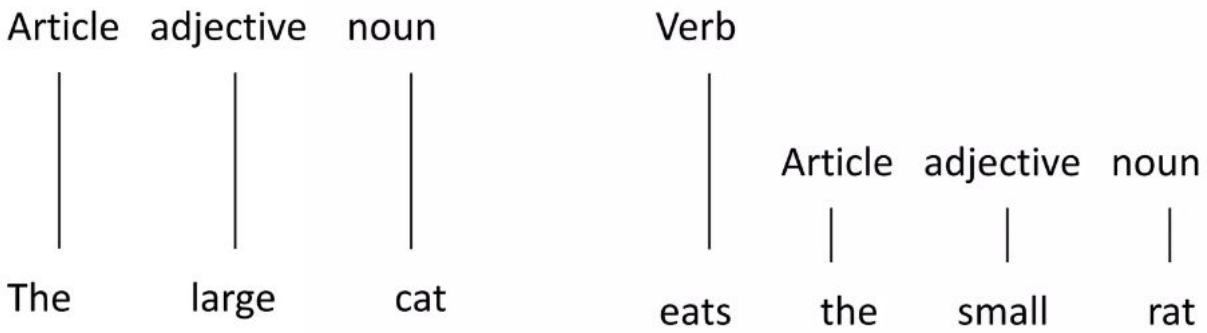
- Parsing is the process of using grammar rules to determine whether a sentence is legal,
- and to obtain its Syntactic Tree

Syntactic Tree

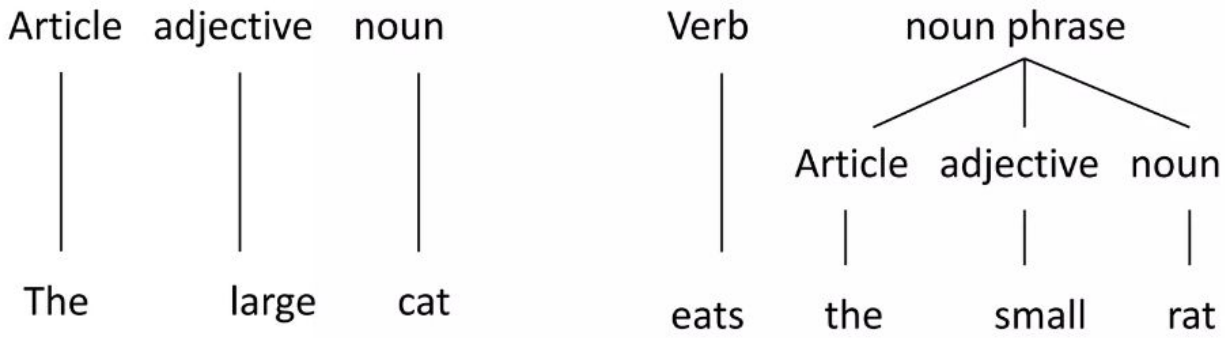
'The large cat eats the small rat'

