



NLP

**Natural language processing**

Parsa Ahani



## Definition

Natural Language Processing or NLP is a field of Artificial Intelligence that gives the machines the **ability to read, understand and derive meaning from human languages.**

Goal :

How machines process and  
understand human language

London is the capital and most populous city of England and the United Kingdom. Standing on the River Thames in the south east 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. London's ancient core, the City of London, largely retains its 1.12-square-mile (2.9 km<sup>2</sup>) medieval boundaries.



## Use cases :

In simple terms, NLP represents the automatic handling of natural human language like speech or text, and although the concept itself is fascinating, the real value behind this technology comes from the use cases.

- 1) Companies like Yahoo and Google filter and classify your emails with NLP by analyzing text in emails that flow through their servers and **stopping spam** before they even enter your inbox.
- 2) Amazon's Alexa and Apple's Siri are examples of intelligent **voice driven interfaces** that use NLP to respond to vocal prompts and do everything like find a particular shop, tell us the weather forecast, suggest the best route to the office or turn on the lights at home.



## Use cases :

### 3) health :

couple of years ago Microsoft demonstrated that by analyzing large samples of search engine queries, they could identify internet users who were suffering from **cancer** even before they have received a diagnosis of the disease



difficulty:

The main drawbacks we face these days with NLP relate to the fact that language is very **tricky**. The process of understanding and manipulating language is extremely complex or

Some of these rules can be high-leveled and abstract; for example, when someone uses a sarcastic remark to pass information.

Computers are great at working with **structured** data like spreadsheets and database tables. But us humans usually communicate in words, not in tables. That's unfortunate for computers.



# Algorithms in NLP



Step 1 ) sentence segmentation



## Step 2 ) Tokenization:

Splitting on blank spaces may break up what should be considered as one token, as in the case of certain names (e.g. **San Francisco** or New York)

Tokenization can remove punctuation too, easing the path to a proper word segmentation but also triggering possible complications. In the case of periods that follow abbreviation (e.g. dr.), the period following that abbreviation should be considered as part of the same token and not be removed.

## Step 3 ) Predicting parts of speech for each token

The part-of-speech model was originally trained by **feeding it millions of English sentences** with each word's part of speech already tagged and having it learn to replicate that behavior.

<b>London</b>	<b>is</b>	<b>the</b>	<b>capital</b>	<b>and</b>	<b>most</b>	<b>populous ...</b>
Proper Noun	Verb	Determiner	Noun	Conjunction	Adverb	Adjective

## step 4 ) lemmatization:

Has the objective of reducing a word to its base form and grouping together different forms of the same word. For example, verbs in past tense are changed into present (e.g. **“went” is changed to “go”**) and synonyms are unified (e.g. “best” is changed to “good”), hence standardizing words with similar meaning to their root. Although it seems closely related to the stemming process, lemmatization uses a different approach to reach the root forms of words.

**Lemmatization is typically done by having a look-up table of the lemma forms** of words based on their part of speech and possibly having some custom rules to handle words that you’ve never seen before

