Introduction to Natural Language Processing (2/2)

Advanced Machine Learning AI

Prof. Juan Martin Maglione

Contact information - Prof. Juan Martin Maglione

E-mail: maglionejm@gmail.com
Mobile Argentina: +54 9 11 3691 7966
Mobile Spain: +34 691 82 69 82

Juan Martín Maglione holds a degree in Business Administration, with a Master's degree in Strategic and Technological Management from ITBA, a Master's Degree in Strategic and Technological Management from EOI (Spain). He has specialised in International Business Consultancy at Fontys University of Applied Sciences in Eindhoven, Holland, holds several specialisations in Big Data in ITBA, in Product Management, Agile Methodologies, Cloud Engineering and Deep Learning also in ITBA.

He is Global Product Transformation Manager for **Santander** (Madrid, Spain), Head of Artificial Intelligence and Technology for **SAI** (Amsterdam, Holland), co-Founder of **Fictionary**, a mobile game and of **Fon.ai**, an organisation that provides Artificial Intelligence and Analytics solutions for national and international public and private organisations.

He served as Regional Engagement Manager in **Cognitiva** for Argentina, Paraguay and Uruguay. He was the leader in the creation, construction and updating of the consultative methodologies used by Cognitiva, having coordinated the team of consultants throughout the region (Costa Rica, Colombia, Mexico, Chile and Argentina). He was part of the consulting firm **KPMG**, having participated in the creation of the practice of S & OP - Strategy and Operations.

Agenda

- Meeting 1 concepts review
- NLP essentials:
  - Tokenization
  - Stop Words
  - Stemming
  - Lemmatization
  - Tagging
- KNIME:
  - Tool overview
  - Data preparation for model generation
  - Model training & evaluation
- Introduction and documentation review of:
  - spaCy
  - nltk
  - Tensorflow (Model generation)