Syntactic Tree

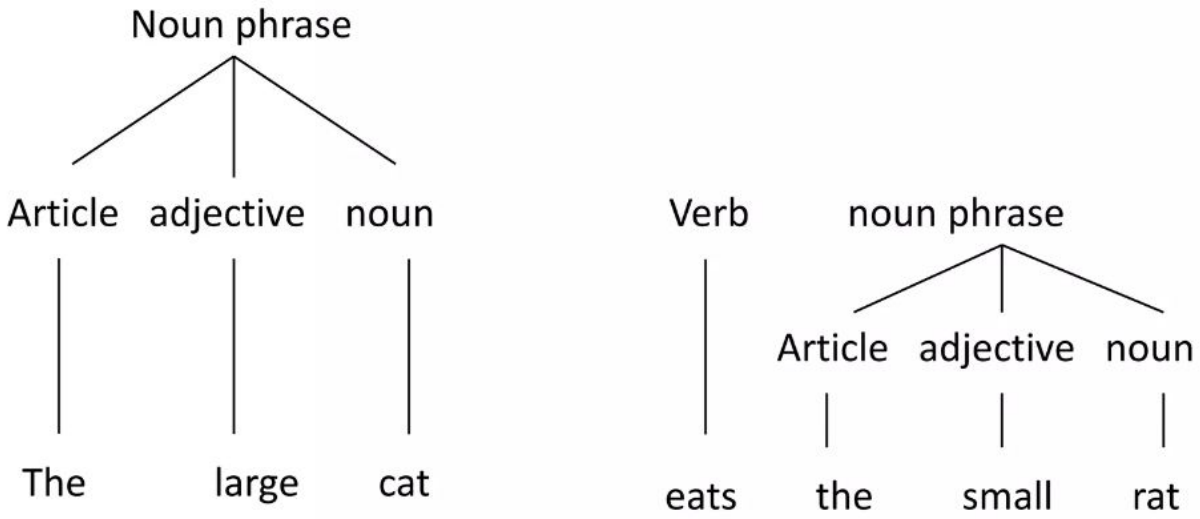
Syntactic Tree



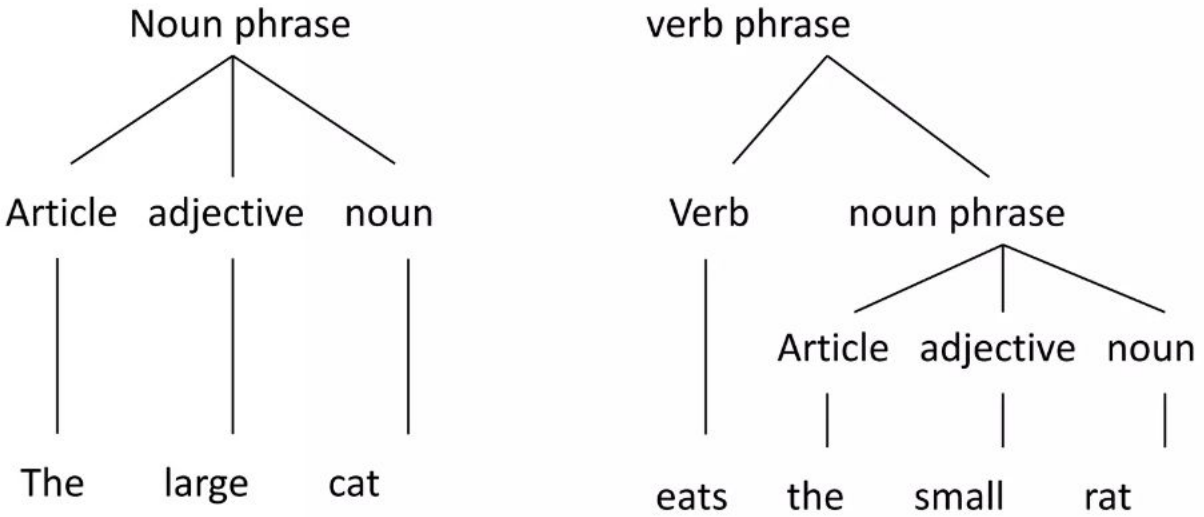
Syntactic Tree



