

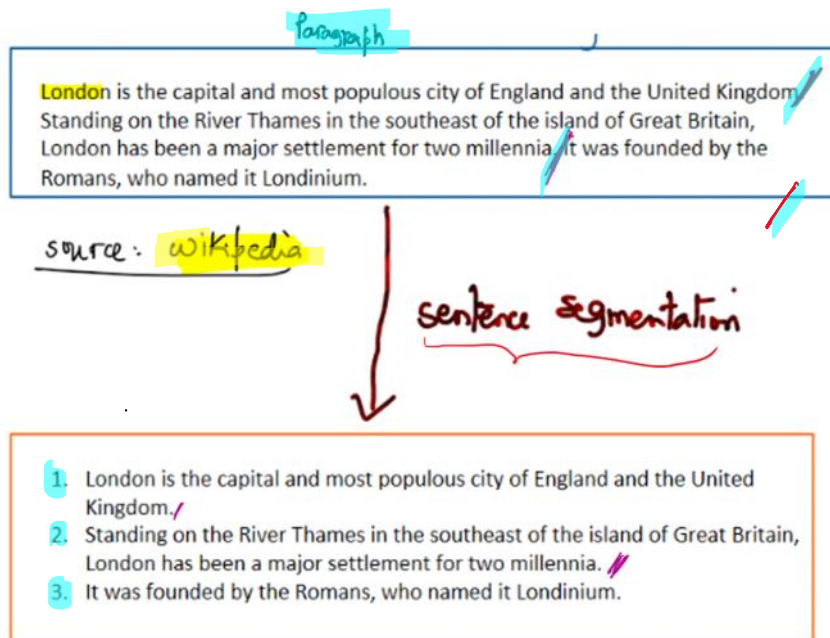
Focus on the steps (not the sequence)

## Natural Language Processing Pipeline



Credit: 7 Turing

### step # 1: Sentence Segmentation



- it is the very first step in the NLP pipeline
- = it divides the entire paragraph into different sentences for clarity and better understanding.

### step # 2 Word tokenization:

What is a token?

In the context of large language models (LLMs), a token is a small unit of text that the model processes at once.

Sentence #1 "I am learning" → 3 tokens: ["I", "am", "learning"]

every token is a word:

Sentence #2 "ChatGPT is powerful" → 4 tokens ["chat", "G", "PT", "is powerful"]

tokens ⇒ words

unbelievable → 2 tokens ["un", "believable"]

# word tokenization is the process of splitting a sentence or paragraph into individual words tokens (word tokens)

# These are building blocks used for further analysis such as POS (parts of speech) tagging, sentiment analysis or even input to machine learning models.

Sentence: I love teaching NLP.

word tokens: ["I", "love", "teaching", "NLP"]

why is word tokenization important?

## ① Feature Extraction

- Many NLP algorithms work on the individual word and each distinct word should be tagged as ONE token even when it is repeating multiple times

① learning | ② teaching | ③ | ④

## ② Reducing Complexity

Break...

## reducing complexity

Breaking down a sentence or text into tokens such as words, group of words → it simplifies the processing tasks

[Text → millions of words → thousands of sentences]

How does LLM internally divide them? Will it follow any algorithm to make it happen?

intent  
↓  
(action)

## Sub-word tokens

I am unhappy with the taxation rate.

(omg! it's such a positive sentiment) X → interpretation is WRONG,  
it's actually -ve sentiment not positive

- unhappiness → ["un", "happiness"]
  - unwell → ["un", "well"]
- } sub-word tokens to understand the context better.

## # Rule-based tokenization

- it handles punctuation & contractions

Don't go! → ["Don", "'", "t", "go", "!", ""]

I've told.

We'll meet.

U.S.A. is a big country → ["U", "S", "A", " ", "i", "s", " ", "a", " ", "b", "i", "g", " ", "c", "o", "u", "n", "t", "r", "y"]  
 ↳ Avoid splitting U.S.A.

↓  
 needs to be considered as one token

# ask2apc @ gmail.com  
 one token

# "hello 🍌"  
 character tokens → support emojis (Unicode handling)  
 "I ❤️ NLP!" → ['I', ' ', '❤️', ' ', 'N', 'L', 'P', '!'] → character tokens

## # RegEx-based tokenizations

↳ Regular Expressions

— splits the text into tokens (words, phrases, characters, symbols)  
 based on custom-pattern matching rules.

Regex	Meaning	Example Input	output
\w+	word characters	let's go!	["let", "s", "go"]
r '#\w+'	Hashtags	#AI is awesome	["#AI"]

Note: • Highly customized for specific text formats  
 • No dependency on pre-trained models → rule based tokenization  
 • Lightweight and fast

Cons: • Regex doesn't have linguistic awareness  
 • The ...

- Cons:
- Regex doesn't have linguistic awareness
  - It's hard to handle context while doing Regex.