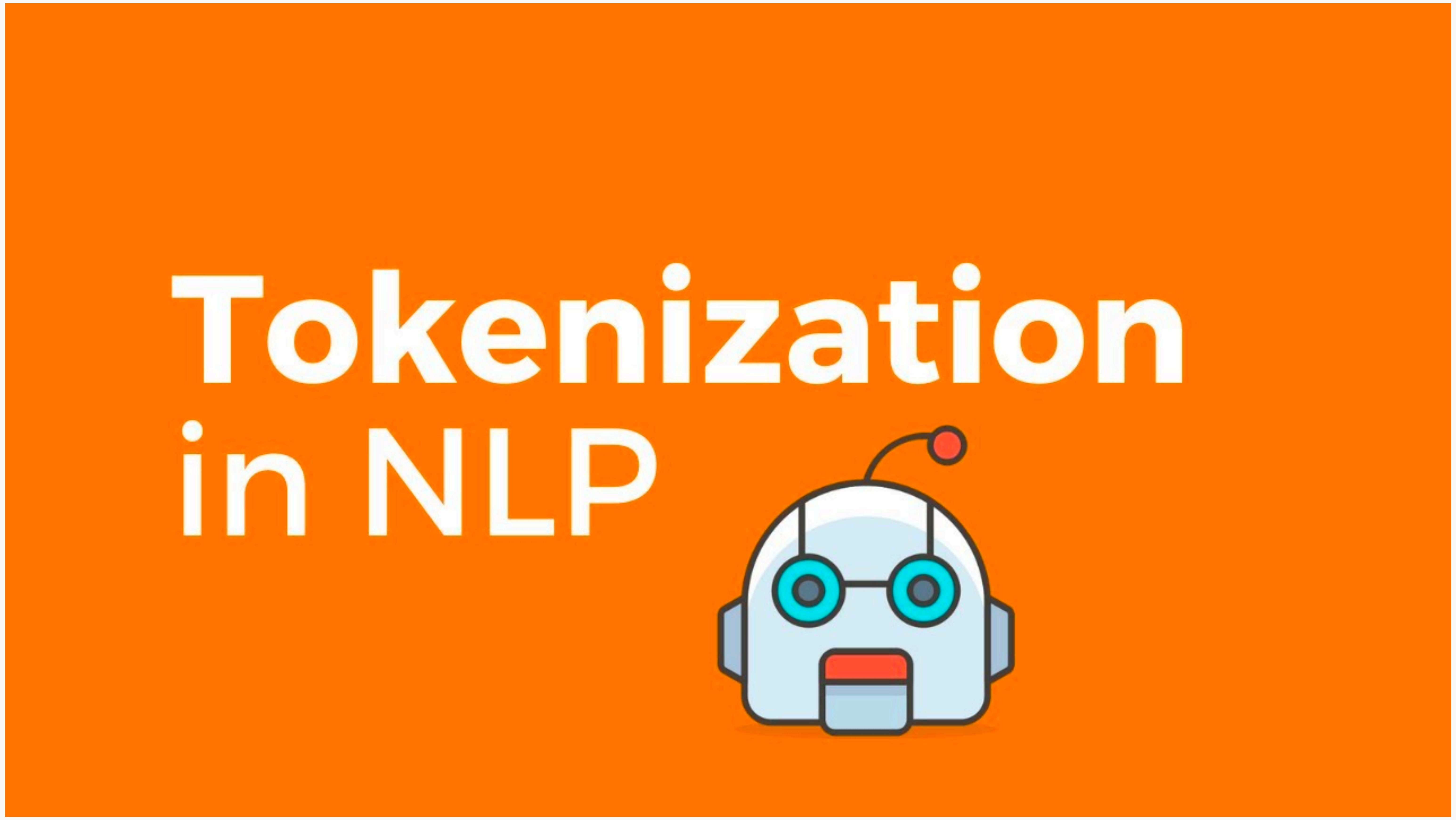
Meeting 1 concepts review

Tokenization

Is a representation of natural language that a computer can understand.

Tokenization is one of the first steps in NLP, and it’s the task of **splitting a sequence of text into units with semantic meaning**.

These units are called tokens, and the difficulty in tokenization lies on how to get the ideal split so that all the tokens in the text have the correct meaning, and there are no left out tokens.