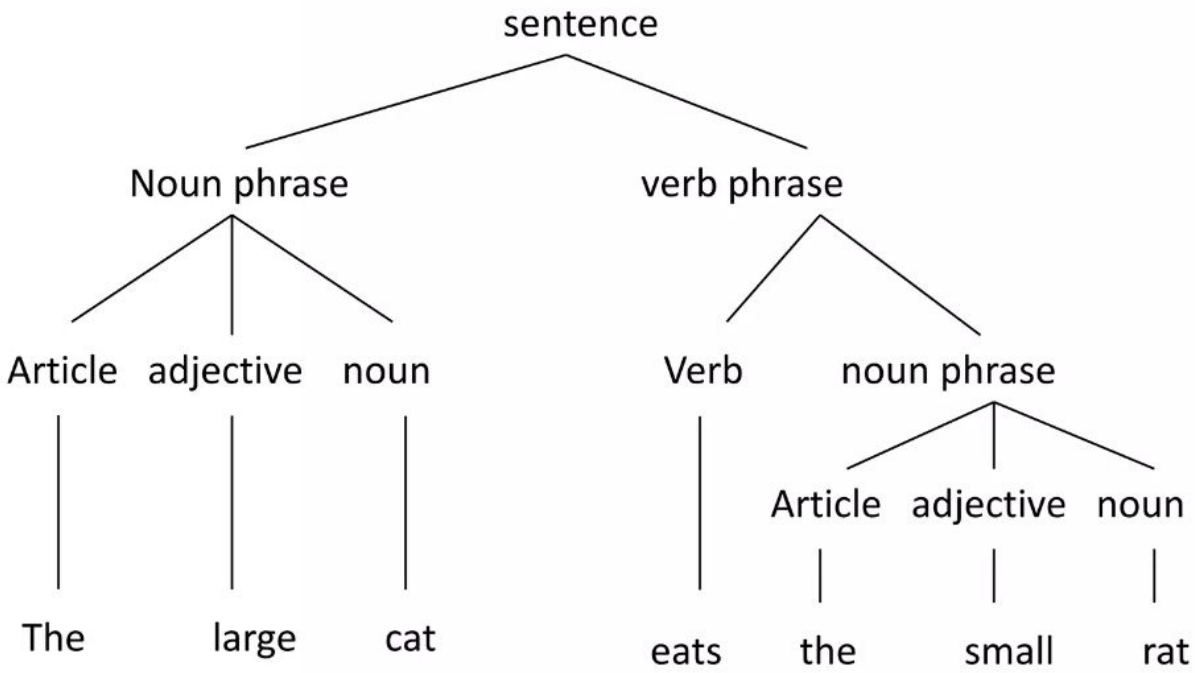
Syntactic Tree



Syntactic Tree



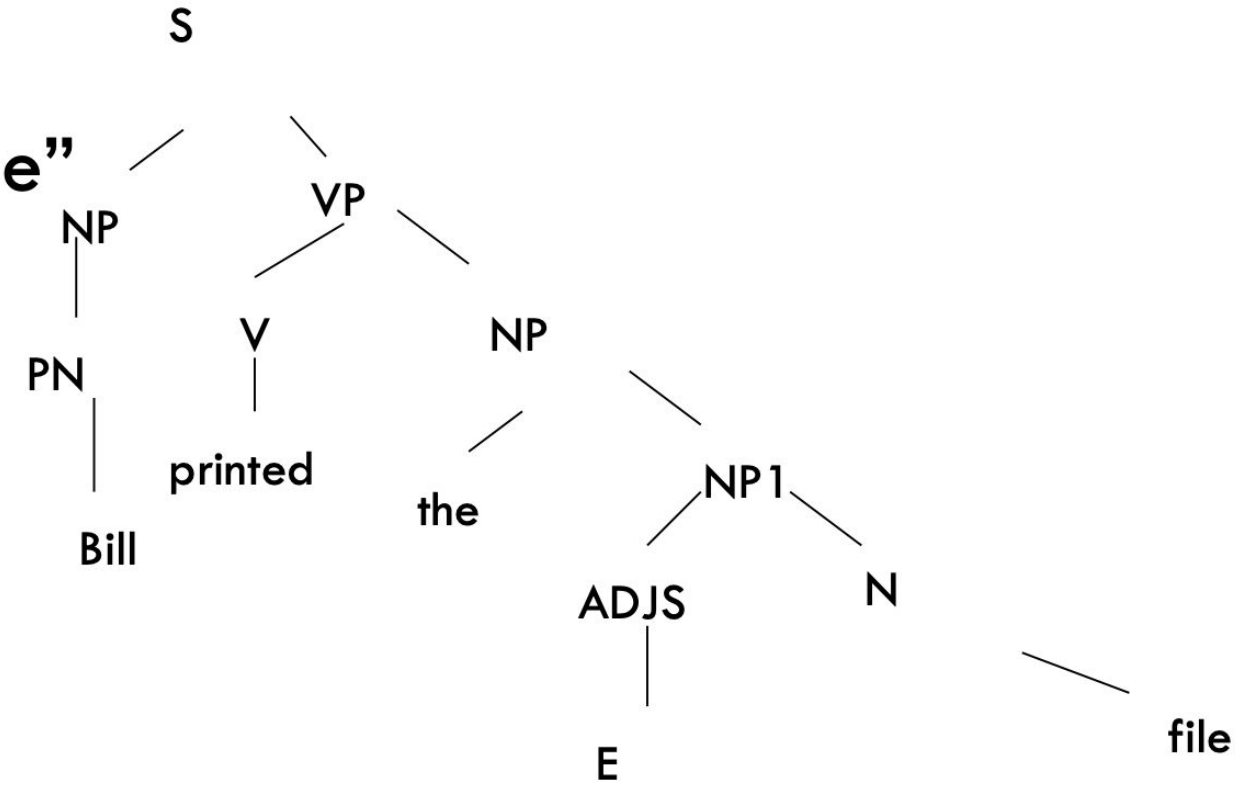
Syntactic Tree