## Step 5 | Bag of words and identifying stop words:

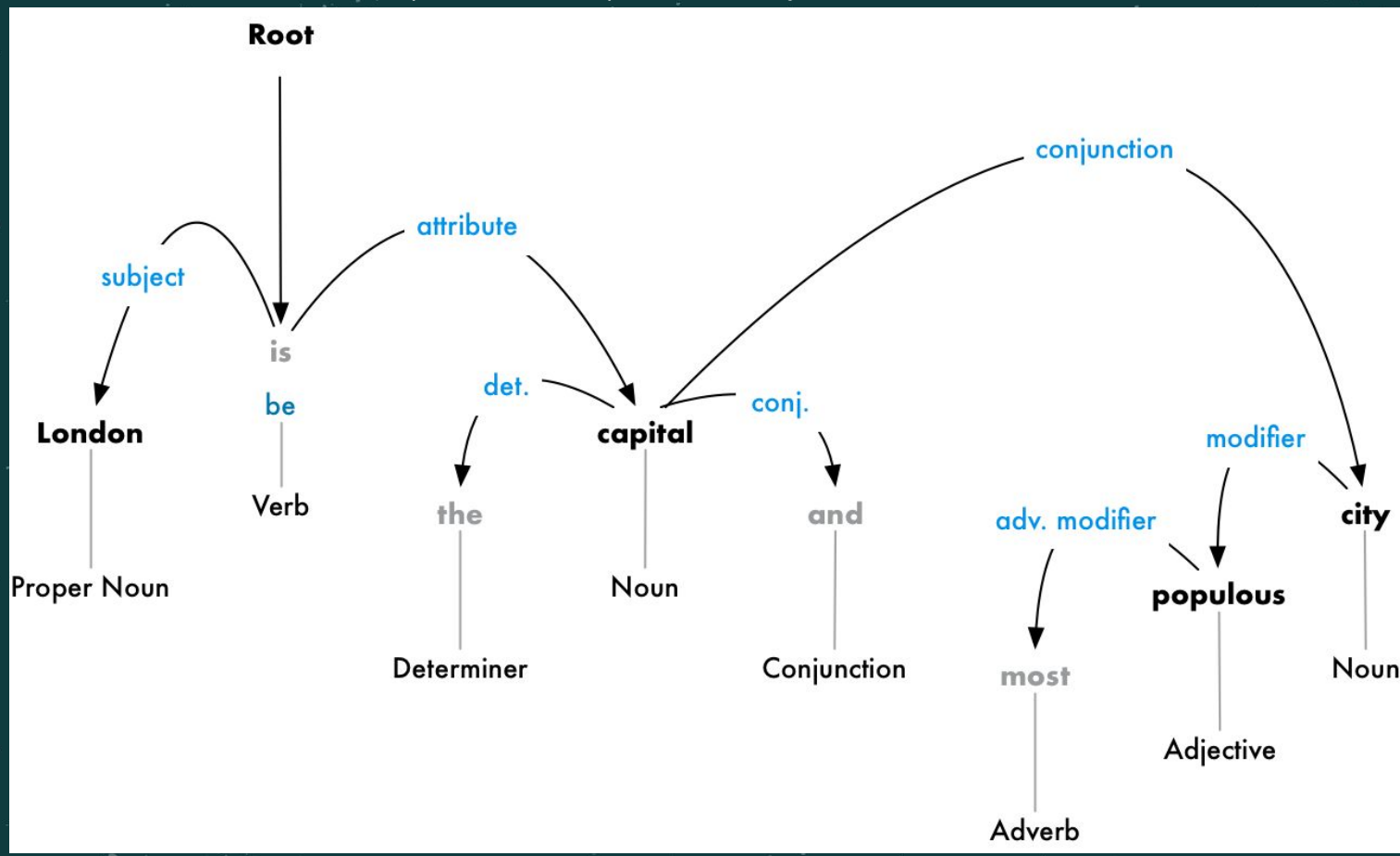
	words	rain	a	paper	they	slip	the	universe	...
<i>Words are flowing out like endless rain into a paper cup,</i>	1	1	1	1	0	0	0	0	...
<i>They slither while they pass, they slip away across the universe</i>	0	0	0	0	3	1	1	1	...

Machine learning algorithms cannot work with raw text directly. Rather, the text must be **converted into vectors of numbers**

some words are not weighted accordingly (“universe” weights less than the word “they”). Stop words are usually identified by just by checking a hardcoded list of known stop words.

For example if you are building a rock band search engine, you want to make sure you don’t ignore the word “The”. Because not only does the word “The” appear in a lot of band names, there’s a famous 1980’s rock band called ***The The!***