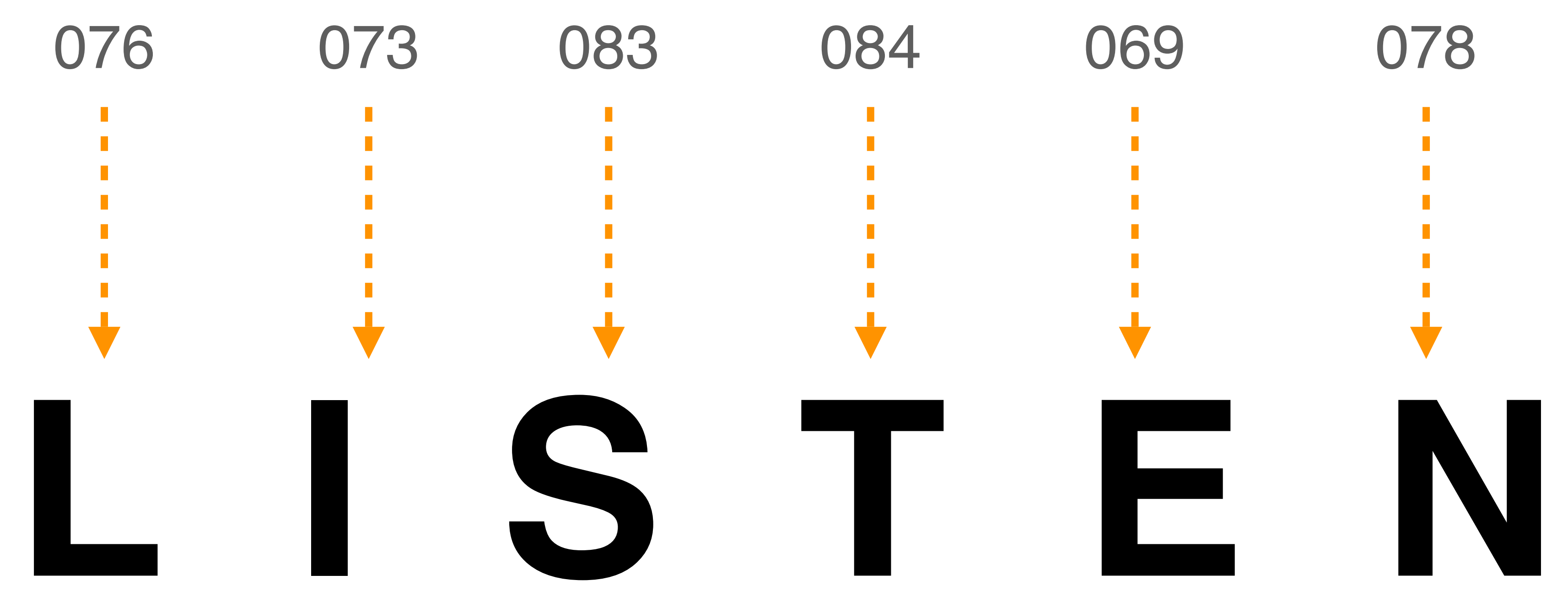


Tokenization - Character level

L I S T E N

Consider the word *“listen”*, it is made by a sequence of letters…

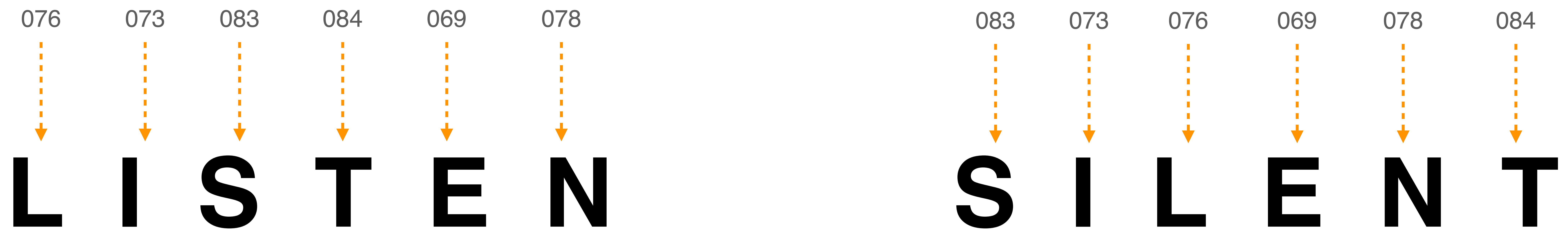
Tokenization - Character level



These letters can be represented by numbers using an **encoding scheme** (*) and the word is represented.

(*) ASCII Encoding table: https://www.asc.ohio-state.edu/demarneffe.1/LING5050/material/ASCII-Table.png

Tokenization - Character level



Hard to understand intent or sentiment of the letters as both have same numbers in different order.

Tokenization - Character level

Drawbacks of this approach:

- **Lack of meaning:** Unlike words, characters don’t have any inherent meaning, so there is no guarantee that the resultant learned representations will have any meaning. Letters may be combined into slightly unusual combinations which are not correct words. More training data should help alleviate this tendency, but we are still left with a situation where the model is losing the semantic-specific feature of words.