NLP

- Parse Tree

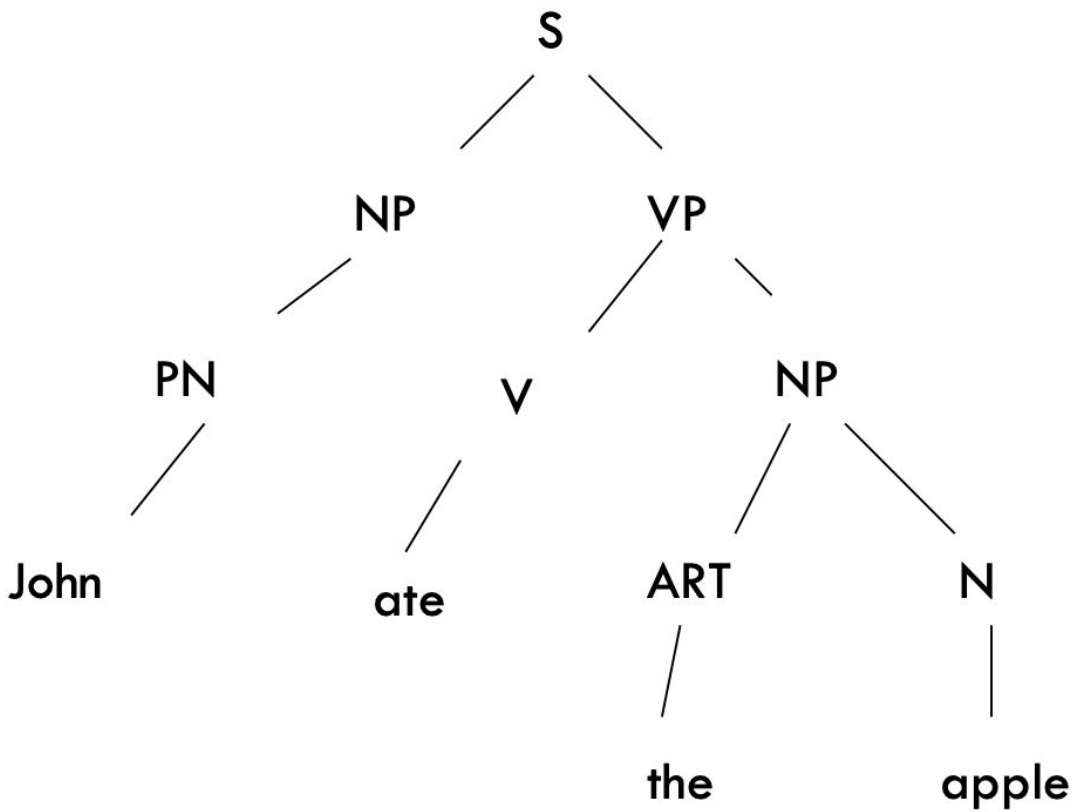
"Bill Printed the file"



A parse tree :

John ate the apple.