# Step 6 : Dependency Parsing:



# Step 7: Named entity recognition (NER)

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

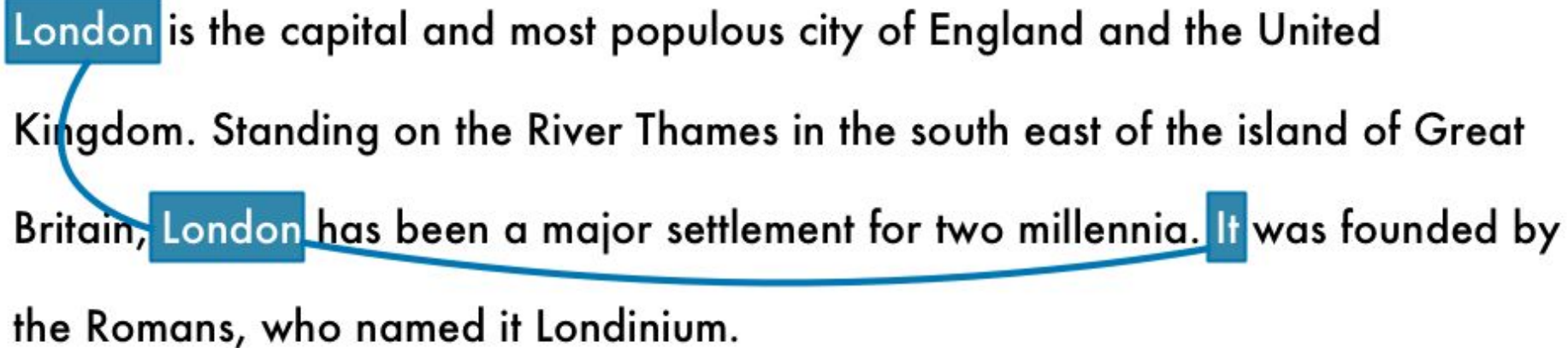
Geographic  
Entity

Geographic  
Entity

Geographic  
Entity

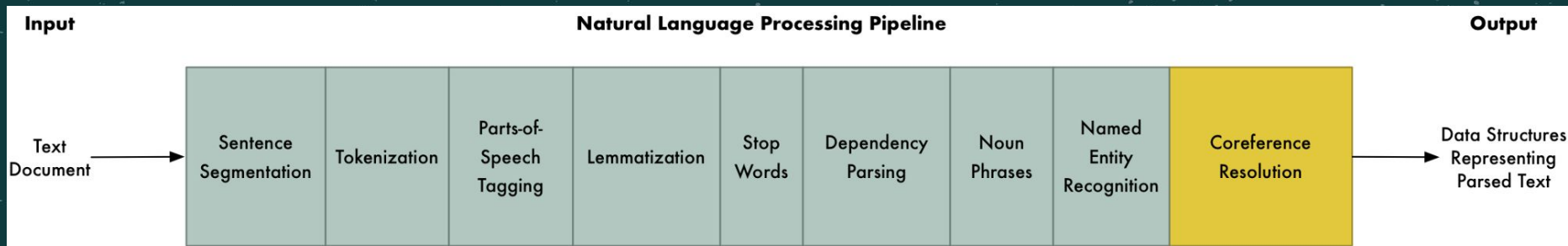
## Step 8 : Coreference Resolution

**London** is the capital and most populous city of England and the United Kingdom. Standing on the River Thames in the south east 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.

A diagram illustrating coreference resolution. The word "London" is highlighted with a blue box at the beginning of the sentence. The word "It" is highlighted with a blue box at the end of the sentence. A blue curved line connects the box around "London" to the box around "It", indicating that they refer to the same entity.



# NLP pipeline:



*Note: Before we continue, it's worth mentioning that these are the steps in a typical NLP pipeline, but **you will skip steps or re-order steps depending on what you want** to do and how your NLP library is implemented*

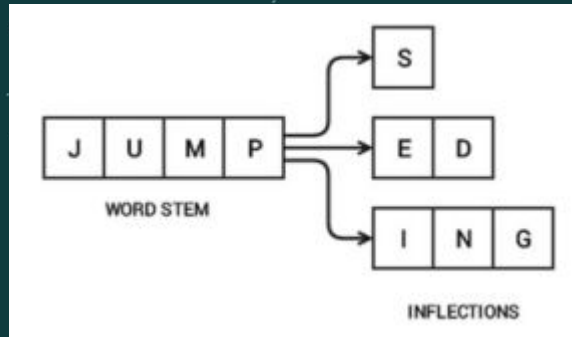
*For example, some libraries like spaCy do sentence segmentation much later in the pipeline using the results of the dependency parse.*

# stemming:

Refers to the process of slicing the end or the beginning of words with the intention of removing affixes (lexical additions to the root of the word).

The problem is that affixes can create or expand new forms of the same word, or even create new words themselves

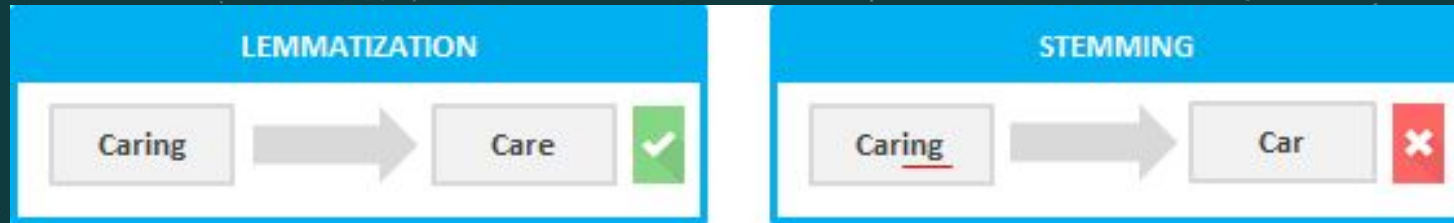
**In English**, prefixes are always derivational (the affix creates a **new word** as in the example of the prefix “eco” in the word “ecosystem”), but suffixes can be derivational (the affix creates a new word as in the example of the suffix “ist” in the word “guitarist”) or inflectional (the affix creates a new form of word as in the example of the suffix “er” in the word “faster”).



Ok so , how can we tell difference and chop the right bit ?

A possible approach is to consider a list of common affixes and rules

# difference between Lemmatization and Stemming:



# Modeling :

**Latent Dirichlet Allocation**

# LDA:

**Each document can be described by a distribution of topics  
and each topic can be described by a distribution of words and  
typically used to detect underlying topics in text  
The aim of LDA is to find topics a document belongs to**

the order of the words and the grammatical role of the words (subject, object, verbs, ...) are not considered

Alpha(mix of topic) , beta(mix of world) , number of topic , iterations

## WORKING BACKWARDS (CONT.)

1. Randomly assign each word in each document to one of the  $K$  topics.
2. For each document  $d$ :
  - Assume that all topic assignments except for the current one are correct.
  - Calculate two proportions:
    1. Proportion of words in document  $d$  that are currently assigned to topic  $t = p(\text{topic } t \mid \text{document } d)$
    2. Proportion of assignments to topic  $t$  over all documents that come from this word  $w = p(\text{word } w \mid \text{topic } t)$
  - Multiply those two proportions and assign  $w$  a new topic based on that probability.  $p(\text{topic } t \mid \text{document } d) * p(\text{word } w \mid \text{topic } t)$
3. Eventually we'll reach a steady state where assignments make sense

Alpha 0.1