- **Increased input computation:** If you use word level tokens then you will spike a 7-word sentence into 7 input tokens. However, assuming an average of 5 letters per word (in the English language) you now have 35 inputs to process. This increases the complexity of the scale of the inputs you need to process.

- **Limits network choices:** Increasing the size of your input sequences at the character level also limits the type of neural networks you can use. It is difficult to use architectures which process input sequentially since your input sequences will be much longer. However, new models such as BERT are based on the Transformer architecture, which processes inputs in parallel, meaning that this is no longer a major limitation. However, it still impacts your choice of networks since it is difficult to perform some type of NLP tasks on characters. If you are training a Parts Of Speech (POS) tagger or a classifier then it is more difficult to work at the character level. You will need to do more work than if you trained it at a word level to optimize for your task.
-

Tokenization - Word level

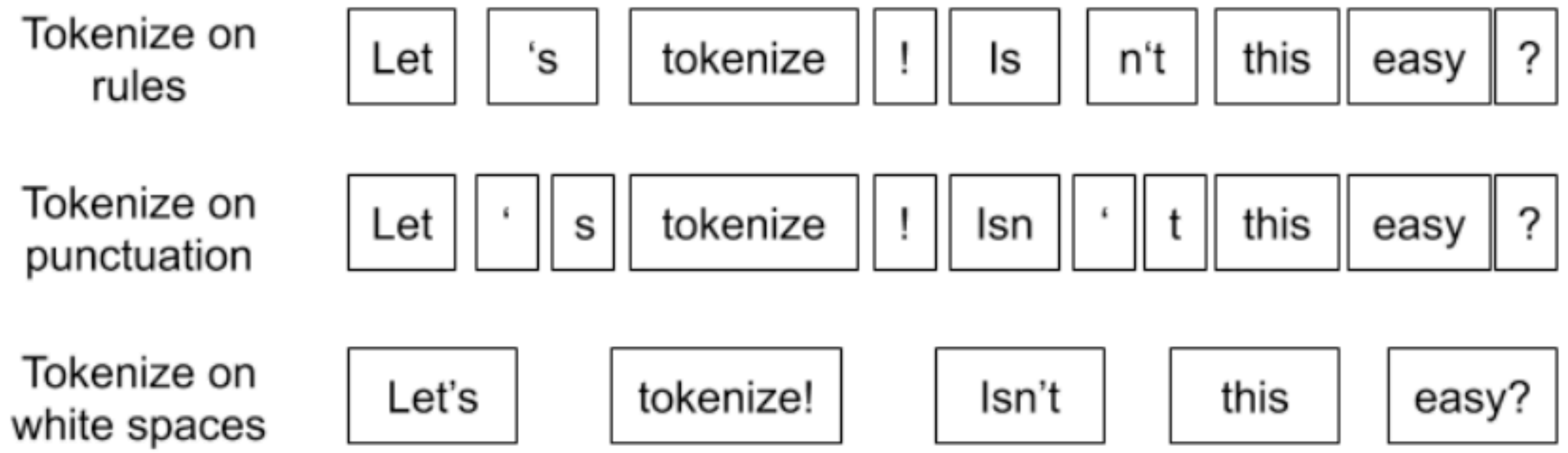
It consists only of splitting a sentence by the whitespace and punctuation marks.