1. S -> NP VP
2. VP -> V NP
3. NP -> NAME
4. NP -> ART N
5. NAME -> John
6. V -> ate
7. ART-> the
8. N -> apple



Introduction to NLP (using colab or jupyter notebook)

```
#Create a text file
text=open("untitled.txt")
text=text.read()
print(text)
```

Import Libraries

```
import nltk
from nltk import sent_tokenize
from nltk import word_tokenize
```

Sentence Level tokenization

```
sentences = sent_tokenize(text)

sentences
```

Word level Tokenization

```
words=word_tokenize(text)
print(words)
```

Frequent words

```
from nltk.probability import FreqDist
fdist = FreqDist(words)
fdist.most_common(10)
```

Plotting of the frequency of words

```
import matplotlib.pyplot as plt
fdist.plot(10)
```

Removal of Punctuation

```
words_nopunc = []
for w in words:
 if w.isalpha():
    words_nopunc.append(w.lower())

print(words_nopunc)
```

Frequency distribution (again)

```
fdist=FreqDist(words_nopunc)
fdist.plot(10)
```

Importing stop words

```
from nltk.corpus import stopwords


stopwords = stopwords.words('english')
print(stopwords)
```

Removing unnecessary words

```
words_clean=[]
for w in words_nopunc:
 if w not in stopwords:
   words_clean.append(w)

print(words_clean)
```

```
fdist=FreqDist(words_clean)
fdist.plot(10)
```

Importing word cloud

```
from wordcloud import WordCloud
wordcloud = WordCloud().generate(text)
plt.figure(figsize=(12,12))
plt.imshow(wordcloud)

plt.axis("off")
plt.show()
```

```
str1 = " "
w=str1.join(words_clean)
wordcloud = WordCloud().generate(w)
plt.figure(figsize=(12,12))
plt.imshow(wordcloud)

plt.axis("off")
plt.show()
```

NLP Steps using Streamlit

```
import streamlit as st

import nltk

from nltk import sent_tokenize, word_tokenize

from nltk.probability import FreqDist

from nltk.corpus import stopwords

from wordcloud import WordCloud

import matplotlib.pyplot as plt
```

Downloading the necessary resources

```
st.title("NLP Text Analysis with NLTK")

@st.cache_resource
def download_nltk_data():
    nltk.download('punkt')
    nltk.download('punkt_tab')
    nltk.download('stopwords')


download_nltk_data()
```

Input text area

st.header("Input Text")

text = st.text_area("Enter text to analyze:", height=300)

Create a button and divide the col on sentence and word level tokenization

```
if st.button("Analyze") and text:
    st.divider()
    st.header("Tokenization")
    col1, col2 = st.columns(2)
    with col1:
        st.subheader("Sentences")
        sentences = sent_tokenize(text)
        st.write(f"Number of sentences: {len(sentences)}")
        st.write(sentences)
```

Col 2 for word level tokenization

```
with col2:
        st.subheader("Words")
        words = word_tokenize(text)
        st.write(f"Number of words: {len(words)}")
        st.write(words)


    st.divider()
```

```
st.header("Frequency Distribution")

    fdist = FreqDist(words)
    st.subheader("Most Common Words (Raw)")
    st.write(fdist.most_common(10))

    st.subheader("Frequency Plot (Raw)")
    fig, ax = plt.subplots()
    fdist.plot(10, show=False)
st.pyplot(plt)
    plt.clf()
st.divider()
```

Cleaning text

```
st.header("Text Cleaning")
    words_nopunc = [w.lower() for w in words if w.isalpha()]
    st.write(f"Words without punctuation: {len(words_nopunc)}")
    # st.write(words_nopunc) # Optional: don't show all if too many


    st.subheader("Frequency Plot (No Punctuation)")
    fdist_nopunc = FreqDist(words_nopunc)
    fdist_nopunc.plot(10, show=False)
    st.pyplot(plt)
    plt.clf()
```

Stop words removal

```
st.subheader("Stopwords Removal")
    stop_words = set(stopwords.words('english'))
    words_clean = [w for w in words_nopunc if w not in stop_words]
    st.write(f"Cleaned words (no stopwords): {len(words_clean)}")
    st.write(words_clean)
   st.divider()
```

Word Cloud generation

```
st.header("Word Cloud")

    if words_clean:

        wordcloud_text = " ".join(words_clean)

        wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(wordcloud_text)

        fig_wc, ax_wc = plt.subplots(figsize=(10, 5))

        ax_wc.imshow(wordcloud, interpolation='bilinear')

        ax_wc.axis("off")

        st.pyplot(fig_wc)

    else:

        st.warning("No words left after cleaning to generate Word Cloud.")
```