- Raw text: I ate a burger, and it was good.
- Tokenized text: [’I’, ’ate’, ’a’, ’burger’, ‘,’, ‘and’, ’it’, ’was’, ’good’, ‘.’]

The difficulty in tokenization lies on how to **get the ideal split** so that all the tokens in the text have the **correct meaning**, and there are no left out tokens.

Interesting reading: https://blog.floydhub.com/tokenization-nlp/

Tokenization - Word level



Tokenize on rules	Let	‘s	tokenize	!	Is	n‘t	this	easy	?			
Tokenize on punctuation	Let	‘	s	tokenize	!	Isn	‘	t	this	easy	?	
Tokenize on white spaces	Let’s	tokenize!	Isn’t	this	easy?							

Let’s tokenize! Isn’t this easy?

Tokenization - Word level

Drawbacks of this approach:

- **You need a big vocabulary:** When you are dealing with word tokens you can only learn those which are in your training vocab. Any words not in the training set will be treated as unknown words when using the model and identified by the dreaded “<UNK>” token. So even if you learned the word “cat” in your training set, the final model would not recognize the plural “cats”.

- **We combine words:** The other problem is that there may be some confusion about what exactly constitutes a word. We have compounded words such as “sun” and “flower” to make sunflower and hyphenated words such as “check-in” or “runner-up”. Are these one word or multiple? And we use text sequences such as “New York” or “bachelor of science” as one unit. They may not make sense as isolated words.

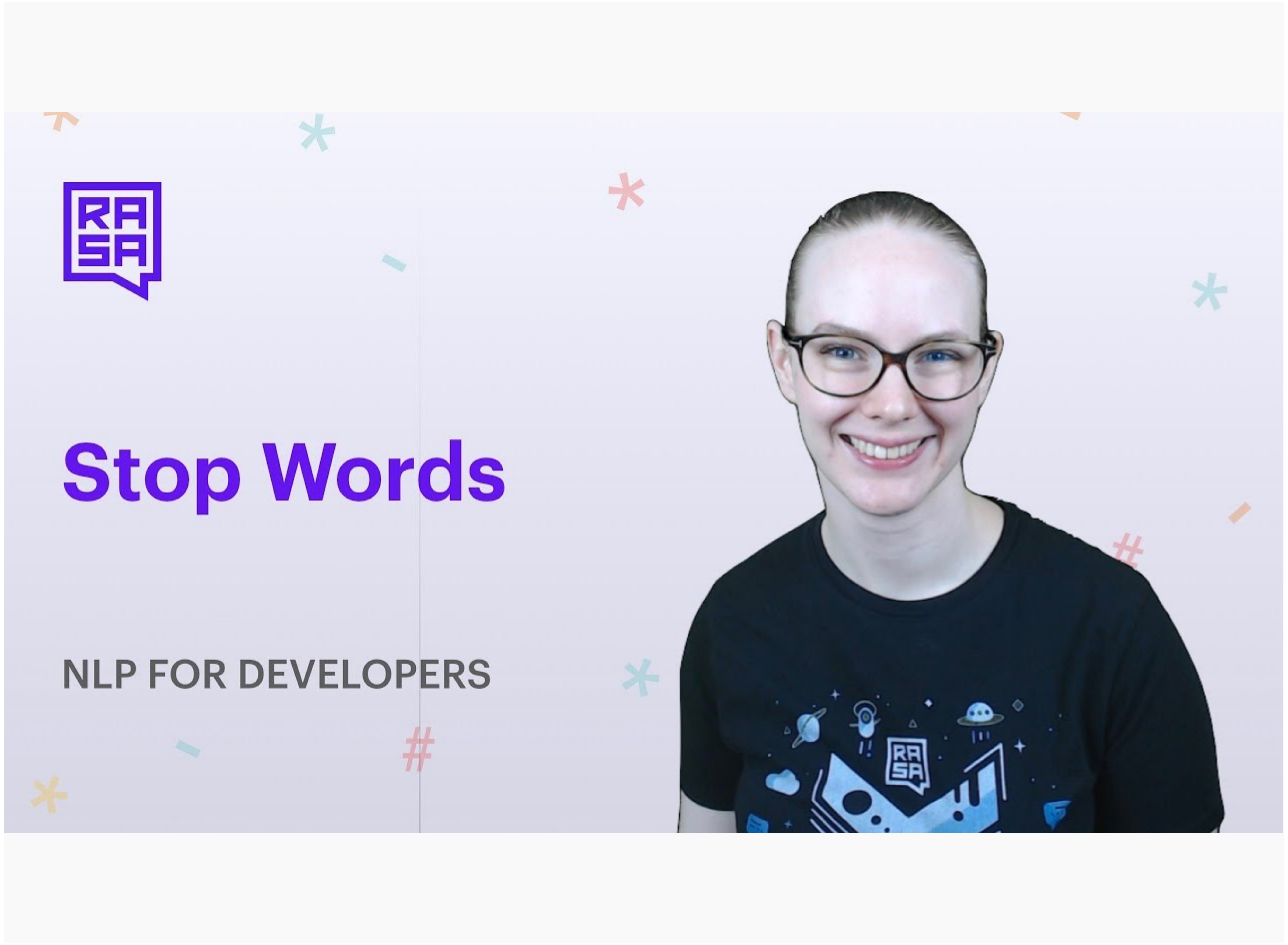
- **Abbreviated words:** With the rise of social media we have shorthand for words such as “LOL” or “IMO”. Are these collections of words or new words?

- **Some languages don’t segment by spaces:** It’s easy to break up english sentences into words for languages such as english which separate words by spaces but this is not true for all languages such as Chinese. In these cases word separation is not a trivial task.

Tokenization

Stop Words

A list of very common but uninformative **words that you want to ignore**



Interesting article: https://towardsdatascience.com/text-pre-processing-stop-words-removal-using-different-libraries-f20bac19929a

Stop Words

Stop words are available in abundance in any human language. By removing these words, we remove the low-level information from our text in order to give more focus to the important information

🙋 **Do we always remove stop words? Are they always useless for us?**

- Movie review: “The movie was not good at all.”
- Text after removal of stop words: “movie good”

We **do not always remove the stop word**s. The removal of stop words is highly dependent on the task we are performing and the goal we want to achieve.

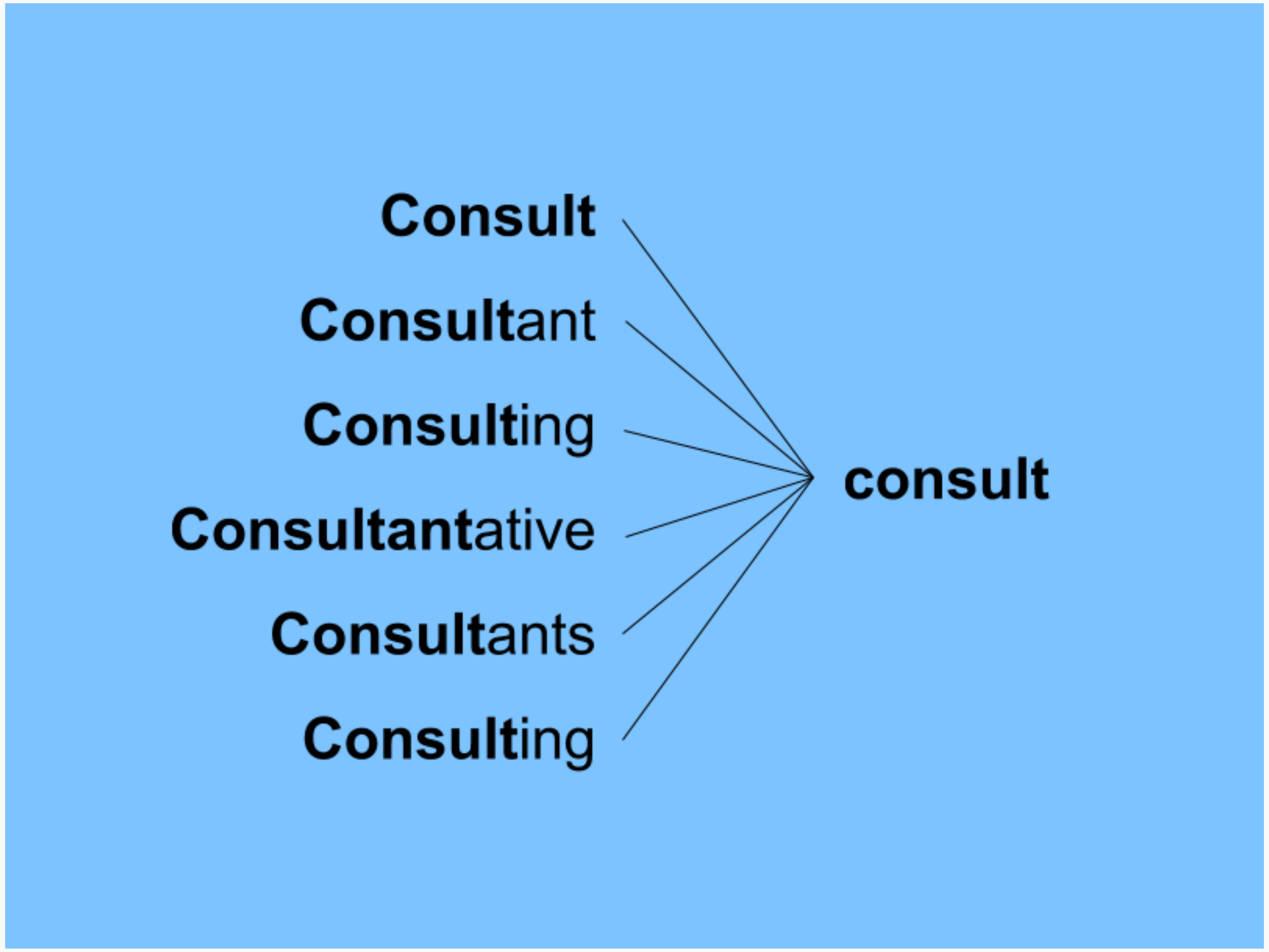
Stop Words

Stemming

Process of **reducing inflected (or derived) words to their root word** or word stem.

Applications of stemming:

- Stemming is used in information retrieval systems like search engines.
- It is used to determine domain vocabularies in domain analysis.



💡**Fun Fact:** Google search adopted a word stemming in 2003. Previously a search for "fish" would not have returned "fishing" or "fishes".

Stemmers - Stemming algorithms

Porter (1979)

It is based on the idea that the suffixes in the English language are made up of a combination of smaller and simpler suffixes.

Advantage: It produces the best output as compared to other stemmers and it has less error rate.

Limitation: Morphological variants produced are not always real words.

Example: EED -> EE means “if the word has at least one vowel and consonant plus EED ending, change the ending to EE” as ‘agreed’ becomes ‘agree’.

Lancaster (1990)

Very aggressive stemming algorithm, sometimes to a fault. With porter and snowball, the stemmed representations are usually fairly intuitive to a reader, not so with Lancaster, as many shorter words will become totally obfuscated.

Snowball (2001)

When compared to the Porter Stemmer, the Snowball Stemmer can map non-English words too. Since it supports other languages the Snowball Stemmers can be called a multi-lingual stemmer.

Stemmers extra reading: https://www.geeksforgeeks.org/introduction-to-stemming/

Stemming challenges

Overstemming

- Too much of the word is cut off (meaning lost)

Example:

University Universities Universal Universe	Univers

Understemming

- 2 words of same stem mapped to different stems

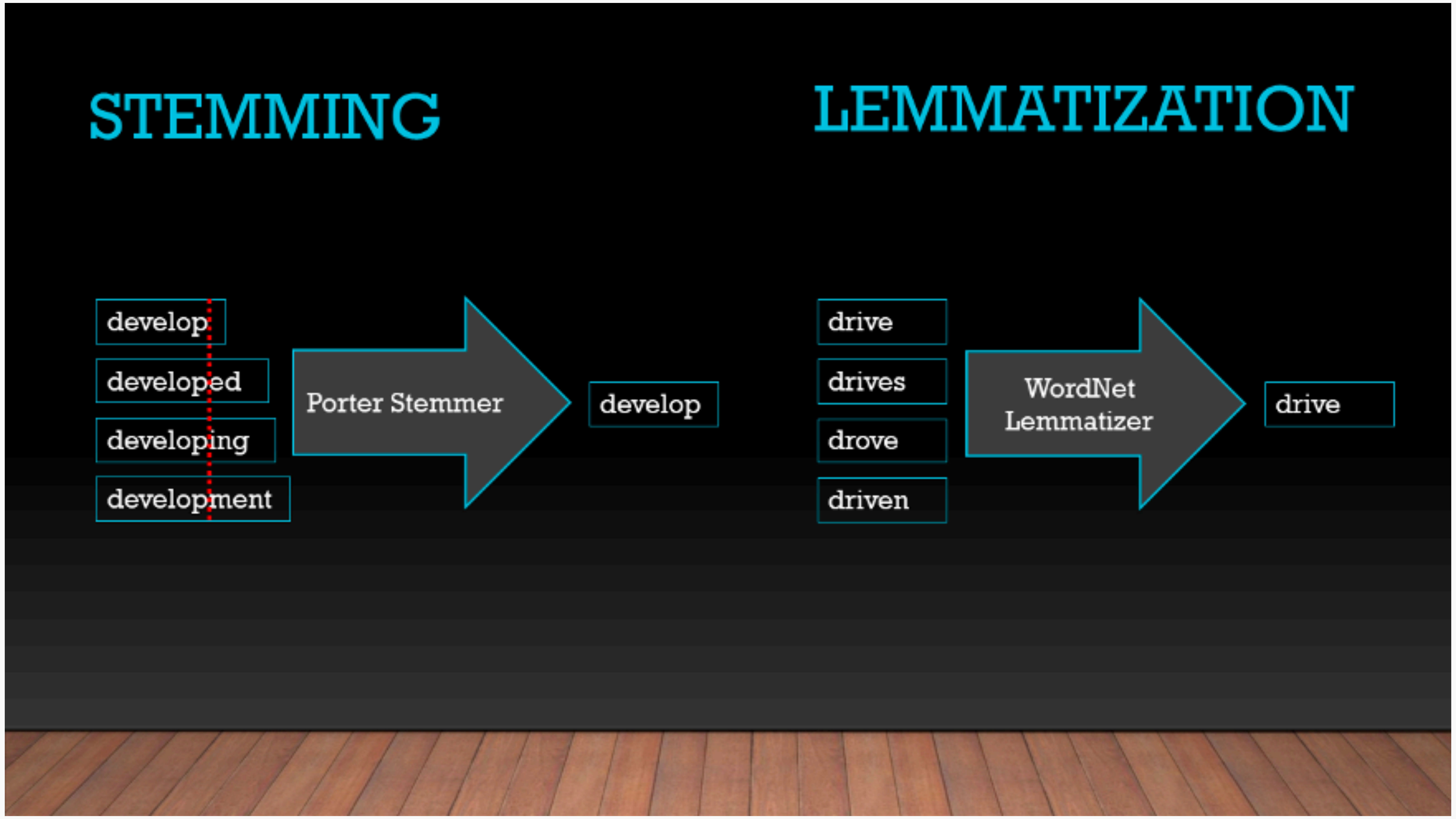
Example:

Data —> Dat
Datum —> Datu

Lemmatization

Lemmatization is a method responsible for **grouping different inflected forms of words into the root form**, having the same meaning.

It is similar to stemming, in turn, it gives the stripped word that has some dictionary meaning. The Morphological analysis would require the extraction of the **correct lemma of each word**.



Stemming vs Lemmatization

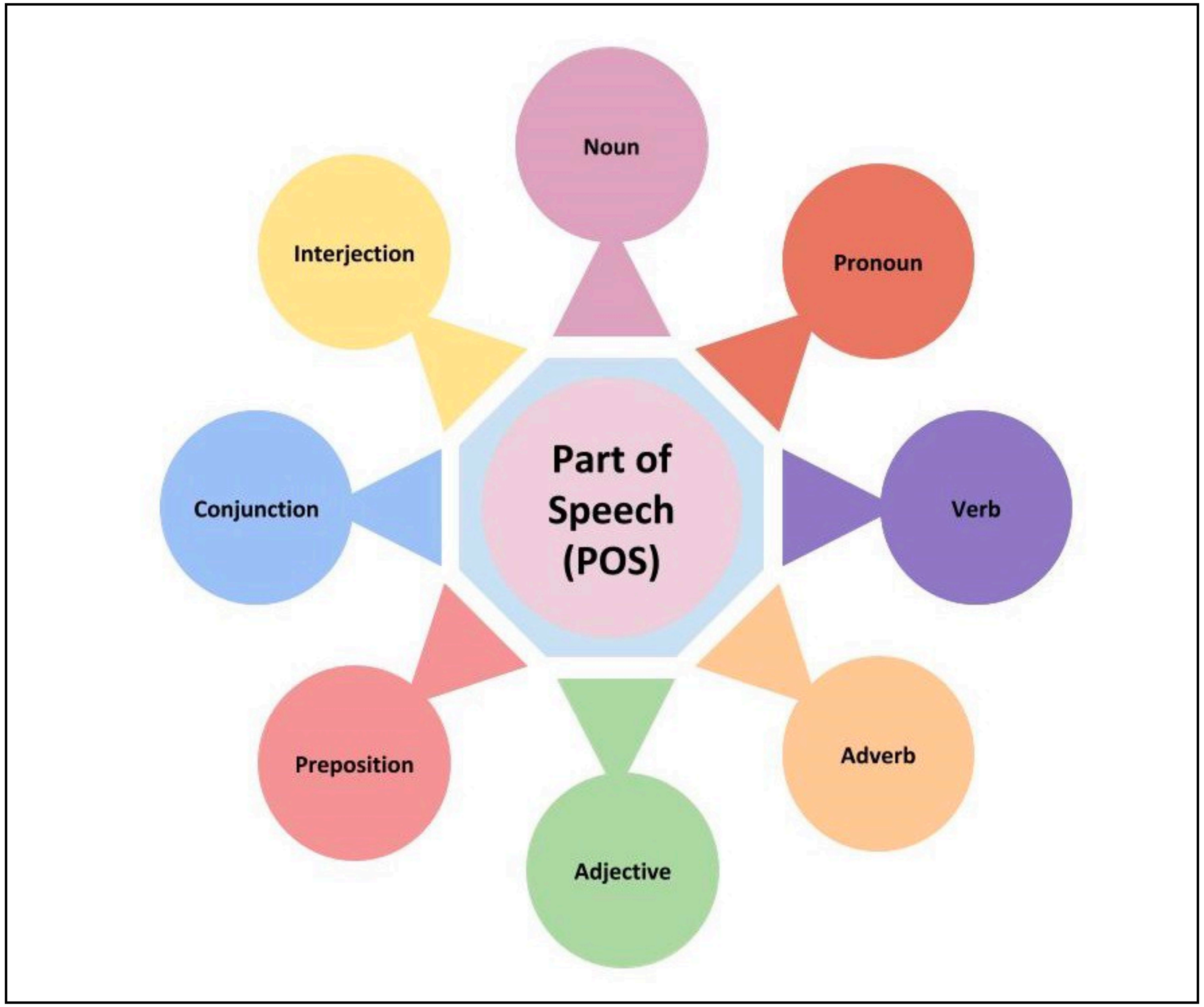
ID	Stemming	Lemmatization
1	Stemming is faster because it chops words without knowing the context of the word in given sentences.	Lemmatization is slower as compared to stemming but it knows the context of the word before proceeding.
2	It is a rule-based approach.	It is a dictionary-based approach.
3	Accuracy is less.	Accuracy is more as compared to Stemming.
4	When we convert any word into root-form then stemming may create the non-existence meaning of a word.	Lemmatization always gives the dictionary meaning word while converting into root-form.
5	Preferred when the meaning of the word is not important for analysis. Example: Spam Detection	Would be recommended when the meaning of the word is important for analysis. Example: Question Answer
6	For Example: “Studies” => “Studi”	For Example: “Studies” => “Study”

Interesting article: https://towardsdatascience.com/introduction-to-nlp-part-2-difference-between-lemmatisation-and-stemming-3789be1c55bc

Stemming
Lemmatization

Tagging

Part-of-speech (POS) tagging is a popular Natural Language Processing process which refers to **categorizing words in a text (corpus) in correspondence with a particular part of speech**, depending on the definition of the word and its context.



Tagging - POS Tagging Techniques

- **Lexical Based Methods — A**ssigns the POS tag the most frequently occurring with a word in the training corpus.

- **Rule-Based Methods —** Assigns POS tags based on rules. For example, we can have a rule that says, words ending with “ed” or “ing” must be assigned to a verb. Rule-Based Techniques can be used along with Lexical Based approaches to allow POS Tagging of words that are not present in the training corpus but are there in the testing data.

- **Probabilistic Methods —** This method assigns the POS tags based on the probability of a particular tag sequence occurring. Conditional Random Fields (CRFs) and Hidden Markov Models (HMMs) are probabilistic approaches to assign a POS Tag.

- **Deep Learning Methods** — Recurrent Neural Networks can also be used for POS tagging.

Tagging

Open for Innovation

KNIME

NLP on KNIME

- First steps on KNIME
- Sentiment analysis on KNIME
- Twitter analysis on KNIME
- BERT on KNIME

or others…

Hands on!